TESIS DE LA UNIVERSIDAD DE ZARAGOZA
2022
80

Idelkys Quintana Ramírez

Técnicas de optimización de parámetros de red para la mejora de la calidad de servicio en servicios IP

Director/es
Ruiz Más, José

ISSN 2254-7606

Prensas de la Universidad
Universidad Zaragoza

ISSN 2254-7606

Universidad Zaragoza
1542

Tesis Doctoral

TÉCNICAS DE OPTIMIZACIÓN DE PARÁMETROS DE RED PARA LA MEJORA DE LA CALIDAD DE SERVICIO EN SERVICIOS IP

Autor

Idelkys Quintana Ramírez

Director/es

Ruiz Más, José

UNIVERSIDAD DE ZARAGOZA
Escuela de Doctorado

Programa de Doctorado en Tecnologías de la Información y Comunicaciones en Redes Móviles

2022

Universidad Zaragoza

TÉCNICAS DE OPTIMIZACIÓN DE PARÁMETROS DE RED PARA LA MEJORA DE LA CALIDAD DE SERVICIO EN SERVICIOS IP

Tesis Doctoral

Idelkys Quintana Ramírez

Dirigida por Dr. José Ruíz Más

Programa de Doctorado en Tecnologías de la Información y Comunicaciones en Redes Móviles

Communications Networks and Information Technologies (CeNIT)

Instituto de Investigación en Ingeniería de Aragón (I3A)

UNIVERSIDAD DE ZARAGOZA

Zaragoza, enero 2022

Técnicas de optimización de parámetros de red para la mejora de la Calidad de Servicio en servicios IP.

Autor: Idelkys Quintana Ramírez
Director: Dr. José Ruíz Más

Texto impreso en Zaragoza
Primera edición, enero 2022

A Papi y Mami, por su infinito amor y por estar siempre.
A Luis, por ser motor impulsor de esta maquinaria.
A mi hermano, David y Diana, por ser fuentes de inspiración.

Resumen

Esta tesis doctoral presenta las contribuciones realizadas en la implementación de un sistema $5G$ que proporciona mecanismos de gestión, orquestación y monitorización. Dicho sistema brinda la posibilidad de desplegar diferentes escenarios como sistemas *Internet of Things*, *Internet of Skills*, *Video Survilleance Systems* y/o *Internet of Video Things*. La arquitectura propuesta colabora en la gestión dinámica de aplicaciones en un conjunto de nodos distribuidos. Además, se realiza una prueba de concepto implementando un sistema de videovigilancia inteligente en vehículos de transporte público basado en dispositivos de *Internet of Things*.

Por otro lado, el presente trabajo proporciona una metodología que brinda un procedimiento a seguir para modificar la forma de cómo el tráfico de servicios que generan ráfagas de paquetes pequeños es enviado a la red. La idea fundamental es que esta metodología sea aplicada en los nodos de cómputo en el borde de la nube, como si se tratase de funciones de red, aplicando el concepto de *Edge Cloud*. De esta manera, se pretende minimizar la congestión en los *buffer* de los dispositivos de acceso, optimizándose el tráfico de dichos servicios y por ende, mejorándose la experiencia del usuario final. En este contexto, se estudian dos métodos de optimización de tráfico: la multiplexión de varios paquetes pequeños en uno más grande, y el alisado, que reduce los picos de *throughput*.

Abstract

This dissertation presents a serie of contributions for implementing a $5G$ system that provides management, orchestration and monitoring mechanisms. The proposed system provides the possibility to deploy different scenarios such as Internet of Things, Internet of Skills, Video Survilleance Systems and/or Internet of Video Things. The suggested architecture collaborates in a dynamic management of applications across a set of distributed nodes. In addition, a proof-of-concept is performed by implementing an intelligent video surveillance system in public transport vehicles based on Internet of Things devices.

On the other hand, a methodology to mitigate the presence of bursty traffic is presented and it provides a procedure in order to modify the way the traffic of real-time services is sent to the network. The main idea is to apply optimization methods to every compute node at the edge of the cloud, as if they were network functions, applying the concept of Edge Cloud computing. In this way, we intent to minimize congestion in the buffer of the access devices, optimizing the traffic of underling services and thus improving the end-user experience. In this context, two traffic optimization methods are studied: multiplexing several small packets into a larger one and traffic shaping which reduces peaks of throughput.

Agradecimientos

El desarrollo de una tesis doctoral conlleva una serie de dificultades y retos que se deben superar; lo cual ha sido posible gracias a la participación de diversas personas e instituciones que han colaborado y facilitado los medios para que un trabajo de tal envergadura llegue a concluirse. Por ello, es para mí un deber utilizar este espacio para expresar mi agradecimiento.

A José Ruíz y Luis Sequeira por su infinita paciencia, invaluable e incondicional apoyo y guía en el desarrollo de esta tesis y en mi formación como investigadora. A Julián Fernández y José Saldaña por su inestimable ayuda, colaboración, revisiones y comentarios a lo largo de todos estos años.

Al *Banco Santander*, la *Universidad de Zaragoza* y el grupo de investigación *Communications Networks and Information Technologies* (CeNIT), y al *Centro de Investigación de Telecomunicaciones* perteneciente al *King's College London* por proveer los medios y el financiamiento necesario para la elaboración de este trabajo.

Muchas gracias,

Idelkys Quintana

enero 2022

Contribuciones científicas

Las siguientes publicaciones científicas se derivaron durante el proceso de esta investigación. La metodología y los resultados presentados en esta tesis están principalmente basados en ellas. Todas las publicaciones han pasado por una revisión por pares. Orden por año de publicación.

Publicaciones en revistas internacionales (Indexadas en JCR)

Idelkys Quintana, Luis Sequeira, José Ruiz-Mas, ***"An Edge-Cloud Approach for Video Surveillance in Public Transport Vehicles"***, IEEE Latin American Transactions, Vol. 19, Issue 10, October 2021.

Idelkys Quintana, Luis Sequeira, Julián Fernandez-Navajas, José Ruiz-Mas, Jose Saldana, ***"Minimizing the Impact of P2P-TV Applications in Access Links"***, IEEE Latin American Transactions, Vol. 17, Issue 2, February 2019.

Publicaciones en revistas internacionales

Idelkys Quintana-Ramirez, Anthony Tsiopoulos, Maria A. Lema, Fragkiskos Sardis, Luis Sequeira, James Arias, Aravindh Raman, Ali Azam, Mischa Dohler, ***"The Making of 5G: Building an End-To-End 5G-Enabled System"***, IEEE Communications Standards Magazine, Vol. 2, Issue 4, December 2018.

Publicaciones en congresos internacionales

Idelkys Quintana-Ramirez, Jose Saldana, José Ruiz-Mas, Luis Sequeira, Julián Fernández-Navajas, Luis Casadesus, ***"Optimization of P2P-TV Traffic by Means of Header Compression and Multiplexing"***, SoftCOM 2013, Split, Croatia, September 18-20, 2013. ISBN 978-953-290-041-5.

Luis Sequeira, Julián Fernández-Navajas, Luis Casadesus, Jose Saldana, **Idelkys Quintana**, José Ruiz-Mas, ***"The Influence of the Buffer in Packet Loss for Competing Multimedia and Bursty Traffic"***, Proc. International Symposium on Performance Evaluation of Computer and Telecommunication Systems SPECTS 2013, Toronto, Canada, July 2013, pp 645-652. ISBN 1-56555-352-7.

Luis Casadesus, Julián Fernández-Navajas, Luis Sequeira, **Idelkys Quintana**, Jose Saldana, José Ruiz-Mas, ***"IPTV Quality assessment system"***, IFIP/ACM LANC 2012 7th Latin America Networking Conference 2012, pp. 52-58. ISBN 978-1-4503-1750-4.

Luis Sequeira, **Idelkys Quintana**, Jose Saldana, Luis Casadesus, Julián Fernández-Navajas, José Ruiz-Mas, ***"The Utility of Characterizing the Buffer of Network Devices in order to Improve Real-time Interactive Services"***, IFIP/ACM LANC 2012 7th Latin America Networking Conference 2012, pp. 19-27. ISBN 978-1-4503-1750-4.

Publicaciones en congresos nacionales

Idelkys Quintana-Ramirez, Jose Saldana, José Ruiz Mas, Luis Sequeira, Julián Fernández Navajas, Luis Casadesus, ***"Optimización del Tráfico P2P-TV mediante el uso de Técnicas de Compresión y Multiplexión"***, Jornadas de Ingeniería Telemática JITEL 2013, pp 345-350, Granada, Spain, October 28-30, 2013. ISBN 978-84-616-5597-7.

Idelkys Quintana-Ramirez, Jose Saldana, José Ruiz Mas, Julián Fernández Navajas, Luis A. Casadesus Pazos, Luis Sequeira. ***"Influencia del Buffer del Router en la Distribución de Video P2P-TV"***, Actas del XXVII Simposium Nacional de la Union Científica Internacional de Radio (URSI 2012). Elche (Spain). Sept. 2012. ISBN 978-84-695-4327-6.

Índice general

Índice de figuras

Acrónimos

$3GPP$	*3^{rd} Generation Partnership Project*.
$4G$	*4^{th} Generation Mobile Network*.
$5G$	*5^{th} Generation Mobile Network*.
ACK	*Acknowledgment*.
ADSL	*Asymmetric Digital Subscriber Line*.
AMQP	*Advanced Message Queuing Protocol*.
APIs	*Application Programming Interfaces*.
AWS	*Amazon Web Services*.
BW	*Bandwidth*.
BWS	*Bandwidth Saving*.
C-RAN	*Cloud Radio Access Network*.
CEE	*Cloud Execution Environment*.
CID	*Context Identifier*.
CoAP	*Constrained Application Protocol*.
CPU	*Central Processing Unit*.
CSP	*Cloud Service Provider*.
DDS	*Data Distribution Service*.
DPDK	*Data Plane Development Kit*.
E2E	*End-to-End*.
ECDF	*Empirical Cumulative Distribution Function*.
eNB	*E-UTRAN NodeB*.
ETSI	*European Telecommunications Standards Institute*.

FTP	*File Transfer Protocol*.
HAG	*Hybrid Access Gateway*.
HTTP	*Hypertext Transfer Protocol*.
HTTPS	*Hypertext Transfer Protocol Secure*.
IEEE	*Institute of Electrical and Electronics Engineers*.
IoS	*Internet of Skills*.
IoT	*Internet of Things*.
IoVT	*Internet of Video Things*.
IP	*Internet Protocol*.
IPHC	*IP Header Compression*.
IPTV	*Internet Protocol Television*.
ISP	*Internet Service Provider*.
JSON	*JavaScript Object Notation*.
JTG	*Jugi's Traffic Generator*.
KCL	*King's College London*.
KPI	*Key Performance Indicator*.
L2TP	*Layer* 2 *Tunneling Protocol*.
LAN	*Local Area Network*.
LTE	*Long Term Evolution*.
MANO	*Management and Orchestration*.
MEC	*Multi-access Edge Computing*.
MIMO	*Multiple-Input Multiple-Output*.
MJPEG	*Motion JPEG*.
ML	*Machine Learning*.
mMTC	*massive Machine-Type Communication*.
mmWave	*millimeter-wave*.
MPTCP	*Multi-Path TCP*.
MQTT	*MQ Telemetry Transport*.
MTU	*Maximum Transmission Unit*.
NAPA-WINE	*Network-Aware P2P-TV Application over Wise Networks*.
NB-IoT	*Narrow-Band-IoT*.
NBI	*Northbound Interface*.
Netem	*Network Emulator*.
NFV	*Network Function Virtualization*.

NGMN	*Next Generation Mobile Networks*.
NS	*Network Simulator*.
OAI	*OpenAPI*.
OSM	*Open Source Mano*.
OvS	*Open vSwitch*.
P2P	*Peer-to-Peer*.
P2P-TV	*Peer-to-Peer TV*.
pps	paquetes por segundo.
QoE	*Quality of Experience*.
QoS	*Quality of Service*.
RAN	*Radio Access Network*.
RO	*Resource Orchestrator*.
ROHC	*Robust Header Compression*.
RTP	*Real-time Transport Protocol*.
RTT	*Round-Trip Time*.
SBI	*Southbound Interface*.
SDN	*Software Defined Networking*.
SDR	*Software Defined Radio*.
SNS	*Simple Notification Service*.
SO	*Service Orchestrator*.
SRIOV	*Single-Root I/O Virtualization*.
TCM	*Tunneling, Compressing and Multiplexing*.
TCP	*Transport Control Protocol*.
TV	*Television*.
UDP	*User Datagram Protocol*.
UI	*User Interface*.
vCore	*virtualized Core*.
VIM	*Virtual Infrastructure Manager*.
VLAN	*Virtual Local Area Networks*.
VLC	*VideoLAN Client*.
VM	*Virtual Machine*.
VNF	*Virtual Network Function*.
VNFD	*Virtual Network Function Descriptor*.
VoIP	*Voice over Internet Protocol*.

vRAN	*virtualized RAN*.
VSS	*Video Surveillance System*.
Wi-Fi	*Wireless Fidelity*.
WLAN	*Wireless LAN*.
XMPP	*Extensible Messaging and Presence Protocol*.

Me enseñaron que el camino del progreso no era rápido ni fácil.

Marie Curie

CAPÍTULO

1

Introducción

1.1 Introducción

Las redes móviles 5^{th} *Generation Mobile Network* ($5G$) en conjunto con el *Internet of Things* (IoT) proporcionan soluciones innovadoras para una amplia gama de casos de uso. La flexibilidad ofrecida por las actuales infraestructuras virtualizadas, softwarizadas y *multitenant*, al igual que el alto rendimiento que promete la tecnología $5G$, son claves para afrontar el despliegue de las aplicaciones IoT actualmente demandadas por diversos sectores. Los casos de uso de IoT en $5G$ suponen un reto en cuanto a requisitos de *Quality of Service* (QoS), en particular la interconexión de millones de dispositivos físicos IoT a través de Internet, para lograr una mayor eficiencia en sistemas *massive Machine-Type Communication* (mMTC) [1]. De la misma manera, la segmentación de las redes más conocida por su terminología en inglés como *network slicing*, es una tecnología imprescindible en las redes $5G$, ya que permite crear redes lógicas virtualizadas, ofreciendo soluciones personalizadas con el fin de satisfacer diversos requisitos de QoS.

Los sistemas IoT consisten en la interconexión de dispositivos físicos con la funcionalidad de detección, monitorización y procesamiento, trabajando cooperativamente para ofrecer servicios. Las ciudades y edificios inteligentes, los vehículos autónomos, la telemedicina, el control y gestión de viviendas, la Industria 4.0, entre otros, son algunos ejemplos en donde el paradigma IoT ya ha sido implantado. Debido a la gran cantidad de datos producidos y el creciente tráfico generado

en la red, la eficiencia en IoT se ha convertido en un foco importante de atención tanto para la academia como para la industria [2].

Entre los diferentes tipos de dispositivos en IoT, las cámaras juegan un papel clave, puesto que pueden capturar gran cantidad de contenidos con información relevante. En los últimos años, el número de cámaras integradas ha ido aumentado exponencialmente, dando lugar al término *Internet of Video Things* (IoVT). En este contexto, el uso de *Video Surveillance System* (VSS) ha ido ganando terreno en el ámbito de las ciudades inteligentes, con el propósito de reducir la tasa de criminalidad, el crimen organizado y el terrorismo. Estos sistemas son ampliamente utilizados en centros de transporte público como aeropuertos, estaciones de trenes y autobuses. Hoy en día, el reconocimiento de imágenes juega un papel muy importante en muchos VSSs ya que no sólo permite realizar reconocimientos faciales, sino que además, ayuda a monitorizar, detectar e interpretar actividades. Los futuros VSSs deben ser confiables, escalables, integrados con tecnologías emergentes como la $5G$ y aprovechar las ventajas del modelo *Network Function Virtualization* (NFV).

Por otro lado, muchos estudios han indicado que las tecnologías *Edge Cloud* ofrecen muchas ventajas para los servicios IoT [3, 4]. A consecuencia de la alta transferencia de datos y un rendimiento limitado de las redes, el núcleo de las infraestructuras de nube no son lo suficientemente eficaces para procesar y analizar las grandes cantidades de datos recolectados por los dispositivos IoT [5]. La mayoría de los dispositivos IoT tienen suficiente capacidad para realizar cálculos relativamente complejos, por lo tanto, pueden ser adecuados para usarse como nodos de cómputo en despliegues *Edge Cloud*. Las tecnologías *Edge Cloud* se centran principalmente en ampliar las capacidades de la nube [6] y proveer servicios con una latencia más reducida entre el nodo que procesa la información y el usuario final [7]. Además, puede presentar una mejora significativa del ancho de banda, el consumo de energía, proveer cierto grado de privacidad y mejorar la disponibilidad de la aplicación, en caso de una falla en la nube [8].

Paralelamente, al mismo tiempo que los dispositivos IoT generan mucha información en las redes, existen muchas otras aplicaciones y servicios en Internet

que transmiten gran cantidad de tráfico a la red, lo cual se intensifica en grandes despliegues. Bajo estas condiciones, la concatenación de diferentes flujos de tráfico en dispositivos intermedios (o incluso en el mismo dispositivo) puede crear cuellos de botella que producen pérdida de paquetes y afectan negativamente la QoS. De igual manera, la gran demanda de los servicios multimedia en tiempo real como son los juegos *online*, *Voice over Internet Protocol* (VoIP), *Peer-to-Peer TV* (P2P-TV), videoconferencia o videovigilancia, aumenta significativamente el volumen de tráfico en la red. Asimismo, los usuarios exigen que estos servicios multimedia en tiempo real estén disponibles con alta calidad, independientemente de las tecnologías de acceso.

Como es de suponer, cada servicio tiene su propio patrón de tráfico que depende de la naturaleza y el tamaño de la información transportada. Algunas aplicaciones, como la videovigilancia y el *videostreaming*, inyectan ráfagas de tráfico en la red, generando grandes volúmenes de información en períodos muy cortos. En estos casos, el *buffer* de los dispositivos de red puede saturarse dando como resultado la ocurrencia de períodos de congestión que provocan una significativa pérdida de paquetes, con la posterior reducción de QoS para los usuarios. Por consiguiente, los enlaces de acceso pueden convertirse en puntos sensibles y producir cuellos de botella en la red debido a su capacidad limitada. Otro aspecto que puede convertirse en una causa del deterioro de la QoS es la concurrencia de diversos flujos de datos en un solo enlace de acceso; de hecho, la combinación del tráfico de varios servicios puede dar lugar a ráfagas de paquetes que eventualmente conducirían a problemas de congestión en ciertos puntos de la red [9].

Para minimizar el impacto de estos picos de tráfico en la red, se han propuesto algunas soluciones, por ejemplo, multiplexar los paquetes pequeños en bloques más grandes [10], aumentando así la eficiencia de los flujos de tráfico y reduciendo el número de paquetes por segundo. Otra posibilidad es el uso de técnicas de conformado de tráfico, que establecen un rendimiento máximo para ráfagas de tráfico, y por lo tanto, pueden reducir el nivel de pérdida de paquetes. Como contrapartida, ambas técnicas introducen retardos, por lo que aparece una disyuntiva entre los nuevos retardos añadidos y la reducción de la pérdida de paquetes obtenida.

1.2 Metas y contribuciones: objetivos

Con el auge actual de la tecnología $5G$ y su integración con diferentes tipos de redes de acceso, se precisa desarrollar procesos que garanticen de manera eficiente la gestión y orquestación de estas plataformas, basándose en sus funcionalidades de softwarización, virtualización y segmentación de la red (*network slicing*); de manera tal que se permita gestionar de manera aislada y segura diversos servicios. Por este motivo, una de las metas en esta tesis doctoral es contribuir a la implementación de un sistema $5G$ que proporcione mecanismos de gestión, orquestación y monitorización. Dicho sistema debe brindar la posibilidad de desplegar diferentes escenarios como sistemas IoT, *Internet of Skills* (IoS), VSS y/o IoVT, entre otros. En este sentido la propuesta es desarrollar una prueba de concepto para un VSS inteligente en vehículos de transporte público basado en dispositivos IoVT. La arquitectura heurística propuesta para este caso de uso tiene como objetivo gestionar dinámicamente diversas aplicaciones dentro de un conjunto de nodos distribuidos. De este modo, se logrará minimizar la latencia a la hora de procesar datos, hacer un mejor uso del ancho de banda disponible y reducir las pérdidas durante períodos de congestión.

Por otro lado, como se ha mencionado anteriormente, la concurrencia de varios flujos pueden provocar ráfagas de paquetes con tiempo entre paquetes muy pequeños. Bajo estas circunstancias, los *buffer* de los *router* de las redes de acceso pueden provocar cuellos de botella durante períodos de congestión, ocasionando la pérdida de paquetes y por tanto, un deterioro de la QoS. En este sentido, una contribución del presente trabajo es proporcionar una metodología que brinde un procedimiento a seguir para modificar la forma de cómo el tráfico de servicios que generan ráfagas de paquetes pequeños es enviado a la red. De esta manera, se podrá minimizar la congestión en los *buffer* de los dispositivos de acceso, optimizándose el tráfico de dichos servicios y por ende, mejorándose la experiencia del usuario final. Además, se estudian dos métodos de optimización de tráfico: la multiplexión de varios paquetes pequeños en uno más grande, y el alisado, que reduce los picos de *throughput* en la red.

Para conseguir todo lo comentado se plantean los siguientes objetivos:

1. Diseñar mecanismos de gestión, orquestación y monitorización para servicios y aplicaciones IoT, y su integración en un sistema $5G$.

2. Desarrollar una prueba de concepto para un VSS inteligente aplicado a vehículos de transporte público, que permita la gestión de diferentes aplicaciones en un conjunto distribuido de dispositivos IoT.

3. Proponer una metodología que facilite el análisis e identificación de los parámetros que influyen en el deterioro de la QoS y que permita la integración con modelos de conformado de tráfico para su optimización.

4. Analizar el efecto que puede tener el *buffer* de los *router* de acceso frente a diferentes tráficos de servicios en tiempo real. Además de ello, evaluar el impacto en la QoS cuando flujos de datos, principalmente a ráfagas, convergen en un mismo enlace de acceso.

5. Analizar la eficiencia en términos de parámetros objetivos de QoS (pérdida de *bytes*, pérdida de paquetes, retardo y *throughput*) de un servicio en tiempo real, cuando se aplican dos métodos de conformado de tráfico: uno con base en multiplexión y otro en alisado.

1.3 Estructura de la tesis

En cuanto a la estructura de la tesis, en este capítulo se presenta una introducción que incluye la motivación y presentación del problema junto con los objetivos de la investigación.

El capítulo 2 comprende el estado del arte de los temas tratados en el presente trabajo, donde se realiza un estudio relacionado con los nuevos paradigmas que permiten la softwarización y virtualización de redes (NFV y *Software Defined Networking* (SDN)). Además, se detallan algunas de las arquitecturas de servicios multimedia en tiempo real, así como también sus principales limitaciones en las redes de acceso. Por último, se exponen algunas técnicas de optimización del tráfico, que pueden utilizarse para modificar y conformar los flujos en Internet, mejorando la QoS de la red.

En el capítulo 3 en primer lugar, se describen las arquitecturas propuestas tanto para el sistema $5G$ extremo a extremo que se implementa en esta tesis, así como también para un caso de uso basado en un sistema de videovigilancia inteligente en el sistema de transporte público. En segundo lugar, se exponen las metodologías para la optimización del tráfico mediante técnicas de conformado de tráfico, utilizados posteriormente en las pruebas y simulaciones presentadas.

El capítulo 4 muestra las pruebas y simulaciones realizadas, así como también los resultados obtenidos cuando se aplican técnicas de conformado de tráfico a una aplicación P2P-TV, que genera un gran flujo de paquetes pequeños a ráfagas. De igual manera, se estudiaron algunas de las características más significativas del tráfico de estas aplicaciones de *videostreaming* en tiempo real y la influencia de la implementación del *buffer* del *router* en el comportamiento de las mismas.

En el capítulo 5 se presenta en primer lugar, una solución de gestión y orquestación de servicios de red y la puesta en funcionamiento de una plataforma de virtualización para un sistema $5G$. En segundo lugar, se detalla la implementación de la arquitectura inteligente para VSS, y se realizan simulaciones con el tráfico proveniente de una aplicación P2P-TV cuando comparte el mismo enlace acceso con un tráfico de fondo. En este caso, se aplica una técnica de conformado de tráfico y se estudian algunos parámetros de QoS (pérdida de *bytes* y paquetes, retardo y *throughput*), que permitirán evaluar el rendimiento de la red.

Finalmente, en el capítulo 6 se describen las conclusiones de la presente tesis y se proponen una serie de líneas para futuras investigaciones.

Hay que sentirse dotado para realizar alguna cosa y esa cosa hay que alcanzarla, cueste lo que cueste.

Marie Curie

CAPÍTULO

2

Estado del arte

En este capítulo se tratarán diferentes temas que serán objeto de estudio en la presente tesis. En primer lugar, se presentan los paradigmas tecnológicos claves que permiten la transformación a las arquitecturas de redes móviles de nueva generación, especialmente la softwarización de las redes haciendo uso de tecnologías NFV y SDN. Posteriormente, se describe algunas de las arquitecturas de servicios en tiempo real ampliamente utilizadas, muchas de ellas con estrictos requerimientos temporales. Además, se explican algunas características importantes a tener en cuenta de las redes de acceso, principalmente sus limitaciones. Para finalizar, se exponen algunas técnicas de optimización del tráfico, principalmente aquellas utilizadas para modificar y conformar el tráfico de servicios que generan ráfagas de muchos paquetes pequeños. Los temas expuestos en este capítulo han sido utilizados para influenciar el diseño, arquitectura y los mecanismos de control de tráfico que se presentan a lo largo de esta tesis.

2.1 Softwarización de red: NFV y SDN

La softwarización de redes permite crear de forma flexible redes virtuales sobre una única red física (*network slicing*). Este es un enfoque que implica el uso de *software* para diseñar, implementar, desplegar, gestionar y mantener equipos, componentes y servicios de red [11, 12]. Esta funcionalidad tiene como objetivo brindar servicios y aplicaciones $5G$ con mayor agilidad y rentabilidad. Paralelamente

a la implementación de los requisitos de una red $5G$ (como por ejemplo, programabilidad, flexibilidad y adaptabilidad), la softwarización de la red está pensada para proporcionar una mejor gestión de servicios *End-to-End* (E2E) y mejorar la *Quality of Experience* (QoE) del usuario final [13, 14]. Gracias a la softwarización de la red y a la virtualización utilizando NFV, SDN y tecnologías de nube, es posible la utilización del *network slicing as-a-service* [15] y la unificación de una plataforma E2E de servicios para $5G$.

En la actualidad, los ecosistemas de IoT y la softwarización de las redes se han convertido en tecnologías imprescindibles para la gestión de la llamada *"Internet de Nueva Generación"*. Por este motivo, la infraestructura física de los sistemas de redes heterogéneos ha ido ganando en complejidad, y por tanto, requiere soluciones eficientes y dinámicas para la gestión, configuración y programación de flujos. La softwarización de la red por medio de arquitecturas NFV y SDN, fundamentalmente para implementaciones IoT, han sido objeto de investigación desde hace ya algunos años. En [16], se presenta un estudio sobre algunas técnicas de virtualización de redes, diseñadas explícitamente para redes IoT, resaltando desafíos a corto y largo plazo, así como también algunas áreas o temas pendientes para la adopción de un sistema IoT basado en *software* [17].

En [18], se presentó el diseño de una arquitectura $5G$ que permite el *network slicing*, utilizando los conceptos de NFV y SDN. En dicha propuesta, se hizo énfasis en las técnicas que proporcionan una utilización eficiente de recursos, optimizándose el rendimiento de la red. Los autores tuvieron en cuenta conceptos fundamentales como son la adaptación a los servicios móviles, el control de QoS/QoE y la orquestación de la red. Además, se señaló la necesidad de abordar temas relacionados al control de la granularidad y la gestión de los segmentos de red.

Una sinopsis sobre las potencialidades de utilizar NFV y SDN en redes $5G$ puede encontrarse en [19], donde se explica cómo ambas tecnologías se complementan entre sí. Asimismo, se describe el concepto de *network slicing* en $5G$ y los retos que esto trae consigo. Se mencionaron algunos de los desafíos que no han sido resueltos para garantizar el correcto funcionamiento de la red $5G$ propuesta. Por otro lado, en [15] se introduce el concepto de *"network slicing as-a-service"*

con el fin de proveer servicios personalizados; y se presenta la orquestación de servicios y un acuerdo a nivel servicios para demostrar la funcionalidad de la arquitectura de gestión de servicios entre diferentes despliegues. Sin embargo, debida a la constante evolución en los requerimientos en $5G$, en este trabajo los autores sugieren mejorar las tecnologías NFV/SDN para desarrollar mejores entornos de comunicación.

En [20], los autores presentaron un despliegue óptimo de *Virtual Network Function* (VNF)s para el *network slicing* y la asignación de recursos, teniendo en cuenta los requisitos del servicio $5G$. Con el modelo propuesto se pudo aprovechar al máximo los recursos de la nube, por medio de una distribución efectiva de la carga de red y la utilización de los recursos computacionales donde fuera necesario. Con esto se mejoró la eficiencia en el uso de recursos y el número de cadenas de VNFs soportadas por el sistema en términos de soluciones estáticas. A pesar de los resultados obtenidos y la eficiencia alcanzada, no se tuvieron en cuenta los límites realistas de esta implementación, como por ejemplo los retrasos en la configuración, la creación de instancias y la orquestación.

2.2 Arquitecturas de servicios en tiempo real

2.2.1 IoVT en videovigilancia

A pesar de la gran cantidad de trabajos de investigación y despliegues de sistemas IoT, no hay un marco único o protocolo a seguir sobre cómo implementar una arquitectura para dichos sistemas. Existe una tendencia de utilizar arquitecturas de tres, cuatro y hasta cinco capas [21] (sensores, red, servicios, aplicación y negocios) en función de la complejidad de los servicios que se desea proveer [22, 23]. En [24], se presenta una arquitectura de capas para la implementación de servicios IoT, el modelo consiste en un conjunto de herramientas de *hardware* y *software* que se utiliza en el despliegue de servicios. Mediante el uso de una interfaz gráfica, los usuarios pueden realizar despliegues de infraestructuras IoT. Sin embargo, dicha propuesta no contempla mecanismos para la integración con otras arquitecturas basadas en modelos NFV.

Por otro lado, en [25] se propone una una arquitectura multinivel que provee *Fog as-a-service* para servicios IoT extremo a extremo. El modelo expuesto hace uso de tecnologías SDN para separar los planos de datos y control, lo cual es un punto a favor para poder interconectar con otros sistemas NFV, pero carece de un despliegue real por lo que su viabilidad no ha sido validada. En [26], se discuten las características de una plataforma distribuida que proporciona servicios de cómputo, almacenamiento y red, con la finalidad de mejorar la latencia al procesar ciertas tareas en dispositivos IoT. La arquitectura se enfoca en servicios para el cuidado de la salud y no contempla dispositivos IoVT, por lo que su uso para la implementación de un VSS u otros servicios de video no parece factible.

Un VSS estable y eficiente, basado en IoVT, necesita protocolos confiables a nivel de acceso y servicios. Los dispositivos IoT incluyen, entre otros, sensores visuales o cámaras que son utilizados por aplicaciones de diferentes índoles. Estas aplicaciones pueden transmitir información al mismo tiempo y utilizar diferentes tipos de protocolos para el acceso, enlace o transporte. *Institute of Electrical and Electronics Engineers* (IEEE) proporciona una amplia gama de estándares, como IEEE 802.3 (Ethernet), IEEE 802.11 (Red de área local inalámbrica) e IEEE 802.15.4 (Red de área personal inalámbrica). IEEE 802.15.4 es un protocolo muy popular entre los sistemas IoT, el cual está diseñado para transmisión de datos de baja potencia y baja velocidad (máximo 250 $kbps$) [27]. Sin embargo, los dispositivos IoVT envían una gran cantidad de datos a la red y requieren velocidades de transmisión más altas. En este sentido, IEEE 802.3 e IEEE 802.11 serán una opción más eficiente en un VSS.

Parte del diseño de una arquitectura consiste en una adecuada selección de tecnologías y protocolos para su implementación. Para tal efecto, existe una amplia gama de protocolos de aplicación utilizados en los sistemas IoVT, entre los cuales se pueden mencionar *MQ Telemetry Transport* (MQTT), *Extensible Messaging and Presence Protocol* (XMPP), *Advanced Message Queuing Protocol* (AMQP), *Data Distribution Service* (DDS), *Constrained Application Protocol* (CoAP) [28] y *Hypertext Transfer Protocol* (HTTP)/*Hypertext Transfer Protocol Secure* (HTTPS) [29]. La selección de los protocolos de transporte (es decir, *User Datagram Protocol* (UDP) o *Transport Control Protocol* (TCP)) depende de cómo estas aplica-

ciones o servicios hayan sido implementados. Por ejemplo, CoAP y DDS utilizan UDP para el transporte de datos, mientras que el resto usa TCP. En este sentido, algunos de los algoritmos de control de congestión proporcionarán un mejor rendimiento, menos pérdida de paquetes o más retardo bajo determinas condiciones o niveles de congestión [30]. Otro aspecto que podría afectar la QoS, en términos de *throughput* y pérdida de paquetes, es la presencia de ráfagas de paquetes en el tráfico de aplicaciones, tales como *streaming* de videovigilancia y videoconferencia, produciendo una degradación de la calidad [9].

2.2.2 Servicios P2P-TV

Las aplicaciones P2P-TV permiten la disponibilidad de contenido de video, perdiendo cierto grado de QoS con respecto a otros servicios como pueden ser la *Television* (TV) bajo demanda o el modelo cliente-servidor utilizado por *Internet Protocol Television* (IPTV). Las aplicaciones P2P-TV generalmente generan tráfico que consiste en paquetes grandes, asociados con paquetes de video, y altas tasas de paquetes pequeños, asociados con el tráfico de control (por ejemplo, *Acknowledgment* (ACK) del nivel de aplicación) [31, 32]. Esto da como resultado una mezcla de tráfico que incluye paquetes pequeños y grandes a través de Internet. En estos casos, la capacidad de procesamiento del *buffer* puede constituir un cuello de botella cuando tiene que gestionar demasiados paquetes por segundo [33]. Existen estudios [34] en los que se demuestra que los paquetes de video pueden ser penalizados por altas cantidades de paquetes pequeños, aumentando la probabilidad de ser descartados. En consecuencia, el comportamiento de los *peer* dentro de una estructura *Peer-to-Peer* (P2P) no sería el esperado.

Muchos investigadores han centrado sus esfuerzos en la caracterización del tráfico y el comportamiento de diferentes aplicaciones P2P-TV, entre las que se pueden mencionar: *PPLive, TVAnts, Coolstreaming, SopCast y PPStream*. En cuanto a las soluciones P2P-TV existentes hay muchos y variados estudios sobre el impacto de su tráfico en las redes de comunicaciones [32, 35, 36, 37], las mejoras que aporta a los operadores de redes IPTV [38], la QoE proporcionada [39, 40, 41], los algoritmos de distribución alternativos [42], el creciente número de usuarios que

hacen uso de ellas [43], entre otros temas. Todas estas investigaciones abordan el análisis y mejora de los servicios P2P sin tener en cuenta que las ventajas de colaboración entre usuarios, como el ahorro de ancho de banda y costes asociados al servicio, pueden ser cercenadas por los propios dispositivos de acceso del usuario y sean sólo unos pocos usuarios "privilegiados" de la red P2P los que participen activamente en la distribución de contenidos.

En la literatura se pueden encontrar diferentes trabajos centrados en la caracterización del tráfico, incluyendo las velocidades de subida (*upload*) y de bajada (*download*) de contenidos, el tipo de paquetes intercambiados y los protocolos empleados [32], así como los mecanismos empleados en algunas aplicaciones P2P-TV. Otros trabajos se centran en las características y propiedades de las estructuras de las redes P2P [44] y las implicaciones del tráfico P2P-TV para los *Internet Service Provider*s (ISPs) [45], [46]. Sin embargo, en ninguno de los trabajos citados, centran su atención en la optimización del tráfico y su efecto en otros servicios que comparten el mismo enlace de acceso a Internet.

Para las pruebas y simulaciones que se realizarán en esta tesis se seleccionará SopCast como servicio P2P-TV, por ser una de las aplicaciones más estudiadas [43]. Su funcionamiento, el análisis de su rendimiento y las mediciones de la QoE aportada son temas importantes para los investigadores, operadores y usuarios finales. SopCast funciona sobre *SoP technology*, un protocolo de comunicación propietario [47]. Sin embargo, es una aplicación que se puede utilizar gratuitamente para la transmisión de programas de TV con tasas que varían en el rango de $250\ kbps$ a $400\ kbps$, llegando a alcanzar en videos de alta calidad hasta $800\ kbps$. En [31] y [37] se puede encontrar una caracterización detallada de SopCast, enfocada especialmente en el tiempo entre paquetes, el número de *peer* con los que se intercambian datos, la duración de la comunicación, entre otros. En [32] y [40] se estudian los mecanismos básicos de dicha aplicación y se muestra que tanto el tráfico de señalización como el de video es transportado exclusivamente sobre el protocolo de transporte UDP.

El sistema de *streaming* desarrollado por SopCast se sustenta en una arquitectura de distribución no estructurada del tipo *mesh*, donde se posibilita que cada

peer descargue y distribuya los contenidos, llegando a existir múltiples proveedores y consumidores para los mismos contenidos. Este mecanismo es tolerante a fallos, pues los *peer* pueden incorporarse y abandonar la red P2P en cualquier momento y sin previo aviso, asegurando que los errores sean los menos posibles en la visualización del video. Se precisa un mecanismo de control del tráfico para establecer y mantener esta estructura de *peer*; de ahí que SopCast presente un tráfico de señalización, fundamentalmente compuesto por paquetes pequeños (menos de $100\ bytes$) [40]. Este control de flujo es necesario para gobernar el intercambio de los diferentes segmentos (*chunk*) en los que se divide el video. En la bibliografía se proponen tres niveles de clasificación para los *peer* de la aplicación: los *super peer* (suelen ser unos 5 del total) son los responsables de proveer cerca del $90\,\%$ del video a un *peer*; los *ordinary peer* envían algún que otro contenido y finalmente los *supplementary peer* que sólo intercambian paquetes de señalización [37].

Por otro lado, en [43] y [40] los autores presentaron una estimación del retardo tolerado por un cliente SopCast (entre 30 y 45 segundos, aproximadamente), detectándose la presencia de dos *buffer*: el primero en la aplicación, relacionado con el intervalo entre la selección de un canal y el inicio de la transmisión del video; y el segundo en el reproductor de video del cliente.

2.3 Limitaciones de las redes de acceso

Es bien sabido que Internet, en sus comienzos, fue diseñado para transferir una gran cantidad de datos de servicios tales como el correo electrónico, la navegación web o la transferencia de archivos (*File Transfer Protocol* (FTP)). Una de las principales características de estos servicios es la transmisión de tramas con el tamaño máximo permitido por diferentes tecnologías de comunicaciones (tamaño máximo de trama = *Maximum Transmission Unit* (MTU)), aumentando así la eficiencia y reduciendo el procesamiento en dispositivos de redes (*router* y *switch*) [48].

Por otro lado, en las redes de acceso, es común encontrarse con *router* de gama media y baja. Estos dispositivos utilizan el *buffer* como mecanismo de control de tráfico, por lo que el tamaño y el comportamiento del *buffer* se convierten en

parámetros de diseño importantes [9, 49]. Existen una gran cantidad de publicaciones relacionadas con el problema del tamaño del *buffer*, su comportamiento e influencia en diferentes tipos de tráfico multimedia, utilizando características de QoS basadas en parámetros de red objetivos (como *jitter*, pérdida de paquetes, entre otros) [50]. Sin embargo, el crecimiento exponencial de los servicios en tiempo real en Internet, genera altas tasas de paquetes pequeños. Por lo tanto, debido a que estos flujos generan gran cantidad de paquetes por segundo, incluso si mantienen la misma tasa medida en *bits* por segundo, la capacidad de procesamiento de los dispositivos de red podría convertirse en un cuello de botella [33].

En este sentido, se han publicado muchos estudios sobre la problemática del dimensionado del *buffer* [51]; aunque se ha abordado principalmente para los *router* del núcleo de la red y para flujos TCP. El problema de dimensionar el *buffer* fue estudiado en [52], donde se explica que hasta hace unos años se aceptaba la regla no escrita de usar el producto retardo-ancho de banda para dimensionarlo; sin embargo, esta regla está siendo sustituida por el llamado *"Stanford model"*, que sugiere la utilización de *buffer* más pequeños. Posteriormente en [53], se han propuesto *buffer* de menor tamaño (denominados *tiny buffer*) con una capacidad de varias decenas de paquetes.

En la literatura existente se pueden encontrar distintas propuestas respecto a la influencia de los *buffer* en diferentes servicios. En [54], [55] y [56] se muestran resultados en los que diferentes comportamientos, tamaños y políticas de los *buffer* modifican la calidad de los servicios. El conocimiento de los dispositivos de acceso a la red permite tomar decisiones en cuanto a qué técnicas de optimización del tráfico aplicar. Ejemplo de ello son las propuestas sobre servicios en tiempo real como VoIP o juegos *online* [57], cuyo tráfico de paquetes pequeños es optimizado mediante técnicas de multiplexión [58] para ganar en eficiencia y ahorrar ancho de banda pero teniendo en cuenta que el aumento del tamaño de los paquetes puede a su vez perjudicar a la calidad para ciertas políticas de *buffer*.

2.4 Técnicas de optimización de tráfico

Diversas técnicas pueden ser aplicadas para mejorar la utilización del enlace. En [59] se ha introducido y utilizado una técnica de control de la congestión para sistemas de videoconferencia codificados con *H.264*, con el fin de mejorar la QoE en escenarios con limitaciones de capacidad del enlace. Sin embargo, el tamaño y el comportamiento de los *buffer* en los *router* debe tenerse en cuenta para utilizar estas técnicas de forma adecuada. En [10] los autores presentaron un estudio de la influencia de un método de multiplexión en los parámetros que definen la calidad subjetiva de los juegos *online*. Los resultados mostraron que los *buffer* pequeños presentan mejores características para mantener el retardo y *jitter* en niveles adecuados, a costa de aumentar la pérdida de paquetes. Además, se puede obtener una reducción del $38\,\%$ en la utilización del enlace, agregando retardos debido a la retención de los paquetes.

Como se ha mencionado, el tráfico de paquetes pequeños generado por muchos servicios multimedia en Internet (P2P-TV, VoIP, videoconferencia y juegos *online*) pueden producir un *overhead* significativo en la red. Esto se debe a que sus requerimientos de tiempo real hacen que envíen una gran cantidad de paquetes por segundo, con lo que tienen poca eficiencia. Las técnicas de multiplexado y compresión se utilizan hace ya algún tiempo, e incluso se han estandarizado para escenarios donde varios tráficos de tiempo real comparten la misma ruta [58]. En [60] y [61], se emplea un método denominado *Tunneling, Compressing and Multiplexing* (TCM), con el que se logra ahorrar ancho de banda en el tráfico UDP de los juegos *online*. Se comprimen las cabeceras de los paquetes mediante algoritmos estándar; se multiplexa un número de paquetes en uno de mayor tamaño, y finalmente se realiza el envío extremo a extremo empleando un túnel *Layer* 2 *Tunneling Protocol* (L2TP). Los resultados muestran un ahorro de ancho de banda de hasta un $38\,\%$ para los juegos *online* que emplean *Internet Protocol* (IP)v4 y UDP, añadiendo un retardo a causa de la retención de los paquetes originales en el multiplexor.

Respecto a los algoritmos de compresión de cabeceras, en [62] se presentó un método para comprimir cabeceras IP y TCP. De igual manera, en [63] se descri-

bió *IP Header Compression* (IPHC), capaz de comprimir las cabeceras UDP e IPv6. El estándar *Robust Header Compression* (ROHC) se definió en [64] y puede comprimir las cabeceras IP, TCP, UDP y *Real-time Transport Protocol* (RTP). Este último presenta un mejor comportamiento en las redes inalámbricas, aunque conlleva más complejidad en su implementación [65], añadiendo más retardo de procesado que en el caso de IPHC. Todos estos métodos utilizan la redundancia de los campos de las cabeceras IP y UDP, para evitar el envío repetido de algunos campos. Igualmente, utilizan compresión *delta* para reducir el número de *bits* de los campos con un comportamiento incremental (por ejemplo, el número de secuencia). En los protocolos IP, UDP, TCP y RTP se define un *contexto*, con el fin de garantizar una correcta reconstrucción de los paquetes en el destino final; el *contexto* se transmite inicialmente con las primeras cabeceras y almacena los valores de los campos que no se envían, el mismo debe estar sincronizado entre el emisor y el receptor.

Considerando todo lo anteriormente expuesto, las técnicas de optimización presentadas en [60] pueden ser muy útiles si se aplican al tráfico P2P-TV, caracterizado por generar altas tasas de paquetes pequeños. Se podrían multiplexar los paquetes pertenecientes a un mismo flujo, por ejemplo los paquetes de acuse de recibo del nivel de aplicación. Incrementar la eficiencia de este tipo de tráfico podría ser beneficioso para las redes residenciales y de agregación, puesto que el ahorro de ancho de banda puede llevar a una mejor utilización de los recursos de la red [66].

Sin embargo, a pesar de las ventajas de las técnicas de multiplexión, no siempre su uso es apropiado, por ejemplo, cuando una aplicación genera paquetes grandes. En estos casos, no se pueden multiplexar en un paquete más grande y es conveniente usar otras técnicas de optimización del tráfico. Otra opción es hacer uso del alisado, o también conocido como *Traffic Shaping*, como mecanismo de conformado de tráfico. Esta técnica tiene como objetivo optimizar y garantizar el rendimiento de la red, mejorando así el uso de los recursos, a expensas de introducir pequeños retardos bajo ciertas circunstancias y un mínimo costo computacional.

Las técnicas de alisado juegan un papel esencial como una solución bastante robusta utilizada por los diferentes ISPs a la hora de controlar la congestión de

la red, el ancho de banda y la demanda. Como se explica en [67], algunos ISPs bloquean el tráfico de BitTorrent en sus redes, mientras que otros regulan o limitan el ancho de banda a aquellas aplicaciones que generan un tráfico invasivo, llegando a consumir la mayor cantidad del ancho de banda ofrecido por el proveedor.

Por lo general, los ISPs ofrecen un amplia gama de técnicas de conformado de tráfico, con el mero propósito de reducir la congestión en las redes y distribuir el ancho de banda ofrecido, de una manera justa, entre todos los usuarios [67]. Para ello, se intenta acortar al máximo el tiempo cuando aparecen picos en el rendimiento de la red. Estos mecanismos pueden aplicarse tanto al flujo agregado por un usuario, así como a un tipo de flujo en específico. Sin embargo, según [68], muchos proveedores hacen su mayor esfuerzo por no afectar a todos los tráficos, intentando así minimizar los efectos secundarios del alisado.

Por otro lado, en [67] y [68] se definieron tres tipos de políticas para el alisado:

- Conformación del tráfico para aplicaciones masivas: en este caso, se le asigna un ancho de banda *"fijo"* a cada flujo perteneciente a dichas aplicaciones.

- Modelo del tráfico agregado: en esta política, se le aplica el modelado de tráfico a todos los flujos ("tráfico agregado") proveniente de diferentes redes. Con el fin de evitar el congestionamiento de la red, a aquellos usuarios que consumen un gran ancho de banda durante una ventana de tiempo fijada por el ISP, se les retrasará el tráfico o se le asignará una menor prioridad.

- Modelo de tráfico programado en tiempo: en este sentido, el modelado de tráfico sólo se aplicará a determinadas horas picos, reduciendo los costos y la congestión. Además, este método puede aplicarse en paralelo con los dos anteriores.

Ahora bien, dar soporte y gestionar eficazmente las aplicaciones de video en tiempo real, se ha convertido en un gran reto para los ISPs, debido a las ráfagas de tráfico generadas. En este sentido, en [69] se propone *SAVE*, un algoritmo de alisado para video en redes adaptativas. Con el uso este algoritmo se alisa el video en su origen antes de ser enviado a la red, manteniéndose su calidad y asegurando

que el retardo en el *buffer* del cliente no supere cierto límite. Enfoques similares se han planteado en [70] y [71].

Como se ha hecho alusión anteriormente, los servicios *videostreaming* en tiempo real, como por ejemplo la transmisión de TV en vivo, videoconferencias, P2P-TV, entre otros, son tolerantes a ciertos niveles de retardos. En estos casos, las técnicas de alisado *"en línea"* (clasificadas como un método pasivo [72]) podrían reducir considerablemente la variabilidad de los recursos. En [73] se analizan algunos de estos algoritmos.

En la vida no hay cosas que temer, sólo hay cosas que comprender.

Marie Curie

CAPÍTULO

3

Arquitectura y Metodología

El crecimiento exponencial de los servicios de video móvil en dispositivos inteligentes (*Youtube*, *Netflix*, TV móvil, entre otros), en conjunto con los avances de las tecnologías y aplicaciones para IoT (como por ejemplo, el desarrollo y montaje de ciudades inteligentes con la presencia de miles de dispositivos interconectados y gestionados, la telemedicina, los nuevos sistemas de educación, aplicaciones de realidad virtual, videojuegos y videovigilancia), han desencadenado iniciativas a nivel mundial para el desarrollo de sistemas de comunicaciones móviles e inalámbricos $5G$ en conjunto con tecnologías de softwarización de redes tales como NFV, SDN y *Multi-access Edge Computing* (MEC). Dichos servicios y muchos otros más, demandan una plataforma con capacidad de procesamiento suficiente que garantice la conectividad *"en cualquier lugar y en cualquier momento"*, con velocidades más altas, menor latencia extremo a extremo, mayor ahorro de energía, así como también mayor fiabilidad y disponibilidad comparada con plataformas ya existentes.

Las arquitecturas de redes $5G$ deben integrar tanto los nuevos elementos que posibilitan la tecnología $5G$ móvil al igual que los sistemas heredados, tales como 4^{th} *Generation Mobile Network* ($4G$), *Wireless Fidelity* (Wi-Fi) y redes de acceso fijos. Esto trae consigo la convergencia de tráfico proveniente de múltiples aplicaciones, lo que podría ocasionar la congestión en las redes de acceso y por ende, la degradación de la QoS para el usuario final.

En este capítulo, primeramente, se propone una plataforma de gestión y orquestación para una red $5G$ extremo a extremo, en la cual se integran diferentes

subsistemas ($5G$ móvil, $4G$ e implementaciones experimentales). En este sentido, se detalla la implementación de dicha arquitectura, desde la virtualización del núcleo de la red, la *Radio Access Network* (RAN), la integración de los diversos tráficos provenientes de cada subsistema, así como la plataforma de orquestación puesta en marcha haciendo uso de *OpenStack* y *Open Source Mano* (OSM).

Seguidamente, se plantea la arquitectura de una prueba de concepto considerando un sistema inteligente de videovigilancia en sistemas de transporte público, utilizando dispositivos IoVT localizados en el borde de la nube, con base en el concepto conocido como *Edge Cloud*. La inteligencia del sistema de videovigilancia radica en la gestión y orquestación de las distintas funciones desplegadas, considerándose cada una de ellas como distintas VNFs. Como punto de partida, se utiliza la arquitectura $5G$ detallada previamente y se explica cómo es la interacción entre las diferentes capas de la arquitectura propuesta para este caso de uso.

También, se describe una metodología para la optimización y el análisis del tráfico de aplicaciones, la cual tiene como objetivo minimizar la congestión en los *buffer* de los *router* de las redes de acceso, evitando los cuellos de botella, reduciéndose la pérdida de paquetes y por ende, mejorando la QoS de los servicios. Finalmente, se describen las dos técnicas de conformado de tráfico que se utilizan en este trabajo: *(1)* un método para la multiplexión y compresión de cabeceras y *(2)* un método de alisado de tráfico.

3.1 Arquitectura 5G extremo a extremo

A día de hoy, la dinámica y flexibilidad necesarias para poner en marcha diferentes servicios en redes $5G$ dictan un enfoque de orquestación automatizado, integrado y coordinado, más allá de intentar modificar y reestructurar las redes tradicionales condicionadas por una arquitectura de red estática, donde se precisaría reemplazar y establecer procedimientos de gestión y control para cada nodo existente. Asimismo, las redes $5G$ requieren de un conjunto de tareas adicionales indispensables para la gestión de recursos, que garantice que los mismos sean reconfigurables y distribuidos, además de permitir la implementación de VNFs. Por lo tanto, el diseño de una arquitectura apropiada de gestión y orquestación es

esencial para asegurar una utilización eficiente, conjunta y coordinada de un grupo de recursos en la infraestructura móvil, con el propósito de desplegar servicios de red satisfaciendo sus requerimientos y alcanzando el mayor beneficio posible de los recursos disponibles. Con este fin, se ha de tener en cuenta los siguientes elementos: la gestión adecuada de los modelos de abstracción de los recursos, la interacción con los algoritmos de ubicación de los VNFs, la aplicación del concepto de *network slicing* para segmentar y proveer recursos aislados posibilitando la creación de diferentes instancias de servicios de red y la disponibilidad de interfaces abiertas y estandarizadas.

La Figura 3.1 muestra el esquema de alto nivel correspondiente a la plataforma de orquestación de servicios de red propuesta y validada en el presente trabajo. El sistema heterogéneo E2E a implementarse estará compuesto por diferentes elementos habilitados para $5G$, al mismo tiempo que integrará sistemas de arquitecturas heredadas o también conocidos como sistemas *legacy*. Todas las funciones de red se virtualizarán y ejecutarán en máquinas virtuales en diferentes infraestructuras de nube, las cuales serán gestionadas y orquestadas a través de un mismo sistema de orquestación. Los diferentes elementos que se integran en esta arquitectura son una combinación de: *(a)* prototipos de equipamiento pre-comercial para $5G$ móvil, *(b)* equipamiento comercial para $4G$ e *(c)* implementaciones experimentales. Esta integración de diferentes sistemas posibilita la incorporación de las contribuciones más actuales en la convergencia de las redes, así como también de las soluciones basadas en tecnologías de nube para las arquitecturas móviles, incluyendo la RAN y la separación o segmentación de funciones en el núcleo de la red.

Por un lado, el sistema $5G$ móvil consiste en un conjunto de antenas al aire libre, transmitiendo en las bandas pioneras para $5G$ (3,5 GHz y 28 GHz), garantizando un mejor rendimiento y menores latencias en las redes de acceso. Las funciones virtualizadas de la RAN y del núcleo de la red se ejecutan en diferentes entornos de nube, que están interconectados con la plataforma de gestión y orquestación.

Por otro lado, el sistema comercial $4G$ utilizado está compuesto por un grupo de celdas pequeñas ubicadas en un espacio interior transmitiendo a 700 MHz; las

mismas soportan tecnologías *Narrow-Band-IoT* (NB-IoT), como ha sido definido por 3^{rd} *Generation Partnership Project* ($3GPP$) en [74]. El núcleo de la red $4G$ se ha virtualizado y permite el *network slicing* de un extremo al otro, de esta manera se pueden tener múltiples VNFs que satisfagan diferentes niveles de QoS y la separación de servicios. Al igual que el sistema $5G$, el núcleo virtualizado de $4G$ está interconectado con el orquestador.

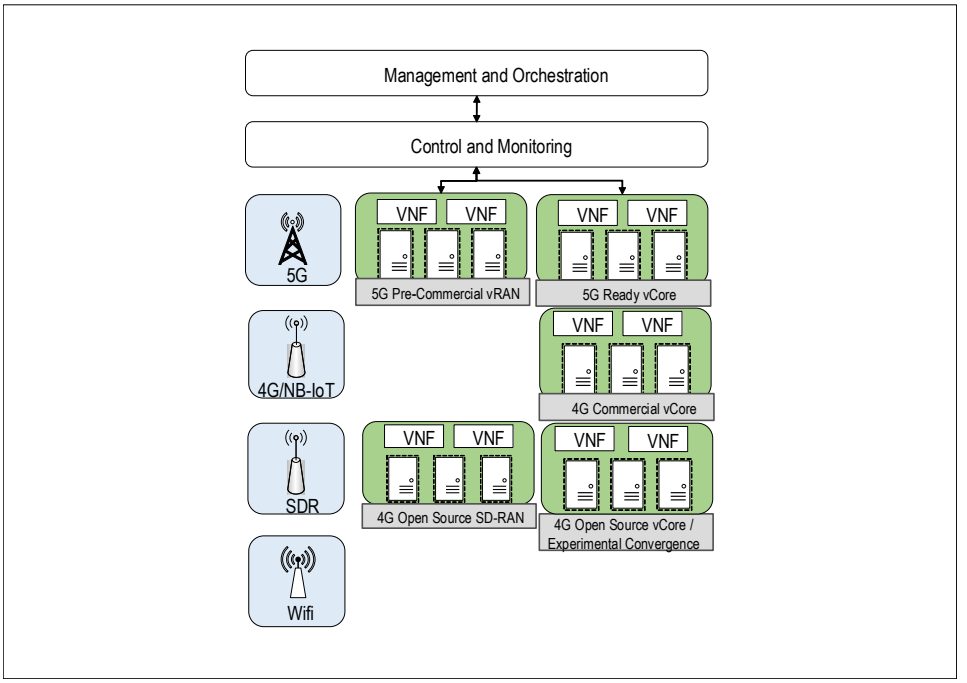

Figura 3.1: Arquitectura de alto nivel de la red 5G propuesta.

Para las implementaciones experimentales, se han integrado algunos elementos específicos de servicios de red, que son el resultado de múltiples trabajos de investigación en el contexto de IoT, gestión de acceso y la ubicación de funciones en el núcleo de la red.

3.1.1 Integración de *cloud-RAN*

La virtualización en las redes de acceso ha ido ganando mucho interés tanto en la comunidad científica como en los organismos de estandarización [75, 76]. En este sentido, no es necesario que toda la pila de protocolos de acceso se encuentre ubicada en la parte de transmisión, lo cual permite considerar implementaciones más flexibles de la red de acceso. Resulta particularmente interesante la separación entre las funciones de la capa de control de acceso al medio respecto a las funciones de la capa física, permitiendo técnicas más cooperativas en la gestión de recursos de radio. Muchos estudios han intentado encontrar una solución óptima para la separación de las funciones en la RAN, lo que maximizaría su capacidad al permitir la cooperación entre diferentes transmisores al mismo tiempo que reduciría la sobrecarga en la red de transmisión frontal (*fronthaul*). Sin embargo, no existe una regla general en este sentido y existen muchas variables que deben ser consideradas a la hora de evaluar un despliegue óptimo [75]. En el contexto de virtualización y redes orquestadas, la selección de la mejor implementación depende de las particularidades de cada caso de uso y es posible alcanzar un punto óptimo a través de la instanciación y combinación de múltiples funciones de red (VNFs).

La implementación de la *Cloud Radio Access Network* (C-RAN) permite la separación de diferentes funcionalidades en la pila de protocolos de RAN, es decir, que diferentes servicios de redes pueden ser implementados e instanciados basándose en los requerimientos de cada caso de uso. Por ejemplo, en las aplicaciones IoT con grandes requerimientos de ancho de banda, las prestaciones de la C-RAN son fundamentales para garantizar una relación de compromiso viable entre el rendimiento y el retardo.

3.1.2 Virtualización del núcleo de red

De la misma manera que sucede con la RAN, la virtualización de las funciones del núcleo de la red permitirá implementaciones más flexibles que pueden abordar algunos de los *Key Performance Indicator* (KPI)s requeridos por $5G$, facilitando el camino hacia un arquitectura orientada a servicios. La modularidad, reusabilidad

y autosuficiencia de las funciones de red constituyen aspectos de diseño adicionales para una arquitectura de red $5G$, descrita conforme a las especificaciones del $3GPP$ [77], donde las funciones de control y del plano de usuario están completamente desacopladas y se comunican entre sí a través de nuevas interfaces. Esta modularidad y autosuficiencia de las funciones de red trae consigo nuevas e innovadoras opciones de implementación, lo que facilitaría las capacidades computacionales en ubicaciones cercanas al usuario final en el borde de la red [78].

Cuando las funciones del plano de control y de usuario se encuentran separadas, el plano de usuario (que presenta limitaciones de tiempo mucho más estrictas que el plano de control) se podría ubicar en el borde de la red, como una solución más sencilla para suministrar servicios y contenidos. Tal implementación permite la descentralización de servicios y la distribución del contenido en caché a través de la red, lo que mejoraría el problema de la latencia y el congestionamiento en la red de transporte.

3.1.3 Integración de redes heterogéneas

Las arquitecturas de redes $5G$ deben proporcionar un marco integrador para diferentes tipos de redes de acceso: desde las redes de acceso celular (redes $3GPP$), las redes de acceso por radio (Wi-Fi o redes específicas para tecnologías IoT), hasta las redes de acceso fijo. Para sacar provecho de la cantidad de recursos disponibles y maximizar el uso de todas las redes disponibles, la arquitectura $5G$ está destinada a garantizar el acceso simultáneo a través de diferentes tecnologías. Esto se conoce como convergencia de redes, y tanto el $3GPP$ como el *Broadband Forum* han invertido muchos esfuerzos para habilitar arquitecturas que así lo permitan [79, 80].

En la implementación presentada en este trabajo y con el fin de proporcionar una arquitectura que permita la convergencia de redes, se incluirá la funcionalidad de un *Hybrid Access Gateway* (HAG). El HAG actuará como un *proxy* y posibilitará que el tráfico fluya hacia el usuario a través de múltiples redes heterogéneas. Para ello, se desarrollará un VNF específico para la función *Multi-Path TCP* (MPTCP), que favorecerá el despliegue de una red $5G$ [81]. La función de

MPTCP agregará tráfico en el HAG y permitirá la selección de rutas en función de un número determinado de parámetros que pueden utilizarse para optimizar la asignación de recursos. En este caso, en el equipamiento del usuario se podría cambiar entre las diferentes interfaces de red que envían y reciben tráfico del HAG. Además, el HAG incluirá la gestión de tráfico y las políticas adecuadas que deben ser aplicadas, complementándose con las partes de control y monitorización proporcionadas por la orquestación para garantizar un comunicación satisfactoria de un extremo a otro.

3.1.4 Orquestación de servicios en la nube

Desde la aparición de las arquitecturas NFV, las funciones de red incluidas en los sistemas de comunicaciones son una combinación de elementos físicos y de *software* que se ejecutan en infraestructuras de nube, trayendo consigo que estas tecnologías sean una herramienta crítica para permitir un despliegue y una gestión dinámica de los VNFs. Uno de los principales requisitos en una infraestructura de nube es que la misma debe adaptarse a diferentes tecnologías, tales como sistemas ya existentes ($4G$, Wi-Fi, entre otros), así como también sistemas heterogéneos (como por ejemplo los variados ecosistemas para IoT). Una de las funciones fundamentales de la arquitectura $5G$ propuesta en este trabajo es proporcionar mecanismos para la gestión y orquestación de múltiples sistemas y garantizar la interoperabilidad entre ellos. Por este motivo, se han considerado las siguientes premisas de diseño para la solución de orquestación de servicios en la nube presentada:

- Un sistema operativo en la nube que controla los recursos informáticos, de almacenamiento y de red a través de un centro de datos.
- Un orquestador que proporciona el marco de trabajo necesario para lograr la integración entre sistemas heterogéneos.
- Un *Virtual Infrastructure Manager* (VIM) que gestiona varios VNFs, utilizando máquinas virtuales y contenedores.

- Una solución SDN que proporciona la capacidad de programar dinámicamente los flujos de tráfico con el fin de modificar las *Virtual Local Area Networks* (VLAN) según reglas de tráfico pre-definidas.

Por lo general, cada VIM contiene su propio conjunto de herramientas de gestión y ofrece una interfaz de usuario con el objetivo de controlar los servicios de cada VIM. Sin embargo, al ser dicha gestión interna para cada VIM, podría no ser funcional en entornos con múltiples VIMs cuando diferentes proveedores coexisten en la misma plataforma. En los últimos años han surgido múltiples soluciones de orquestación: *OpenBaton*, *Cloudify* y OSM de *European Telecommunications Standards Institute* (ETSI), entre otros.

En la propuesta presentada en este trabajo, se utilizará OSM como componente de *Management and Orchestration* (MANO), lo cual permitirá la orquestación, sincronización y gestión de VNFs y/o servicios de red. OSM funcionará como orquestador extremo a extremo debido a su capacidad de gestión de todos los diferentes subsistemas, la creación de servicios de red y la provisión de un marco de comunicación integrado con varios centros de datos. Al mismo tiempo, OSM ofrece un conjunto de *plugin* lo cual posibilita la utilización de diferentes soluciones de *software*, la inclusión de orquestación de recursos internos y el uso de VIMs. Estos *plugin* se pueden clasificar, en función de los componentes de OSM con los que se integran: el *Resource Orchestrator* (RO), la *User Interface* (UI) y el *Service Orchestrator* (SO). El RO es responsable de crear y ubicar los recursos de cómputo y de red, para interactuar con un controlador SDN; también permite la gestión de nuevas VIMs y la coordinación de recursos entre múltiples VIMs. La UI proporciona la interfaz del usuario en el orquestador OSM; y por último, el SO es el responsable de la orquestación de servicios de un extremo a otro.

Por otro lado, la arquitectura VNF incluirá el componente VIM, el cual controlará la infraestructura NFV, es decir, todos los componentes de *hardware* y *software* que constituyen el entorno de implementación de los VNFs. La comunidad de telecomunicaciones ha reconocido el potencial de *OpenStack* y ha establecido el mismo como una plataforma viable para NFV [82]. Por este motivo, en la plataforma presentada se elegirá *OpenStack* con el objetivo de proporcionar las bases

de la arquitectura NFV, ya que ofrece *Application Programming Interfaces* (APIs) estándar entre elementos de NFV, como son la infraestructura y las interfaces de usuario. Se utilizará *OpenFlow* como solución SDN con el fin de mantener operaciones dinámicas del flujo de tráfico. El controlador *OpenDaylight*, ampliamente utilizado en los sistemas de telecomunicaciones, permite segmentar y aislar diferentes redes de transporte para satisfacer diferentes niveles de QoS, mediante la diferenciación de flujos de tráfico de diversas índoles, haciendo posible cumplir con las reglas de reenvío de tráfico en los *switch* SDN.

En la Figura 3.2 se presenta la arquitectura de orquestación propuesta, a través de la cual se podrán administrar múltiples centros de datos simultáneamente. Como ya se ha mencionado, OSM se emplea para orquestar VNFs en diferentes VIMs (en este caso en particular, *OpenStack* y el *Cloud Execution Environment* (CEE) de un proveedor de telecomunicaciones). Debido al carácter propietario de la VIM del proveedor, será necesario implementar adicionalmente un *plugin* que permita la comunicación entre las diferentes tecnologías empleadas en la arquitectura propuesta.

De igual manera, se utilizará *OpenDaylight* para proporcionar dos funcionalidades diferentes: el control de flujo para permitir que OSM pueda administrar redes inteligentes y la monitorización en tiempo real de la infraestructura y los recursos de la red. Además, se desarrollará un *plugin* para la UI que posibilita la monitorización en tiempo real del estado de la red (por ejemplo latencia, *jitter*, rendimiento, entre otros).

3.2 Arquitectura para un sistema de videovigilancia

A continuación, se describe la arquitectura de servicios de red propuesta para implementar un sistema VSS basado en dispositivos IoVT. El modelo aborda la necesidad de proveer videovigilancia inteligente en vehículos de transporte público de una manera eficiente en términos de congestión de red. El objetivo de esta prueba de concepto es proporcionar gestión dinámica de aplicaciones basadas en la nube, donde el procesamiento de datos se realiza en un conjunto de nodos de borde ubicados en una flota de vehículos de transporte público.

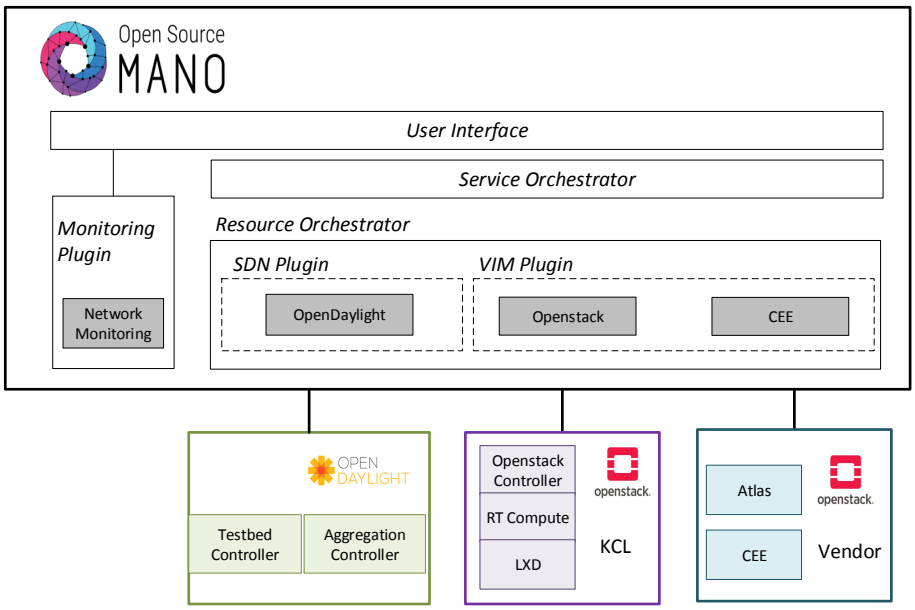

Figura 3.2: Propuesta de orquestación de la arquitectura de virtualización utilizada.

La Figura 3.3 muestra la arquitectura general propuesta, en la cual, la capa *Nodos Edge* tiene la función de extender la capacidad de procesamiento de la nube al proveer cierta potencia de cómputo en los dispositivos IoT. Al ubicar dichos nodos cerca de la fuente de datos, se garantiza una reducción en la latencia, lo que permite una respuesta más rápida de la aplicación y una mejor experiencia del usuario final. Esta capa consta de un conjunto distribuido de dispositivos inteligentes IoT equipados con una cámara integrada. Cada nodo será responsable de establecer y mantener una comunicación segura desde el dispositivo a la capa *Servicios en la Nube*, así como de la ejecución local de aplicaciones *serverless* y contenedores, además de gestionar mensajes, datos y seguridad. La capa *Red de Acceso* proporciona conectividad (por ejemplo, $5G$, $4G$ o Wi-Fi) para cada nodo.

La capa *Servicios en la Nube* garantiza el acceso a recursos ilimitados para tareas de cómputo más intensivas y una rica fuente de servicios adicionales gestionados en la nube. Dicha capa puede verse como un entorno de múltiples nubes, combinando nubes públicas y privadas, favoreciendo una mayor redundancia y

brindando la capacidad de encontrar el servicio óptimo en la nube para una necesidad comercial o técnica particular. También ofrece escalabilidad, ya que la capacidad y los recursos se pueden aumentar o disminuir según la demanda. Dicha elasticidad brinda la posibilidad de ajustar los recursos en función de la demanda real. Esta característica es especialmente importante para empresas cuyas demandas son altamente estacionales y enfrentan muchos picos de demanda, como suele pasar con los servicios de transporte público.

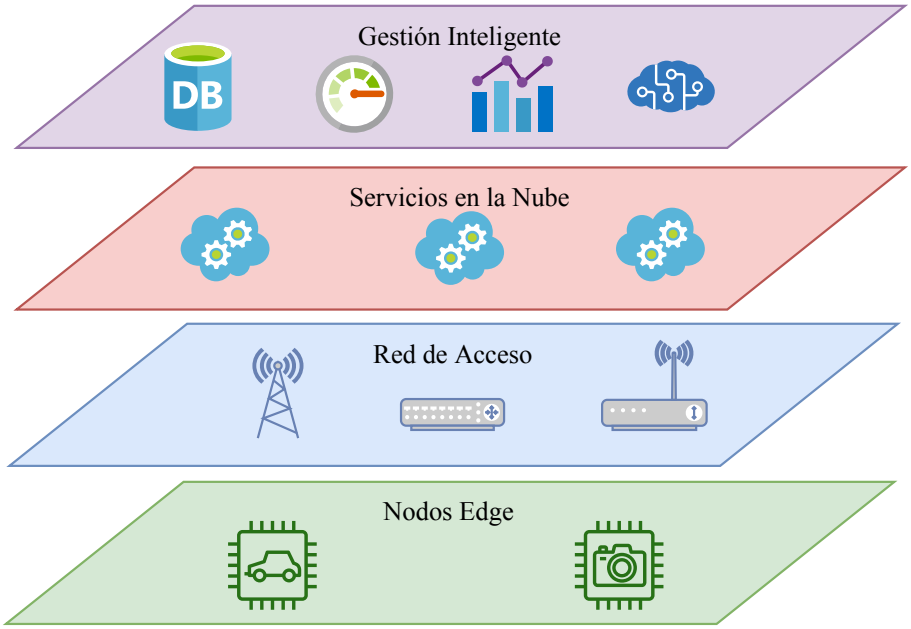

Figura 3.3: Arquitectura general para un VSS en vehículos de transporte público.

Siguiendo el paradigma NFV de ETSI, la capa *Gestión Inteligente* permite la coordinación de múltiples recursos en diferentes VIMs. Esta capa es responsable de la creación, ubicación y gestión del ciclo de vida de aplicaciones y servicios de red; también es capaz de interactuar con un controlador SDN, proporcionando una integración entre la nube e infraestructuras SDN. Por otro lado, organiza los despliegues de servicios en *Nodos Edge* utilizando la VIM correspondiente de la capa *Servicios en la Nube*. Los despliegues de servicios se pueden reorganizar en

función de los datos que proveen las capas *Servicios en la Nube* y *Nodos Edge*.

Para la gestión de aplicaciones, la capa *Gestión Inteligente* recibe métricas de QoS (pérdida de paquetes, *throughput* y retardo) desde la capa *Nodos Edge*. Con esta información, se pueden instanciar las diferentes cadenas de microservicios de una manera eficiente en términos de utilización de ancho de banda de la *Red de Acceso*.

3.3 Metodología de optimización del tráfico

Ahora bien, en plataformas de telecomunicaciones heterogéneas, con múltiples implementaciones de distintas redes de acceso, es obvio que se generen grandes volúmenes de información concurrentemente. Teniendo en cuenta la cantidad de datos gestionados y manipulados por estos sistemas, es de esperar que la concatenación de diferentes flujos de tráfico en dispositivos intermedios o en los propios dispositivos de borde, sobrecarguen la red y generen picos de *throughput* muy altos. En estos casos, el *buffer* de los dispositivos de red puede saturarse dando como resultado la ocurrencia de períodos de congestión, provocando una pérdida significativa de paquetes, afectando negativamente la QoS para los usuarios. En este contexto, los enlaces de acceso pueden convertirse en puntos sensibles y producir cuellos de botella en la red debido a su capacidad limitada.

La metodología de optimización del tráfico que se presenta permite minimizar la congestión en los *buffer* de los dispositivos de acceso. Se propone la utilización de una técnica de conformado de tráfico que previene que dichos dispositivos se conviertan en cuellos de botella, reduciéndose la pérdida de paquetes y aumentando la QoS.

La transmisión de datos de algunas aplicaciones y la congestión de dispositivos de red en ciertos puntos sensibles (por ejemplo, enlaces de acceso domésticos, dispositivos de red de gama media y baja con *buffer* limitados y manejo de grandes cantidades de paquetes) pueden causar muchos problemas en la información que se transmite por las redes. Para reducir la pérdida de paquetes causada por la congestión, se pueden considerar dos enfoques: modificar el código de la aplicación o considerar soluciones externas a la aplicación.

Con el primer enfoque, se asegura un mecanismo automático de control de la congestión para que el tráfico pueda adaptarse a las nuevas condiciones externas e implementar un mecanismo de QoS adecuado (como lo hace *Skype* [83]), sin embargo esto requiere que se tenga acceso al código fuente de la aplicación o servicio. En el segundo enfoque, se utiliza un dispositivo (o *software*) intermedio entre la aplicación y el *buffer* del *router*, que permita el procesamiento o tratamiento externo del tráfico generado [10, 84], con el objetivo de adaptarlo a las situaciones críticas antes mencionadas. En aras de presentar una solución que no dependa del tipo de aplicación, en este trabajo se adopta el segundo enfoque ya que no siempre es posible tener acceso al código fuente de la misma.

La metodología propuesta se divide en dos grandes bloques, según se puede observar en la Figura 3.4. En el primer bloque, denominado *Análisis de Tráfico*, se estudian las características del tráfico, con el propósito de definir y seleccionar los parámetros de QoS que influyen en su comportamiento. Lo primero que se requiere es implementar un mecanismo de monitorización que permita capturar el tráfico de la aplicación. Luego, se realiza un análisis de los parámetros objetivos de QoS, como por ejemplo: el tamaño de los paquetes, el tiempo entre paquetes y el *throughput*.

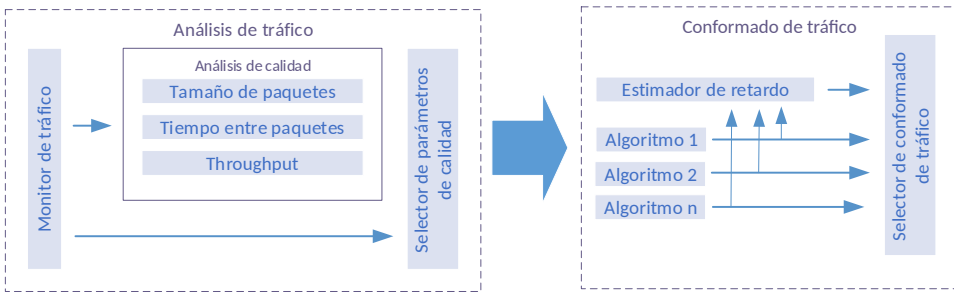

Figura 3.4: Metodología Propuesta.

Mediante el *Selector de parámetros de calidad*, se definen los parámetros más restrictivos en la utilización del enlace de acceso por dicho tráfico (tamaño de paquetes, tiempo entre paquetes y *throughput*). En el caso del tamaño de los paquetes, la idea principal es determinar si el tráfico transmitido consiste de muchos

paquetes grandes y/o pequeños, siendo esto un factor a tener en cuenta al seleccionar la técnica de conformado de tráfico más conveniente. Otro de los parámetros útiles en el estudio del tráfico de una aplicación de tiempo real es el tiempo entre paquetes, ya que ayuda a identificar cómo es el comportamiento de la aplicación: (a) si presenta ráfagas de paquetes o (b) si los paquetes presentan un patrón de envío constante. El *throughput* se utiliza para poder determinar un umbral mínimo garantizable para la aplicación, de tal manera que tenga un funcionamiento adecuado.

En el bloque de *Conformado de Tráfico* se selecciona la técnica de conformado de tráfico más adecuada y se entrega como salida el tráfico optimizado. El objetivo de este bloque es minimizar el impacto causado por el tráfico a ráfagas en el *buffer* de los dispositivos de acceso. De esta manera se evita la pérdida de paquetes en los *buffer* durante los períodos de congestión, introduciéndose ciertos niveles de retardo que deben ser tolerables por las aplicaciones. Este bloque puede contener diferentes algoritmos o técnicas de conformado de tráfico y se selecciona el que ofrezca la mejor relación de compromiso en cuanto al retardo soportado por la aplicación y la utilización del enlace. Con el uso de estas técnicas se pretende limitar los picos altos de *throughput* que podrían causar congestionamiento.

El *Estimador de retardo* permite determinar el retardo añadido a cada paquete teniendo en cuenta las restricciones de tiempo real del tráfico, esto es especialmente importante para poder comparar con los niveles tolerables de retardo de cada aplicación. Por último, el *Selector de conformado de tráfico* permite determinar la técnica de conformado de tráfico que más se ajuste al comportamiento de la aplicación. Estos valores de retardo se deben seleccionar para controlar la pérdida de paquetes causada por la llegada del tráfico a ráfagas y para mejorar la calidad de la comunicación. Por lo tanto, se debe valorar una relación de compromiso entre los retardos añadidos y los niveles de pérdida de paquetes, según las características de la aplicación y sus requisitos de retardo.

Con el propósito de validar esta metodología, se ha seleccionado una aplicación de *video streaming* en tiempo real para analizar su comportamiento y determinar sus principales parámetros y características con el objetivo de garantizar un mejor rendimiento en la transmisión y reducir la pérdida de paquetes en el enlace

de acceso. Estas mejoras consisten en adaptar la tasa del tráfico, teniendo en cuenta las restricciones de los dispositivos de acceso.

3.4 Técnicas de conformado de tráfico

En esta sección se presentan dos técnicas de optimización de tráfico para servicios en tiempo real que podrían ser aplicadas cuando varios flujos comparten la misma ruta de acceso a Internet, con el fin de reducir la pérdida de paquetes a costa de introducir ciertos niveles de retardos, los cuales deben ser tolerados por dichos servicios. Para mostrar la viabilidad del uso de esta metodología, estas técnicas se han aplicado al tráfico de una de las aplicaciones P2P-TV en tiempo real: SopCast (ver capítulo 4). Esta aplicación genera un tráfico a ráfagas con un gran volumen de paquetes grandes y pequeños, siendo este un tráfico representativo para el estudio mostrado.

3.4.1 Multiplexión y compresión

En las redes IP, cada unidad de información está compuesta por una cabecera IP, en la cual se especifica entre otros campos: el origen, destino, tamaño y número de secuencia. Muchos de los campos de estas cabeceras, son idénticos en todos los paquetes pertenecientes a un mismo flujo (con un mismo origen y un mismo destino), o varían muy poco entre ellos. Esto ha conllevado a buscar métodos para reducir el *overhead* mediante la supresión de algunos de estos campos, como por ejemplo la compresión de cabeceras. Esta técnica tiene como inconveniente de que sólo se puede utilizar de nodo a nodo, o bien mediante un túnel extremo a extremo. Otra técnica ampliamente empleada para la mejora del tráfico es la multiplexión, la cual consiste en unir el contenido de varios paquetes que comparten la misma cabecera. Sin embargo, si se multiplexa el contenido de varios paquetes en uno más grande, se introducen ciertos retardos que pueden ser intolerables por las aplicaciones en tiempo real. Mediante el uso de la compresión de cabeceras y la multiplexión de varios paquetes en uno más grande, se logra una mejoría en la eficiencia de las redes cuando aparecen ráfagas de paquetes con redundancia en sus cabeceras.

En primer lugar, se necesitará un protocolo que pueda comprimir cabeceras IP/UDP. En este caso se seleccionará IPHC, por ser suficiente para los propósitos de este trabajo y por presentar una implementación más sencilla que la de ROHC. IPHC es capaz de comprimir las cabeceras UDP a 2 $bytes$, empleando solo 8 $bits$ para el *Context Identifier* (CID) y evitando el campo de control opcional (*checksum*). La cabecera IPv4 puede comprimirse también a 2 $bytes$, por lo que se considerará una media de 4 $bytes$ para todas las cabeceras comprimidas, excepto para las cabeceras completas que serán de 28 $bytes$ y que, de acuerdo con la especificación de IPHC se envían cada 5 segundos [62].

Para ilustrar el *overhead* que puede generar el tráfico original de SopCast y el ahorro de ancho de banda que es posible obtener gracias a la compresión de cabeceras y la multiplexión de los paquetes, la Figura 3.5 muestra la reducción del tráfico alcanzado cuando cuatro paquetes de esta aplicación P2P-TV se multiplexan en uno más grande.

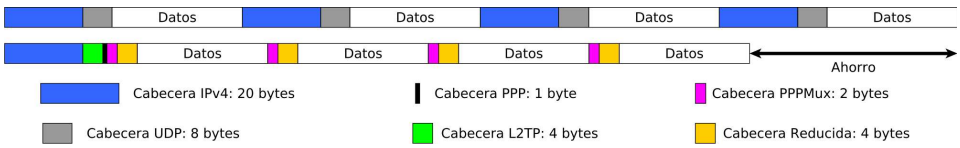

Figura 3.5: Tráfico original y multiplexado de SopCast.

Se utilizarán dos políticas diferentes para seleccionar cómo los paquetes serán multiplexados y están basadas en el uso de: *(1)* un *período* fijo o *(2)* un *umbral* de tiempo entre paquetes, como se muestra en las Figura 3.6 y Figura 3.7 respectivamente. Los paquetes generados por la aplicación SopCast se denominarán *nativos*, para diferenciarlos de los multiplexados.

Ambas políticas tratan de mantener los valores del retardo añadido por debajo de una cota superior, para evitar afectar la QoE de los usuarios de SopCast. Algunos servicios multimedia, como VoIP o los juegos *online* presentan restricciones muy rigurosas para el retardo añadido. Sin embargo, en el caso de P2P-TV este problema es menos severo, pues los contenidos descargados se almacenan en el *buffer* de la aplicación antes de reproducirse. En SopCast, por ejemplo, el *buffer*

puede almacenar aproximadamente hasta 60 segundos de video [43], por lo que la mayor limitación en el número de paquetes a multiplexar, en este caso, vendrá dada por la MTU de la red.

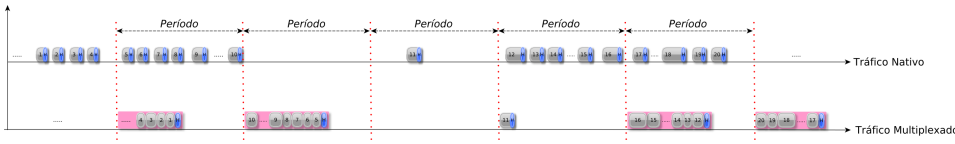

Figura 3.6: Política de multiplexión basada en un *período*.

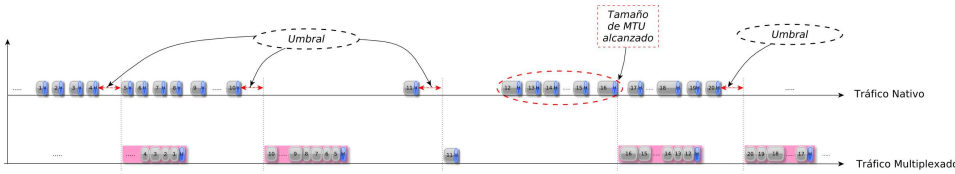

Figura 3.7: Política de multiplexión basada en un *umbral*.

Para la multiplexión basada en un *período*, se define un lapso de tiempo fijo de forma que se envía un paquete multiplexado, incluyendo todos los paquetes que han llegado hasta ese momento (Figura 3.6), al final de cada intervalo de tiempo. Existen tres excepciones: si no ha llegado ningún paquete, no se envía nada a la red; si sólo llega un paquete, se envía en su forma original; finalmente, si se alcanza el tamaño de la MTU, se envía el paquete multiplexado y se reinicia un nuevo *período*. Si el valor del *período* aumenta, el ahorro de ancho de banda mejorará, pues los paquetes multiplexados serán más grandes y el *overhead* total disminuirá. Los valores seleccionados para el *período* no pueden incrementarse indefinidamente, ya que se perdería el contacto con los *peer* proveedores del video.

Sin embargo, para la multiplexión basada en un *umbral* se ha tenido en cuenta cómo los paquetes entre dos *peer* de SopCast son enviados a la red, siguiendo un patrón a ráfagas y observándose grandes grupos de paquetes consecutivos cada ciertos intervalos de tiempo [31]. En la Figura 3.8 se aprecia que los *peer* reciben la información de video concentrada en bloques, es decir, en un intervalo

determinado de tiempo se recibe una cantidad notable de paquetes de video consecutivamente; posteriormente sólo se reciben paquetes de señalización hasta que comienza nuevamente la transmisión de información. De ahí que se puede concluir que el tráfico analizado sugiere un patrón a ráfagas.

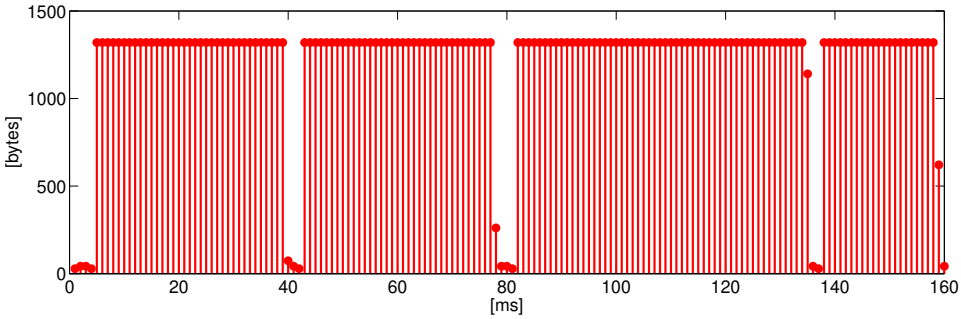

Figura 3.8: Tráfico recibido por un *peer* de SopCast, sugiriendo un patrón de tráfico a ráfagas.

En la Figura 3.7 se muestra la política de multiplexión basada en *umbrales*, capaz de adaptarse al comportamiento del tráfico antes mencionado. Se define el diagrama de estado, mostrado en la Figura 3.9, que comienza en un estado de *Espera* hasta que llega el primer paquete de la ráfaga. Una vez que se recibe un paquete, el sistema pasa al estado de *Almacenamiento* (*transición A*), en el cual se irán acumulando los paquetes a medida que van llegando y que presentan un tiempo entre paquetes menor o igual al *umbral* seleccionado (*transición B*). Sin embargo, si el tiempo entre dos paquetes supera el valor de dicho *umbral* o si se alcanza el valor de la MTU (*transición C*), el sistema considera que la ráfaga concluye, entonces se multiplexan los paquetes que han llegado hasta ese momento y se envían; cuando esto sucede, el sistema retorna al estado de *Espera*.

Para lograr que esta política se adapte al tráfico, se debe seleccionar un valor adecuado del *umbral* (tiempo entre paquetes), que permita discernir claramente cuáles paquetes forman parte de cada ráfaga.

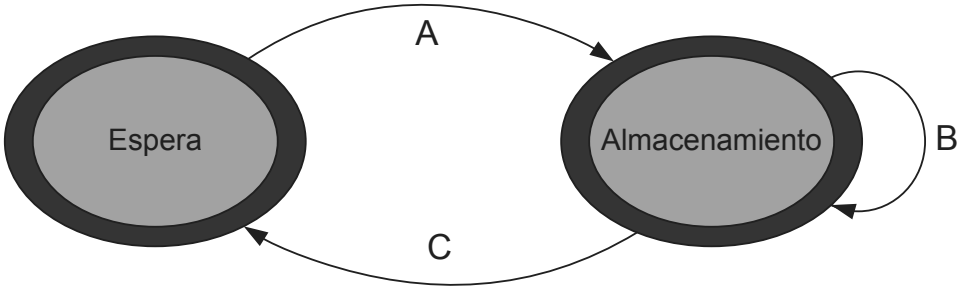

Figura 3.9: Diagrama de estado empleado en la política de multiplexión basada en un *umbral*.

3.4.2 **Alisado de tráfico**

Otro de los mecanismos de conformado ampliamente utilizado para minimizar el impacto causado por el tráfico a ráfagas en las redes es el alisado, también conocido como *Traffic Shaping*, como se ha explicado en el capítulo 2. La función principal de este mecanismo de control de tráfico es el de optimizar y garantizar el rendimiento de la red, mejorando el uso de los recursos, a expensas de introducir pequeños retardos bajo ciertas circunstancias y un mínimo coste computacional. Con el alisado, se limitan los picos de *throughput* que ocasionalmente aparecen cuando se envía tráfico a la red. De esta manera, se minimiza el impacto causado por las ráfagas de paquetes en los *buffer* de los dispositivos de acceso. Es importante tener en cuenta que el bajo coste computacional que requieren estos mecanismos de control de tráfico es de gran utilidad cuando se trata de dispositivos IoT, los cuales presentan algunas restricciones en cuanto al consumo de energía y procesamiento.

En este trabajo se implementará el algoritmo de bajo costo computacional que se muestra en la Figura 3.10, donde se calculará el *throughput* instantáneo de cada paquete utilizando el tamaño del paquete y el tiempo entre paquetes; luego, se comparará con un umbral de alisado seleccionado. Si el *throughput* instantáneo excede el umbral seleccionado, el paquete será retrasado; de lo contrario, el tiempo de transmisión del paquete no se modificará. De esta forma, es posible obtener un flujo con una tasa de transmisión menor que el umbral seleccionado, optimizando

así el rendimiento de la transmisión. Independientemente del nivel de alisado que se aplique, se debe garantizar que la traza alisada que se obtiene no exceda la duración de la traza original, es decir que si la comunicación entre dos *peer* dura 30 minutos, la traza final alisada no debe tener una duración mayor.

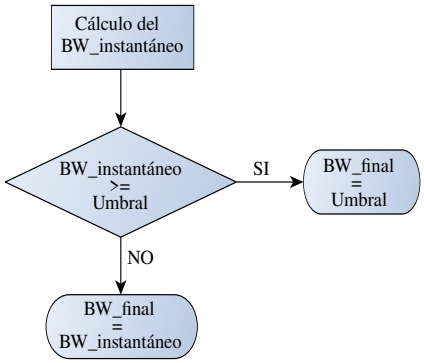

Figura 3.10: Algoritmo para la técnica de alisado.

Para validar el mecanismo de alisado propuesto, se ha aplicado a la traza de SopCast siete niveles de alisado diferentes, los cuales variarán entre 2 y 5 $Mbps$. Se han escogido estos valores de alisado, pues se ha tenido en cuenta las capacidades del enlace de subida que se utilizarán en las pruebas presentadas posteriormente en el capítulo 4. Velocidades de acceso entre 2 y 5 $Mbps$ son comunes en las redes de acceso residenciales o de capacidad moderada. De ahí, que aplicar niveles de alisado fuera de este rango no ofrecería resultados realistas en el escenario planteado. Además, no se deben utilizar niveles de alisado inferiores al *throughput* promedio de la traza de SopCast, puesto que la nueva traza alisada duraría más que la original.

Soy uno de los que piensan como Nobel, que la humanidad sacará más bien que mal de nuevos descubrimientos.

Marie Curie

CAPÍTULO

4

Optimización de tráfico

En capítulos anteriores se ha presentado cómo algunos servicios y aplicaciones presentan problemas a la hora de enviar tráfico a la red, en dependencia de la naturaleza del mismo o cómo han sido implementados. De la misma manera, se ha expuesto que existen limitaciones debido a los dispositivos utilizados en las redes de acceso, puesto que los *buffer* de los *router* podrían influenciar o modificar las características del tráfico de diferentes servicios, como por ejemplo P2P-TV, videovigilancia, entre otros. Cuando ambas situaciones convergen podrían causar la aparición de cuellos de botellas y por ende, la pérdida de paquetes afectando negativamente a la QoS.

En este capítulo, en primer lugar, se presenta un análisis y caracterización del tráfico de una aplicación P2P-TV (específicamente, SopCast), la cual genera gran cantidad de paquetes pequeños y a ráfagas. El conocimiento de dichos parámetros es de vital importancia para determinar si el tráfico obtenido es relevante para este estudio. Para ello se ha implementado un escenario para capturar una traza real y a partir de ella, se han destacado sus principales características: tamaño de los paquetes, tiempo entre paquetes y el *throughput* alcanzado por la aplicación.

En segundo lugar, se estudia la influencia de los *buffer* de los *routers* de las redes de acceso en el tráfico generado por dicha aplicación, teniendo en cuenta el retardo en el propio *buffer* así como también, la pérdida de paquetes provocada. En este caso, se ha utilizado un generador de tráfico que combina una traza de SopCast obtenida del proyecto *Network-Aware P2P-TV Application over Wise Networks*

(NAPA-WINE) [85] con un tráfico de fondo, y se envía a Internet compartiendo el mismo *router*.

En tercer lugar, con el objetivo de garantizar un mejor rendimiento en la transmisión de servicios P2P-TV y reducir la pérdida de paquetes en el enlace de acceso, se evalúan los siguientes sistemas de optimización del tráfico: compresión de cabeceras, multiplexión y alisado. En estos casos, es importante tener en cuenta los niveles de ahorro del ancho de banda obtenidos, pero también los inconvenientes que estos métodos traen consigo como el retardo introducido, la pérdida de paquetes y la influencia sobre otros tráficos que podrían estar compartiendo el mismo enlace de acceso a Internet. En el caso del alisado, se desarrollan dos pruebas diferentes:

- la primera, cuando la aplicación P2P-TV no comparte el enlace de acceso a Internet
- y la segunda, cuando el enlace de acceso es compartido con un tráfico de fondo FTP.

4.1 Características del tráfico en P2P-TV

4.1.1 Análisis del tráfico P2P-TV

Según el tipo de pruebas y simulaciones desarrolladas, se han utilizado diferentes trazas del tráfico de la aplicación P2P-TV seleccionada, las cuales se han obtenido de diversas fuentes. Por un lado, para determinar la influencia mutua entre la implementación de los *buffer* de los dispositivos en las redes de acceso y la naturaleza del tráfico de aplicaciones de *videostreaming* en tiempo real, se han seleccionado trazas del tráfico generado por la aplicación SopCast, obtenidas como parte del proyecto europeo *NAPA-WINE* [85]. Por otro lado, y para el posterior análisis de algunos detalles de la propia aplicación, se ha obtenido y utilizado para el resto de las pruebas una traza real de SopCast obtenida en un escenario controlado de laboratorio bajo condiciones con suficientes recursos computacionales y de energía, de manera que las posibles limitaciones de las redes de acceso no afecten la forma en que el tráfico de red es generado por la aplicación.

En [85] se realizaron varias capturas del tráfico correspondiente a diversas aplicaciones P2P-TV, entre ellas SopCast, utilizando diferentes tecnologías de acceso a Internet. Específicamente, se seleccionaron las trazas del tráfico de SopCast cuando accede a Internet a través de ***(a)*** una red *Local Area Network* (LAN) (*High Bandwidth (BW)* de ahora en lo adelante) y ***(b)*** una red doméstica del tipo *Asymmetric Digital Subscriber Line* (ADSL). Para la generación y obtención de dichas trazas, se utilizaron 44 *peers* distribuidos en diferentes localizaciones geográficas en 4 países. La duración de los experimentos fue de aproximadamente 60 minutos, donde los *peers* tenían seleccionado el mismo canal de TV. En todos los casos, la velocidad de transmisión del video fue de 348 $kbps$ y la calidad del video percibida por todos los usuarios fue muy similar.

Ahora bien, para la obtención de una traza real del tráfico de la aplicación P2P-TV, se implementa el escenario mostrado en la Figura 4.1. Dicho escenario se encuentra en un entorno controlado de laboratorio, empleándose *Tshark* como herramienta de captura y análisis de tráfico. En este caso, se capturaron alrededor de 30 minutos de un partido de fútbol durante la *Champions League* 2013. Se ha utilizado un ordenador convencional (*host* Debian con Linux (kernel $2{,}6{,}38-7$) y con un procesador Intel® Core™ i3 a $2{,}4$ GHz) donde se ejecuta el cliente SopCast, ubicado dentro del campus de la Universidad de Zaragoza con un enlace a Internet de 100 $Mbps$, utilizando de una dirección IP pública y una interfaz de red *Gigabit Ethernet*; cabe especificar que la aplicación P2P-TV se comunicaba con otros 300 *peers* en Internet durante la captura. Como se puede apreciar en la Figura 4.1, se añade un *sniffer* en la mejor ubicación para garantizar que las mediciones no interfieran en el rendimiento de la aplicación.

Una vez obtenida la traza, se procesa y se analiza, siendo el tamaño de los paquetes uno de los principales parámetros a tener en cuenta para una posterior caracterización del tráfico. Un primer elemento a resaltar es que el trasporte del tráfico se lleva a cabo mediante el protocolo UDP. La Figura 4.2 muestra el histograma del tamaño de los paquetes generados por el cliente SopCast, donde se puede apreciar que aproximadamente el $50\,\%$ de los mismos se corresponden a paquetes pequeños de tamaños menor o igual a 100 $bytes$ y alrededor de un $45\,\%$ a paquetes grandes de aproximadamente 1300 $bytes$. Los paquetes grandes transmitidos

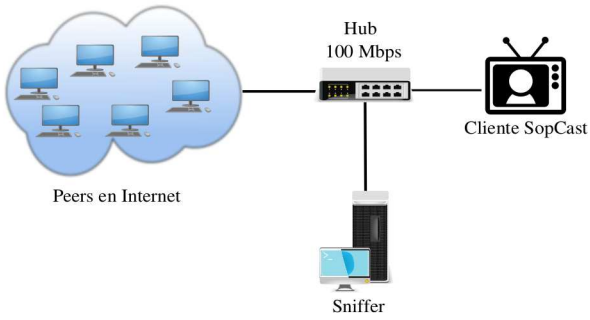

Figura 4.1: Escenario para capturar el tráfico.

transportan la información del video intercambiado con otros *peers* en Internet. Los paquetes pequeños son utilizados por la capa de aplicación para gestionar el estado de cada *peer* presente en la comunicación, pero también para monitorizar, controlar y reorganizar los paquetes de video [86].

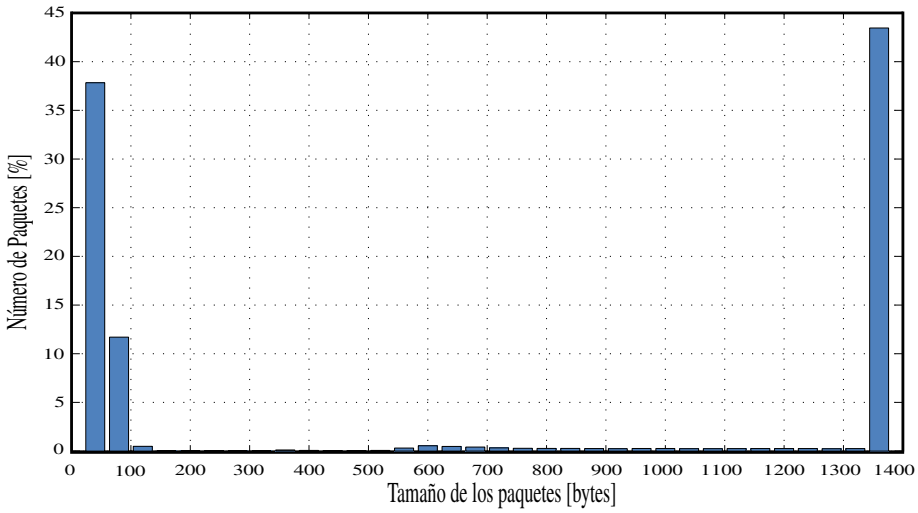

Figura 4.2: Histograma del tamaño de los paquetes.

Para las pruebas se utiliza el tráfico intercambiado con el *peer* que provee más del 90 % del video durante toda la comunicación y con el cual, por tanto, existe

la mayor cantidad de paquetes intercambiados [32]. Analizando este tráfico, se nota que cada paquete de video es confirmado por un paquete ACK de nivel de aplicación, con una carga útil de 28 $bytes$, como se puede observar en la Figura 4.3. Este hecho genera una gran cantidad de paquetes pequeños, y por lo tanto, una baja eficiencia en términos de uso de recursos de red. La capa de aplicación utiliza paquetes pequeños para administrar el estado de cada *peer*, pero también para monitorizar y controlar los paquetes de video a fin de reconstruir y reproducir adecuadamente el contenido.

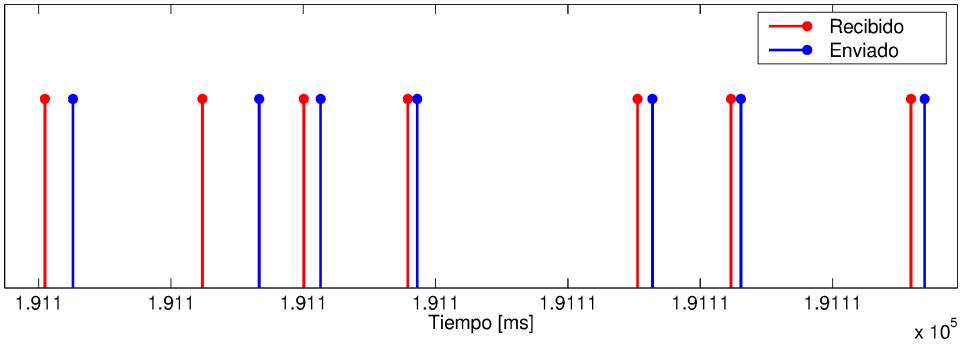

Figura 4.3: Tráfico durante una comunicación entre dos *peer* (paquetes de video y de confirmación).

Desde el punto de vista de cómo se realiza el envío de los paquetes, de la Figura 4.4 se puede inferir que el tráfico de la aplicación P2P-TV presenta una amplia dispersión en cuanto al tiempo entre paquetes y no se percibe una tasa uniforme de entrega de los mismos. Esta aleatoriedad dificulta la búsqueda de un modelo estadístico para este tipo de tráfico, por lo cual se ha decidido utilizar una traza real para las pruebas, como se ha mencionado con anterioridad.

La Figura 4.5 muestra la *Empirical Cumulative Distribution Function (ECDF)* del tiempo entre paquetes de la traza, observándose con más detalles la dispersión antes mencionada. Como bien se puede apreciar, cerca de un 30 % de los paquetes se envían con un tiempo entre paquetes menor a 100 microsegundos, alrededor de un 20 % entre 100 microsegundos y 1 milisegundo, otro 40 % entre 1

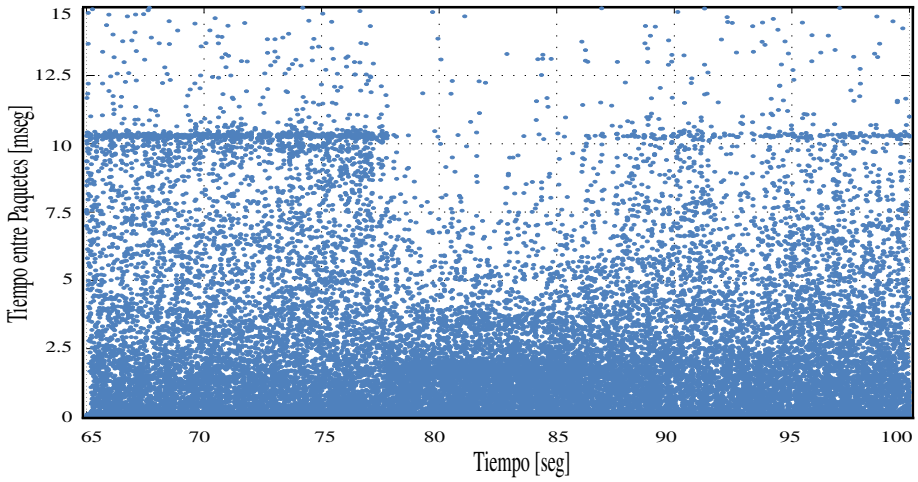

Figura 4.4: Tiempo entre paquetes para el tráfico utilizado.

y 10 milisegundos y un 10 % superior a 10 milisegundos, que podría llegar hasta 1 segundo. Al analizar el tráfico se detectaron muchas ráfagas, compuestas por paquetes con tiempos entre ellos muy pequeños. En la Figura 4.5 se observa que aproximadamente un 50 % tienen un tiempo menor o igual a 1 milisegundo. Bajo estas condiciones de tráfico a ráfagas, pueden producirse pérdidas de paquetes y la QoS podría deteriorarse en dispositivos con más baja capacidad.

De la misma manera que el tamaño de los paquetes y el tiempo de llegada entre ellos es importante en el análisis y caracterización del tráfico, también es necesario tener en cuenta el *throughput* alcanzado por la aplicación P2P-TV. Para estudiar el intervalo de tiempo más crítico en el comportamiento de la aplicación en cuanto a la demanda de recursos de red, se selecciona una muestra representativa de su tráfico, donde se concentran la mayor cantidad de picos de *throughput*. La muestra seleccionada tiene una duración de 18 minutos (recuadro rojo en la Figura 4.6) y el *throughput* promedio de la aplicación es de aproximadamente 1,84 $Mbps$.

4.1.2 Retardo en *videostreaming*

En ocasiones, el envío de datos entre un nodo y un servidor donde se procesara la información enviada, se ve afectado por el retardo entre estos dos dispositivos.

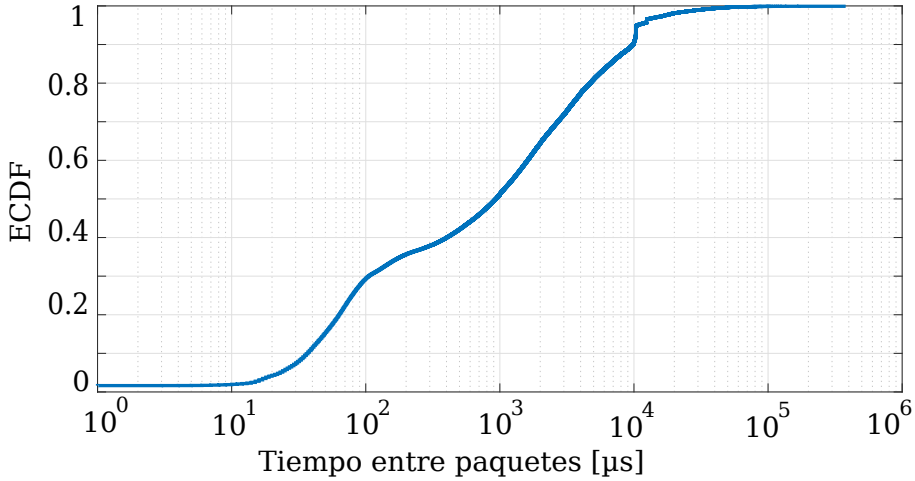

Figura 4.5: ECDF del tiempo entre paquetes para una transmisión de *videostreaming* P2P.

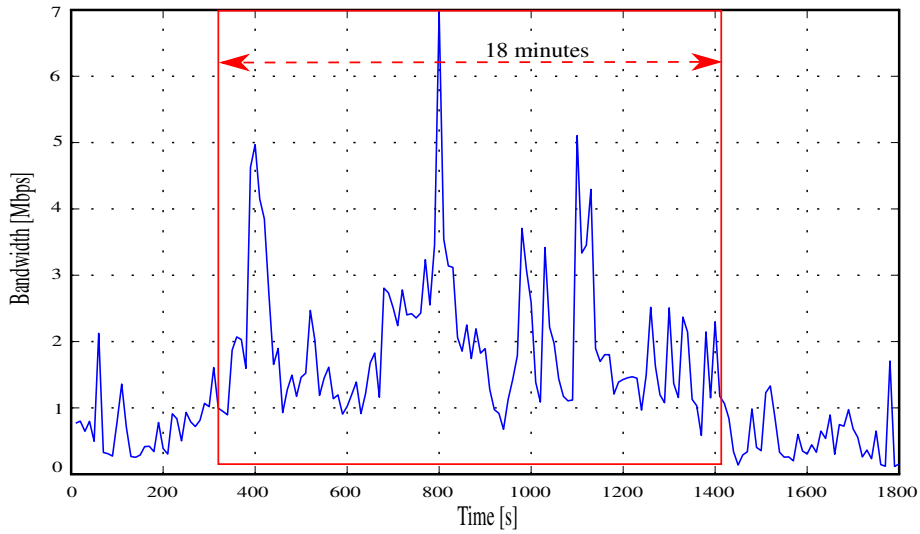

Figura 4.6: Muestra de tráfico utilizada en las pruebas.

Con el siguiente experimento se quiere demostrar que no solamente el retardo se incrementa, sino que el comportamiento del tráfico de la aplicación cambia, deteriorándose la calidad de la misma.

Para analizar el efecto que el retardo tiene en el tiempo entre paquetes de una aplicación de *videostreaming*, se ha utilizado un *Raspberry Pi* como nodo *Edge-Cloud*, que envía tráfico de video a un servidor ubicado en una nube privada implementada en OpenStack (la misma utilizada para la capa de *Gestión Inteligente* [87]). De esta forma se mantienen las mismas condiciones que si se utilizara un *Cloud Service Provider* (CSP), y por lo tanto, no se altera la arquitectura propuesta en 3.1. En cada prueba, el nodo envía una transmisión de video al servidor en formato *Motion JPEG* (MJPEG) utilizando la librería de Python video4Linux, la cual implementa por defecto TCP para el transporte de datos. Al mismo tiempo, se agrega un retardo a la tarjeta de red del servidor (usando la herramienta *traffic control* en Linux) para emular el retardo desde el nodo a la nube. Para obtener valores reales de retardo, se realiza *ping* a tres regiones (o centros de datos) de *Amazon Web Services* (AWS) (de forma similar que en [88]). Los valores promedio de 100 repeticiones a cada localidad son: Londres ($0{,}451$ milisegundos), Frankfurt (22 milisegundos) y Virginia (79 milisegundos).

En la Figura 4.7 se muestra la comparativa del tiempo entre paquetes cuando se transmite video a las 4 localidades mencionadas. El eje *"Y"* representa el tiempo entre paquetes, mientras que el eje *"X"* representa el tiempo en el que cada paquete ha sido transmitido. Como se observa, a medida que la ubicación cambia (el retardo emulado), no solo aumenta el tiempo entre paquetes (lo que es de esperar), sino que además, es evidente que en un mismo período hay menos paquetes. El aumento del retardo produce mayor dispersión en la generación de paquetes por parte de la aplicación, es decir hay más aleatoriedad en los tiempos de envío de paquetes, lo cual se aprecia mejor al comparar la Figura 4.7a con la Figura 4.7c. A pesar de que el retardo está en un rango tolerable para muchas aplicaciones, de $0{,}451$ milisegundos en la Figura 4.7b hasta 79 milisegundos en la Figura 4.7d, se aprecia que el impacto que tiene el protocolo de transporte (en este caso TCP) en el tiempo entre paquetes influye en cómo el tráfico es enviado a la red. Por ejemplo, bajo posibles condiciones de congestión o pérdida de paquetes, TCP podría incrementar

el retardo experimentado por la aplicación debido a sus mecanismos de control de congestión, pudiendo deteriorar la calidad. Está claro que el principal beneficio de *Edge Cloud* es que las operaciones de procesamiento tienen menos retardo, ya que están más cerca de la fuente de información. Por otro lado, el comportamiento del tráfico de red de las aplicaciones TCP puede verse afectado negativamente (más allá del retardo adicional), lo que a su vez empeorará la QoS.

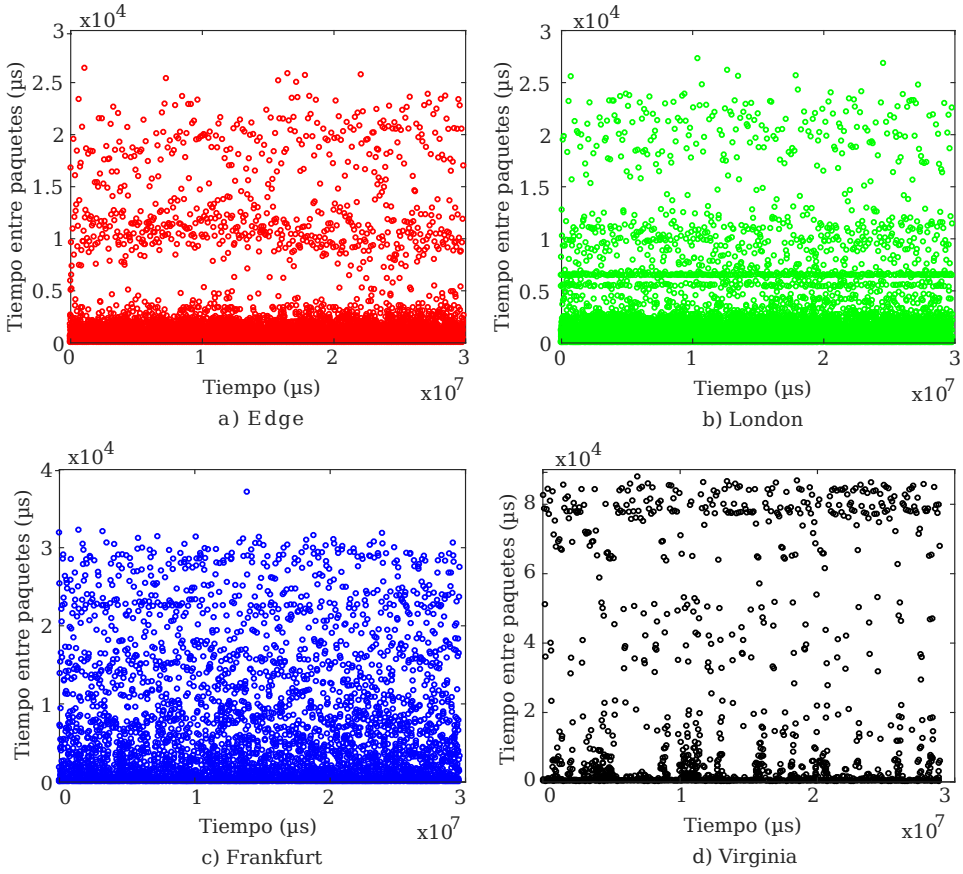

Figura 4.7: Comparación del tiempo entre paquetes para diferentes ubicaciones de un servidor en la nube.

4.1.3 Influencia del *buffer* en P2P-TV

A continuación se presenta un análisis de las características de los *buffer* de los *routers* en las redes de acceso, fundamentalmente su tamaño y la pérdida de paquetes y cómo estas características podrían afectar el comportamiento y la calidad de las aplicaciones multimedia cuando las mismas generan tráfico a ráfagas. También, se presenta cómo el aumento de la capacidad de la red interna puede causar desbordamiento de los *buffer* y producir pérdidas de paquetes de tal magnitud que podrían deteriorar la QoS.

Como se mencionó con anterioridad, es importante tener en cuenta las características de los *buffer* (especialmente su tamaño y la pérdida de paquetes) en los dispositivos de acceso, ya que se puede conocer cómo estas características afectan a la calidad de las aplicaciones multimedia cuando éstas generan tráfico a ráfagas en la red local.

Para las pruebas se ha utilizado tráficos provenientes de diferentes aplicaciones multimedia: videovigilancia y *videostreaming* P2P-TV.

En estas pruebas, se utilizan dos trazas de la aplicación P2P-TV obtenidas del proyecto NAPA-WINE: la primera cuando el cliente SopCast utiliza un enlace *High BW* de acceso a Internet y la segunda con un enlace ADSL [85]. El escenario utilizado para las pruebas se muestra en la Figura 4.8, donde el tráfico de SopCast y el tráfico de fondo se envían usando el generador *Jugi's Traffic Generator* (JTG) [89], el cual es capaz de enviar las trazas exactamente como son, ya que JTG es capaz de leer el tamaño de los paquetes y el tiempo entre paquetes de los ficheros descargados de la página web del proyecto NAPA-WINE, por lo que no ha sido necesario modelar el tráfico de la aplicación P2P-TV. Además, el tráfico de fondo empleado tiene la siguiente distribución de paquetes: el $50\,\%$ son de $40\ bytes$, el $10\,\%$ de $576\ bytes$, y el $40\,\%$ restante de $1500\ bytes$ según [90].

Se ha utilizado *Matlab* como herramienta para procesar simulaciones en las que el tráfico P2P y un tráfico de fondo comparten el mismo enlace de acceso a Internet. Las simulaciones se repiten para cada una de las trazas de SopCast y se han usado tres *buffer* con tamaño definido en *bytes* (10, 100 y 1000 $kbytes$) y otros tres limitados en número de paquetes (27, 270 y 2700) paquetes, para un total de

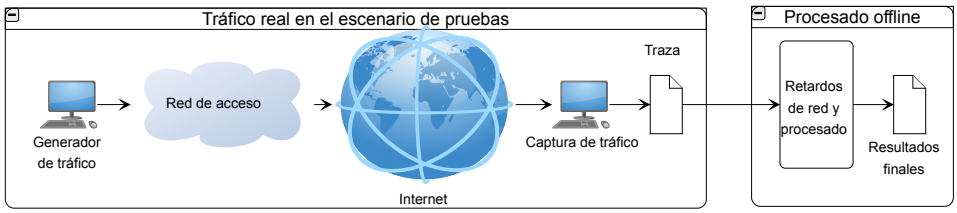

Figura 4.8: Escenario para las pruebas.

seis diferentes implementaciones del *buffer*. Los anchos de banda del enlace de subida empleados fueron 512, 1024 y 2048 $kbps$. A continuación, se analizan el retardo y las pérdidas para las distintas políticas de implementación del *buffer* empleadas cuando se utilizan diferentes valores de tráfico de fondo para saturar el *buffer* del *router*.

La Figura 4.9 muestra el retardo en el *buffer* cuando el enlace tiene una capacidad de 1024 $kbps$ y se emplea la traza *High BW*. Se puede apreciar que el retardo se incrementa conforme se va alcanzando el límite del ancho de banda y los *buffer* tienden a llenarse. Esta figura ilustra que el comportamiento es muy similar para ambas implementaciones de los *buffer* (tamaño y paquetes) siendo ligeramente mayor para aquellos limitados en paquetes. Cuando el ancho de banda del enlace es de 512 y 2048 $kbps$ y para la traza ADSL, se observa el mismo patrón en cuanto al comportamiento del retardo.

En las Figura 4.10 y Figura 4.11 se muestran las pérdidas de paquetes para un *peer* usando la traza *High BW* y para los tres posibles valores de ancho de banda. En este caso, las pérdidas en los *buffer* limitados en número de paquetes son mucho mayores comparadas con las pérdidas obtenidas cuando los *buffer* son limitados en tamaño. Esto se debe a la gran cantidad de paquetes por segundo que genera la aplicación P2P-TV y que ocupan un lugar en el *buffer* sea cual sea su tamaño.

En las Figura 4.12 y Figura 4.13 se observa un patrón de comportamiento de las pérdidas similar al de las Figura 4.10 y Figura 4.11, pero en este caso, utilizando la traza ADSL, donde los *buffer* limitados en tamaño presentan pérdidas mucho

menores que aquellos limitados en número de paquetes.

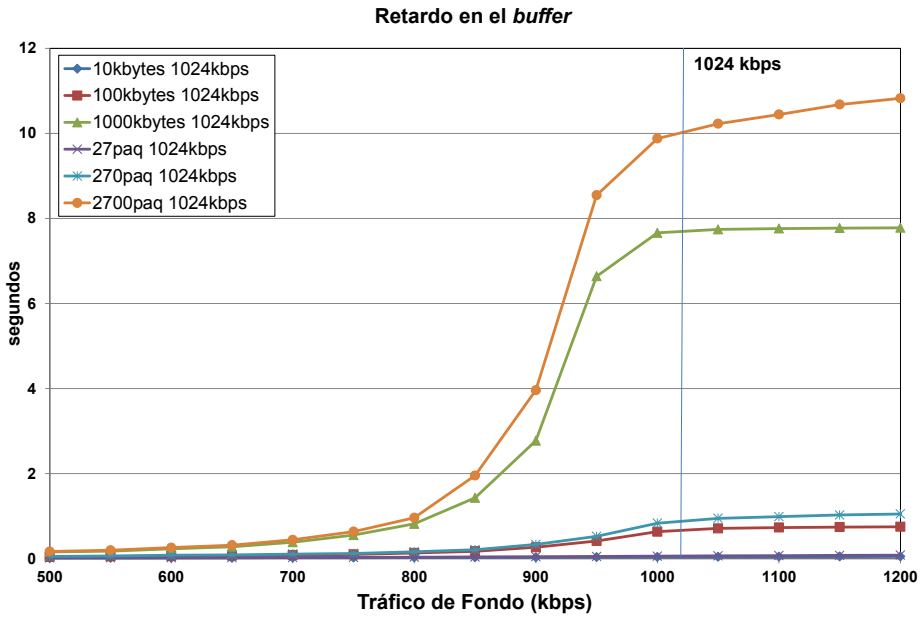

Figura 4.9: Retardo en el *buffer* cuando se utiliza un enlace a 1024 $kbps$ y la traza *High BW*.

En las Figura 4.14 y Figura 4.15 se presentan las pérdidas en el *buffer* según el tipo de paquetes (grandes o pequeños). Estas figuras se muestran solo para un ancho de banda de salida de 1024 $kbps$ pues como se ha mencionado el comportamiento es muy similar para 512 y 2048 $kbps$. Para los *buffer* medidos en *bytes* la pérdida de paquetes pequeños es mínima si se compara con la de los paquetes de video, que son de tamaño mucho mayor. Como consecuencia de esto, puede ocurrir que el *peer* no contribuya suficientemente en la distribución de video a otros usuarios. En cambio, para los *buffer* limitados en número de paquetes se desechan en la misma medida tanto los paquetes de video como los de señalización. A pesar de su pequeño tamaño, cada paquete de señalización ocupa un puesto en el *buffer*. Esto hace que la cola se llene más rápido, penalizando a todos los paquetes y perjudicando tanto a la distribución como a la propia recepción del video.

El efecto anteriormente descrito es independiente de la tecnología de acceso empleada. En las redes *High BW* al incrementarse el ancho de banda de salida, los

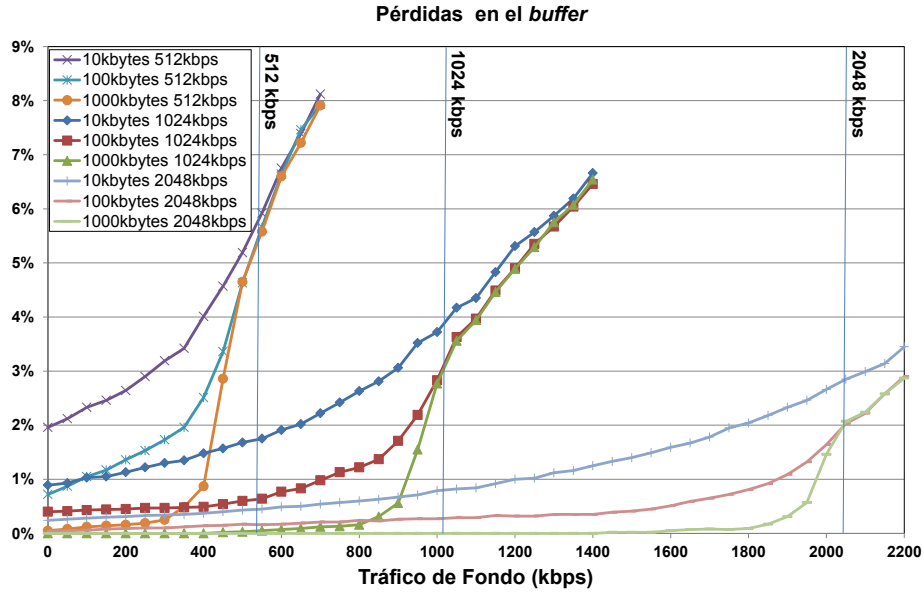

Figura 4.10: Pérdidas para el *buffer* limitado en tamaño con la traza *High BW*.

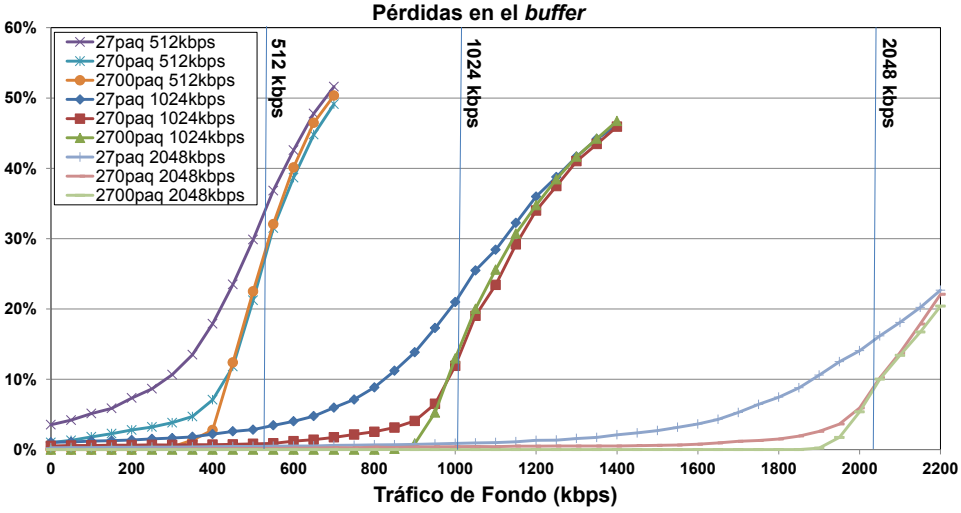

Figura 4.11: Pérdidas para el *buffer* limitado en número de paquetes con la traza *High BW*.

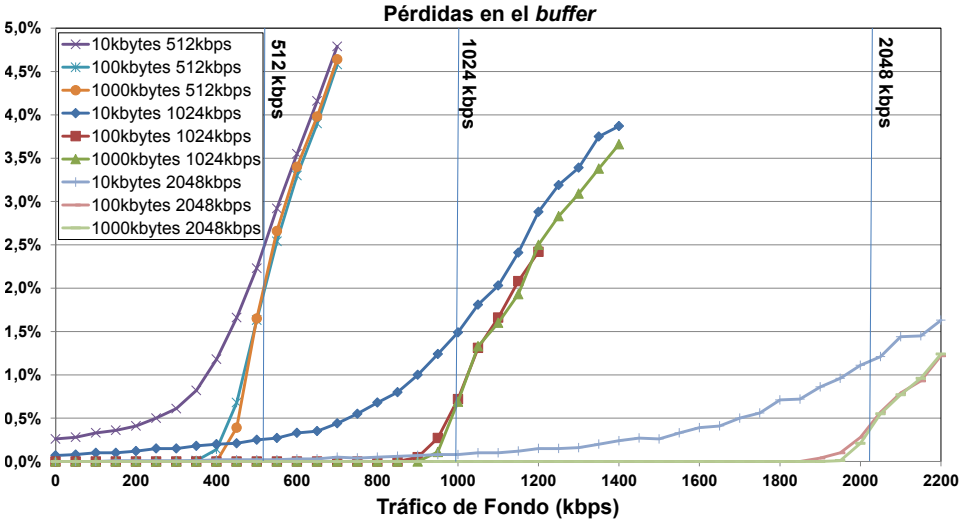

Figura 4.12: Pérdidas para el *buffer* limitado en tamaño con la traza ADSL.

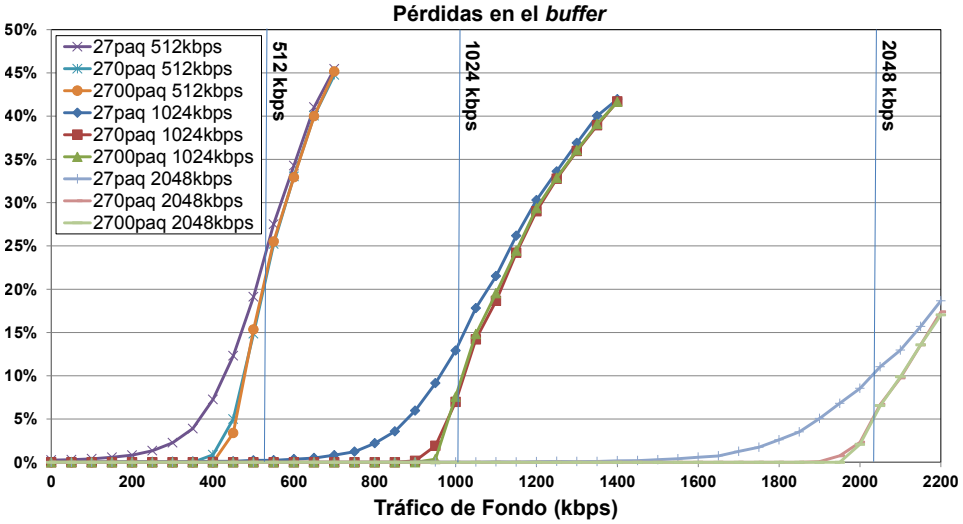

Figura 4.13: Pérdidas para el *buffer* limitado en número de paquetes con la traza ADSL.

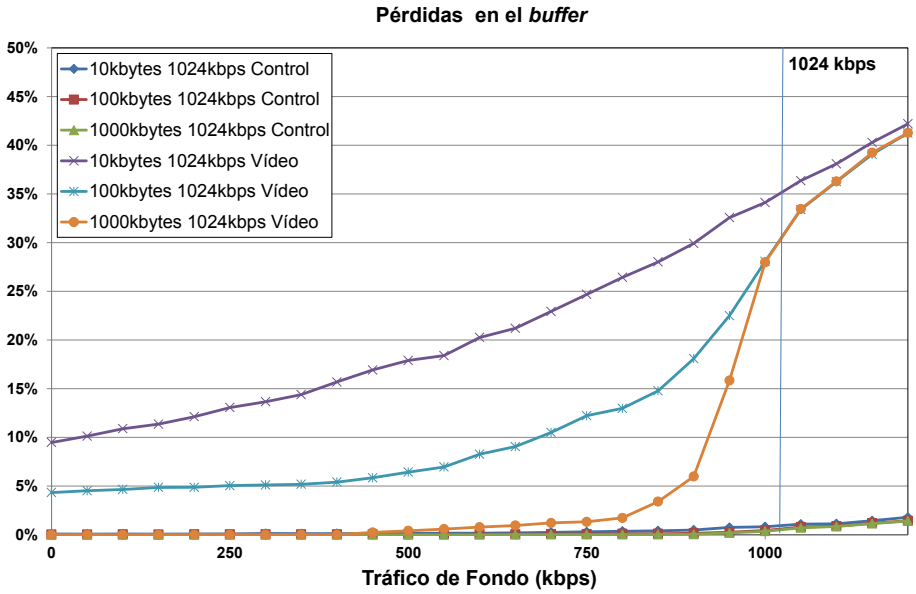

Figura 4.14: Pérdidas según el tipo de paquetes para el *buffer* limitado en tamaño con la traza *High BW* y 1024 $kbps$.

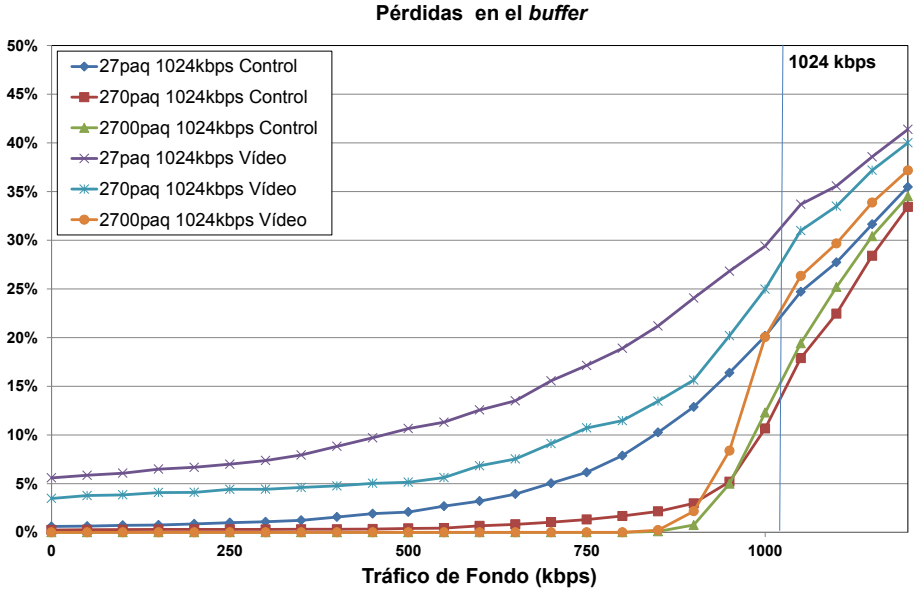

Figura 4.15: Pérdidas según el tipo de paquetes para el *buffer* limitado en número de paquetes con la traza *High BW* y 1024 $kbps$.

peer podrían enviar mayor número de paquetes de video contribuyendo en mayor medida a la distribución. Sin embargo, como se puede apreciar en las Figura 4.14 y Figura 4.15 las pérdidas pueden llegar a ser considerables si el *buffer* no es lo suficientemente grande.

Igualmente se puede considerar, que el tráfico de fondo de paquetes grandes que se está desechando junto al tráfico de paquetes de video, podría corresponderse a otro tipo de servicios como FTP y HTTP. El buen funcionamiento de estos otros servicios que comparten el mismo enlace de acceso a Internet también se vería perjudicado. Por otro lado, la utilización en el mismo enlace de servicios que utilicen paquetes pequeños, como VoIP y juegos *online* podrían llegar a perjudicar la distribución de video P2P.

4.2 Técnicas de conformado de tráfico

Como se ha mencionado en capítulos anteriores, existen diversas técnicas para mejorar la utilización del enlace de acceso. En esta sección se presentan los resultados obtenidos cuando el tráfico de la aplicación P2P-TV es modificado o adaptado, haciendo uso de los diferentes algoritmos de multiplexión y de alisado descritos en la sección 3.3 de la Metodología. Se puede observar, en ambos casos, mejorías en cuanto al ahorro del ancho de banda y pérdidas de paquetes, mejorándose por ende la QoS de las aplicaciones P2P que generan un gran cantidad de paquetes pequeños.

4.2.1 Técnicas de multiplexión

En el capítulo de metodología se describieron los dos métodos de multiplexión utilizados (basados en: (a) *períodos* y (b) *umbrales*) con el propósito de analizar cómo esta técnica de conformado de tráfico influye, fundamentalmente, en el ahorro del ancho de banda del enlace de acceso a Internet. De igual manera, se estudia cómo el número de paquetes por segundo generado contribuye a un mejor comportamiento de los dispositivos de acceso a la red ante este tipo de tráfico.

Es necesario resaltar que, primeramente y con el fin de comprobar que los retardos añadidos (en el orden de las decenas o centenas de milisegundos) por las

técnicas de optimización empleadas no afectan a la visualización del video, se ha realizado una prueba utilizando *Network Emulator* (Netem) para filtrar los paquetes ACK enviados desde la aplicación local al resto de los *peers* durante la visualización de un video. A medida que los *peers* dejan de recibir las confirmaciones, asumen que se ha desconectado la sesión y por tanto dejan de enviar contenido. Aún así, el *streaming* de video se sigue reproduciendo sin problemas en el ordenador local durante aproximadamente 1 minuto. El comportamiento observado cuadra con [43], donde se llega a la conclusión de que el tamaño del *buffer* de SopCast tiene esa misma duración.

Para las pruebas aquí presentadas, y como se ha indicado previamente, se utiliza el tráfico intercambiado con el *peer* que provee más del $90\,\%$ del video durante la comunicación. También, se han seleccionado valores entre 10 y 50 milisegundos para definir el *período* y los *umbrales* utilizados en las simulaciones, puesto que más del $84\,\%$ de los paquetes presentan un intervalo de llegada en este rango.

La Figura4.16 muestra el ahorro del ancho de banda que se obtiene para ambas políticas. Primeramente se observa que los valores de ahorro de ancho de banda alcanzados son significativos, notándose entre un $25\,\%$ y el $35\,\%$ del total enviado por el *peer* local. Cuando se emplea la política basada en un *umbral* de tiempo entre paquetes, se obtiene un *Bandwidth Saving* (BWS) entre un $33\,\%$ y un $35\,\%$. Como el tiempo entre los paquetes pertenecientes a una misma ráfaga es aproximadamente 10 milisegundos en la mayor parte de los casos, si se selecciona dicho valor como *umbral*, no se obtienen valores óptimos de BWS, pues se multiplexan muy pocos paquetes. Sin embargo, para *umbrales* superiores a $12{,}5$ milisegundos, el BWS presenta un mejor comportamiento y se aprecia que se mantiene prácticamente constante para el resto de los *umbrales* seleccionados. Para la política basada en un *período*, el BWS alcanzado varía entre el $26\,\%$ y el $33\,\%$. De manera similar a los resultados obtenidos en [61], los valores de BWS presentan un comportamiento asintótico.

En las Figura 4.17 y Figura 4.18, se presentan los histogramas del tamaño de los paquetes multiplexados en ambas políticas, usando un *período* y un *umbral* de 20 milisegundos. Comparando los resultados obtenidos, se observa que con el uso del *período* se genera mayor cantidad de paquetes pequeños, y muy pocos

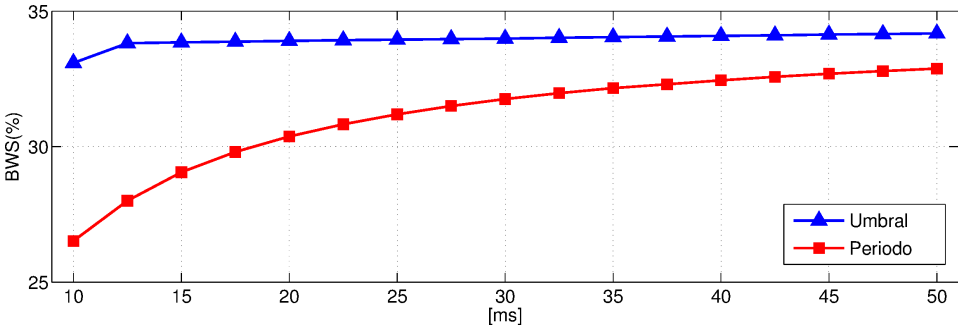

Figura 4.16: Ahorro de Ancho de Banda usando las dos políticas de multiplexión.

paquetes grandes, con respecto a la política basada en un *umbral*. En el segundo caso, se consigue multiplexar una mayor cantidad de paquetes, al adaptarse mejor al tráfico generado. Este resultado avala los mejores resultados de BWS para esta política.

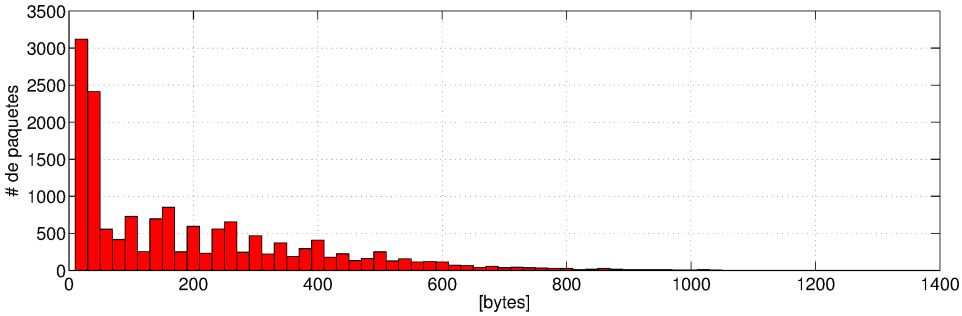

Figura 4.17: Histograma del tamaño de los paquetes multiplexados para la política de *período*.

La Figura 4.19 presenta el número de paquetes por segundo (pps) generados en el caso del tráfico *nativo* y cuando se utiliza cada una de las políticas de multiplexión. Como bien se puede ver, hay una reducción significativa de este parámetro, disminuyendo desde aproximadamente unos 50 pps hasta unos 5 pps al multiplexar. Esta disminución resulta interesante a la hora de reducir la carga en los *router* y al mismo tiempo, se muestra que con el aumento del *período* o el *umbral* aumenta

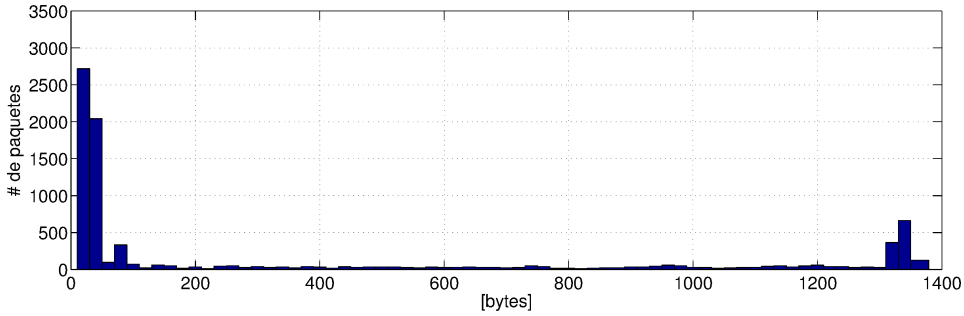

Figura 4.18: Histograma del tamaño de los paquetes multiplexados para la política de *umbrales*.

también el número de paquetes multiplexados. Sin embargo, existe una diferencia: mientras aumenta el valor del *período*, la cantidad de paquetes por segundo disminuye; en el caso del *umbral*, llega un momento en que su incremento no produce ninguna mejora y la cantidad de paquetes por segundo permanece prácticamente constante para valores superiores a 12,5 milisegundos.

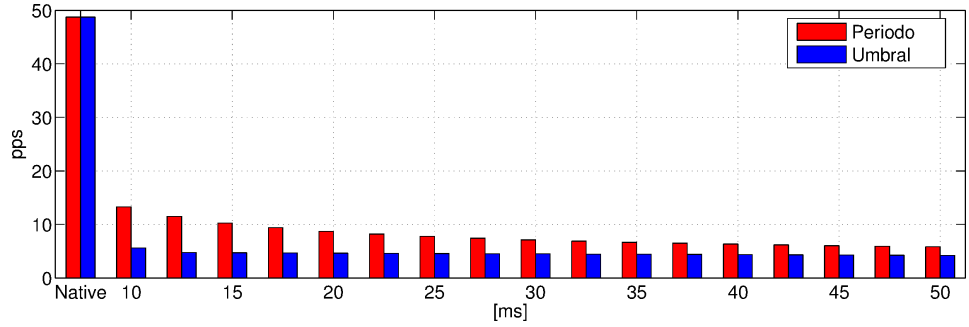

Figura 4.19: Paquetes por segundo.

A partir de los resultados de las simulaciones y los valores de ahorro de ancho de banda y paquetes por segundo obtenidos, se ha mostrado que multiplexando el tráfico P2P-TV, aún cuando se considera la comunicación con un único *peer*, se logra un buen ahorro de tráfico y de paquetes por segundo, reduciendo también las necesidades de procesamiento. Estos resultados son muy prometedores para las redes residenciales, donde el enlace de subida (*uplink*) es limitado y se comparte con

otros servicios. Si se considera además que los usuarios utilizan normalmente *router* de gama media y baja con capacidad de procesamiento limitada, la reducción del numero de paquetes por segundo conseguida adquiere mayor relevancia.

4.2.2 **Algoritmos de alisado**

Con el fin de validar la metodología propuesta al modificar o adaptar el tráfico de una aplicación P2P-TV mediante un algoritmo de alisado, se han propuesto dos pruebas diferentes. En ambas pruebas, se analizan la pérdida de *bytes* y de paquetes, y también el *throughput* logrado por diferentes aplicaciones cuando comparten un mismo enlace de subida. Inicialmente, se hace un análisis de la pérdida de *bytes* y de paquetes por separado, aunque son dos parámetros muy relacionados. Por un lado, analizar la pérdida de *bytes* ayudará a determinar si el *throughput* de la aplicación aumenta o disminuye. Por otro lado, es necesario determinar la naturaleza de los paquetes perdidos (si son grandes o pequeños), ya que el funcionamiento de la aplicación P2P-TV se vería afectado en función de qué paquetes se pierden (como se explicó en 4.1.1).

En la Prueba 1, los parámetros de QoS se analizan cuando la aplicación P2P-TV tiene acceso a toda la capacidad del enlace de subida. Por otro lado, en la Prueba 2 se estudia un entorno más realista, donde la aplicación P2P-TV comparte el enlace de subida con un tráfico de fondo, el cual consiste en un servicio FTP que utiliza TCP-SACK (una de las variantes más utilizadas del protocolo TCP [30]) como protocolo de transporte. En la Prueba 2 se analizan la tasa de pérdida y el *throughput* alcanzado por ambas aplicaciones.

En ambas pruebas, se seleccionaron siete valores para la capacidad de enlace ascendente (2, 2,5, 3, 3,5, 4, 4,5 y 5 $Mbps$), y para cada uno de ellos, a la traza P2P utilizada se le aplica siete niveles de alisado (2, 2,5, 3, 3,5, 4, 4,5, 5 $Mbps$), además de tenerse en cuenta la traza sin alisar, por lo que se tienen un total de 56 situaciones diferentes para cada una de las pruebas. Los resultados muestran el promedio de 100 simulaciones para cada situación. La traza con duración de 18 minutos, referida anteriormente, se ha utilizado en ambas pruebas. Todos los

datos podrían ser comparables entre diferentes tecnologías de acceso (por ejemplo, *Ethernet*, ADSL y *cable módem*) utilizando capacidades de enlace similares, debido a que el análisis de los flujos se realiza a nivel de IP.

Como contrapartida, al alisar la traza de la aplicación P2P-TV se añaden retardos a cada paquete de la misma, por lo que es necesario realizar un análisis adecuado del retardo añadido y de esta manera poder seleccionar cuáles son los niveles de alisado más adecuados que se podrían utilizar sin afectar o modificar sobremanera el funcionamiento de dicha aplicación. Para mostrar el peor de los casos (mayor retardo introducido), la Figura 4.20 presenta un zoom del retardo añadido a cada paquete para cada uno de los niveles de alisado (series de 2 a 5 $Mbps$). Para el nivel más alto de alisado (2 $Mbps$), se introducen retardos intolerables por la aplicación de *videostreaming* y obviamente, para cualquier otro servicio en tiempo real (aproximadamente unos 90 s). Sin embargo, a medida que disminuye el umbral de alisado, también disminuyen el valor y la frecuencia de los retardos añadidos. En la Figura 4.20 se puede observar que una cantidad significativa de paquetes no cambian su tiempo de envío, y por tanto se podría deducir que los retardos añadidos ocurren cuando los paquetes llegan en ráfagas.

Para cada nivel de alisado, se obtienen los retardos añadidos a cada paquete con el objetivo de determinar el umbral de alisado apropiado a aplicar a este tipo de flujo (Figura 4.20). Como ejemplo, se puede mencionar que al 80 % de los paquetes, de la traza alisada a 3 $Mbps$, se les ha añadido un retardo inferior a los 20 s, el cual es un valor tolerado por la aplicación utilizada de acuerdo con [43]. Es por ello que, se ha seleccionado la traza alisada a 3 $Mbps$ en las pruebas que se discuten a continuación. Estos valores de retardo se han seleccionado en función del comportamiento del tráfico, para controlar la pérdida de paquetes causada por la llegada del tráfico a ráfagas y para mejorar la calidad de la comunicación.

4.2.2.1 **Escenario de las pruebas**

La Figura 4.21 muestra el escenario utilizado para las Pruebas 1 y 2, donde se analiza el comportamiento de los parámetros de QoS. Para la selección e implementación del escenario de pruebas se han considerado puntos sensibles y críticos

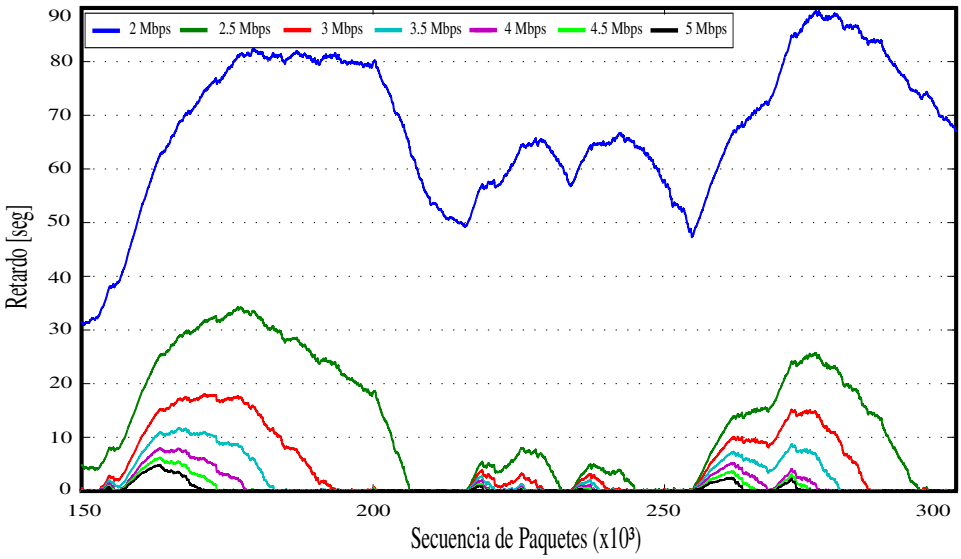

Figura 4.20: Retardo introducido para cada nivel de alisado.

de las redes de acceso, principalmente los entornos domésticos los cuales presentan limitaciones como ya se ha mencionado. En este caso el enlace de subida es el más restrictivo y puede ser compartido por el tráfico de diversas aplicaciones. Más específicamente, los resultados muestran el comportamiento del enlace entre el Nodo 0 y el Nodo 1, analizándose algunos parámetros de QoS, como la pérdida de *bytes* y paquetes, el retardo y el *throughput* alcanzado por cada una de las aplicaciones que comparten el enlace.

En la red interna, un cliente P2P-TV y un cliente FTP comparten el mismo enlace de acceso a Internet, siendo el foco principal de este análisis. El otro extremo está compuesto por una red externa, con cientos de *peers* P2P-TV dispersos por Internet, y un servidor FTP. La capacidad del enlace de subida varía entre 2 y 5 $Mbps$, se ha tenido en cuenta el análisis preliminar del *throughput* medio alcanzado por la aplicación P2P-TV seleccionada. No se han considerado capacidades de enlace inferiores a 1,84 $Mbps$, para evitar el deterioro del rendimiento de la aplicación.

En muchas tecnologías de redes de acceso, el enlace de subida es el enlace más

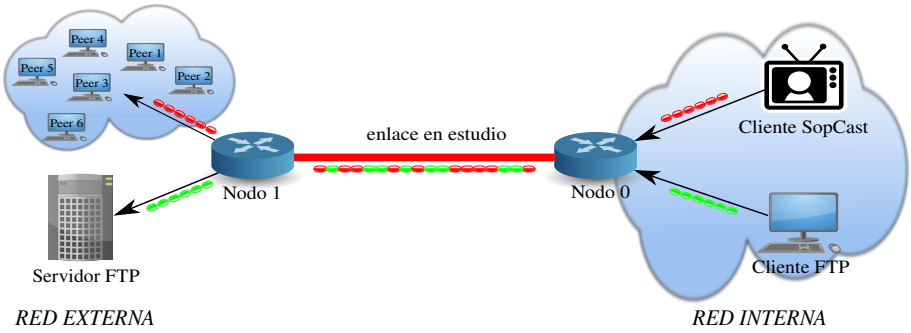

Figura 4.21: Escenario de pruebas.

restrictivo con una capacidad significativamente menor en comparación con el enlace de bajada; por lo tanto, es el lugar donde el estudio de este fenómeno puede resultar más interesante. En el Nodo 0, ciertos parámetros de red son modificados: se implementa un *buffer* de 50 paquetes con una política de gestión *"drop-tail"* (estos valores son ampliamente utilizados en dispositivos de acceso comercial [50]), y la capacidad del enlace se varía, como se mencionó anteriormente. Además, el enlace de bajada está dimensionado con una capacidad de enlace de 100 $Mbps$ y un *buffer* de 500 paquetes, para evitar pérdidas en este enlace.

La topología de red propuesta en la Figura 4.21 se ha implementado usando *Network Simulator* (NS). Se hace uso de la traza real de la aplicación de *videostreaming* capturada y procesada previamente y luego se generan como flujos UDP en NS, utilizando los mismos tamaños de paquetes y tiempos entre paquetes de la traza original. Para el flujo de datos FTP (Prueba 2), se emplearon agentes TCP-SACK y FTP, utilizando las implementaciones estándares de NS para dichos protocolos. Como mismo ocurre en un escenario real, los flujos no se inician simultáneamente, por lo que la simulación tiene un período inicial en el cual los flujos comienzan aleatoriamente.

4.2.2.2 **Resultados de la Prueba 1**

Las Figura 4.22 y Figura 4.23 representan el porcentaje de pérdida de *bytes* y de paquetes para la aplicación P2P-TV, respectivamente. En el eje *"X"* se muestran

los niveles de alisado (de 2 $Mbps$ a 5 $Mbps$) utilizados en las pruebas; también se incluyen los resultados para la traza sin alisar. Cada serie representa una capacidad de enlace diferente. Como es de esperar, en todos los casos la pérdida de *bytes* y paquetes disminuye a medida que la capacidad del enlace es mayor, debido a que el enlace presenta una capacidad suficiente para soportar las ráfagas generadas por la aplicación P2P-TV, y por lo tanto, la pérdida de paquetes en el *buffer* es menor.

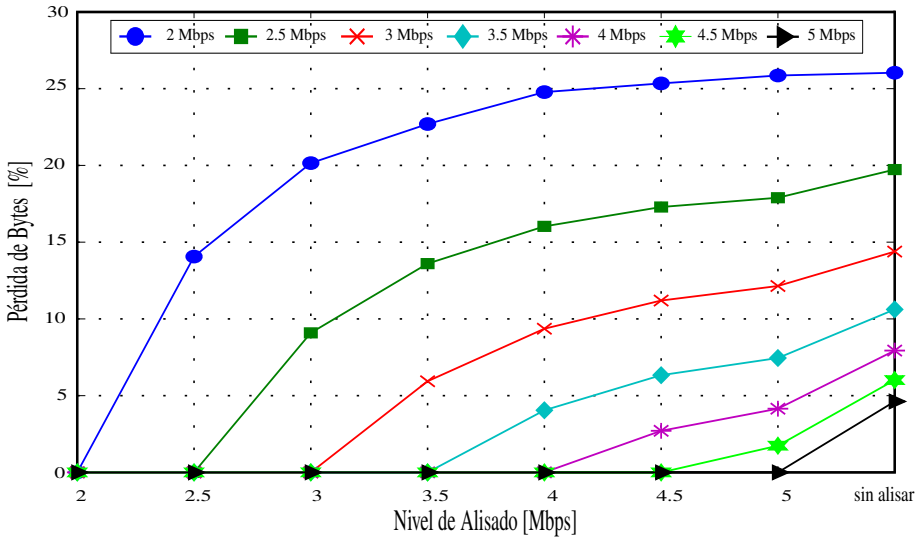

Figura 4.22: Pérdida de paquetes P2P-TV en *bytes*.

Por ejemplo, en la Figura 4.22, para la traza sin alisar y una capacidad de enlace de 2 $Mbps$, la pérdida de *bytes* es aproximadamente del 26 %, mientras que para un enlace de 5 $Mbps$ se reduce a casi al 5 %. Además, aunque los umbrales de alisado disminuyen, también lo hace la tasa de pérdida de *bytes*. Cuando el nivel de alisado es igual o inferior a la capacidad del enlace, la tasa de pérdida se reduce al 0 %. Esto ocurre porque la traza alisada no presenta picos de *throughput* por encima del umbral de alisado utilizado, y por lo tanto, la probabilidad de pérdida causada por el desbordamiento del *buffer* se anula. Como ejemplo, para una capacidad de enlace de 3,5 $Mbps$, la traza sin alizar tiene aproximadamente un 10 % de pérdida de *bytes* y a medida que los umbrales de alisado disminuyen, la pérdida de paquetes

desaparece. Este patrón es el mismo para todas las series representadas.

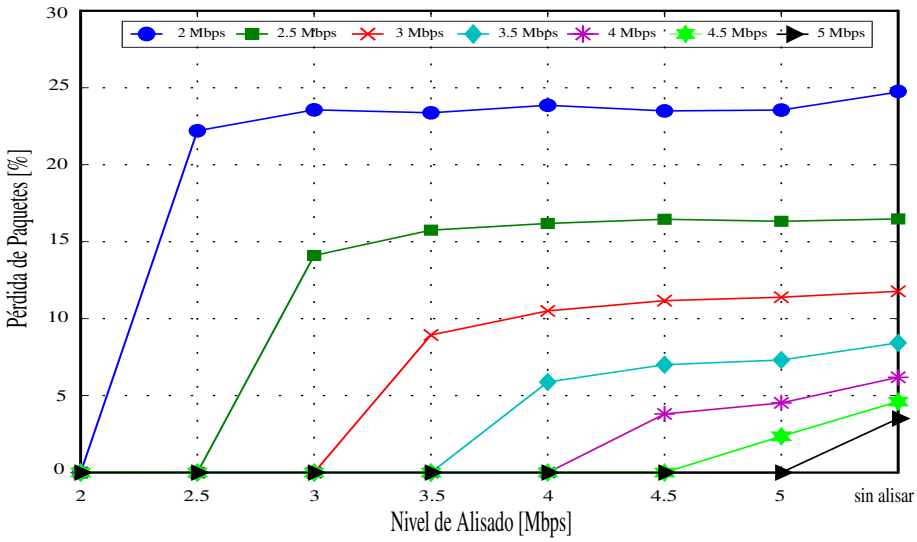

Figura 4.23: Pérdida de paquetes P2P-TV en número de paquetes.

La Figura 4.23 muestra el porcentaje de pérdida de paquetes con un comportamiento similar al que se muestra en la Figura 4.22. La pérdida de paquetes disminuye a medida que los umbrales de alisado disminuyen. Cuando los niveles de alisado son iguales o inferiores a la capacidad del enlace, no hay pérdida de paquetes.

Al mismo tiempo, se puede observar que el porcentaje de pérdida de paquetes es ligeramente menor que el porcentaje de pérdida de *bytes* en todos los casos: en la Figura 4.23 se puede ver que la pérdida de paquetes aumenta abruptamente cuando los niveles de alisado están cerca de la capacidad del enlace. Sin embargo, hay un punto en el que el porcentaje de pérdida de paquetes varía muy poco con el aumento de los umbrales de alisado. Además, en la Figura 4.22 se puede observar que el porcentaje de pérdida de *bytes* aumenta sin cambios abruptos, por lo que se puede concluir que los paquetes pequeños son más susceptibles de ser descartados para niveles de alisado cercanos a la capacidad del enlace; sin embargo, para niveles más altos de alisado, los paquetes grandes son más susceptibles de ser des-

cartados. Además, se puede observar que no hay pérdida de paquetes para niveles de alisado inferiores o iguales a la capacidad de enlace utilizada, pero se agregan valores muy altos de retardo (como se explica en la sección 3.4.2), que no pueden tolerar los usuarios de aplicaciones P2P-TV u otros servicios en tiempo real.

La Figura 4.24 muestra el *throughput* alcanzado por la aplicación P2P-TV usando un enlace ascendente de 3 $Mbps$. En el eje *"X"*, se muestran los niveles de alisado utilizados y la traza sin alisar; en el eje *"Y"*, se presenta el *throughput* (en $Mbps$) alcanzado y la capacidad del enlace que queda disponible. Los resultados muestran que los recursos de red podrían optimizarse si se aplican técnicas de alisado de tráfico, logrando así un mayor *throughput* de la aplicación bajo las mismas condiciones de red.

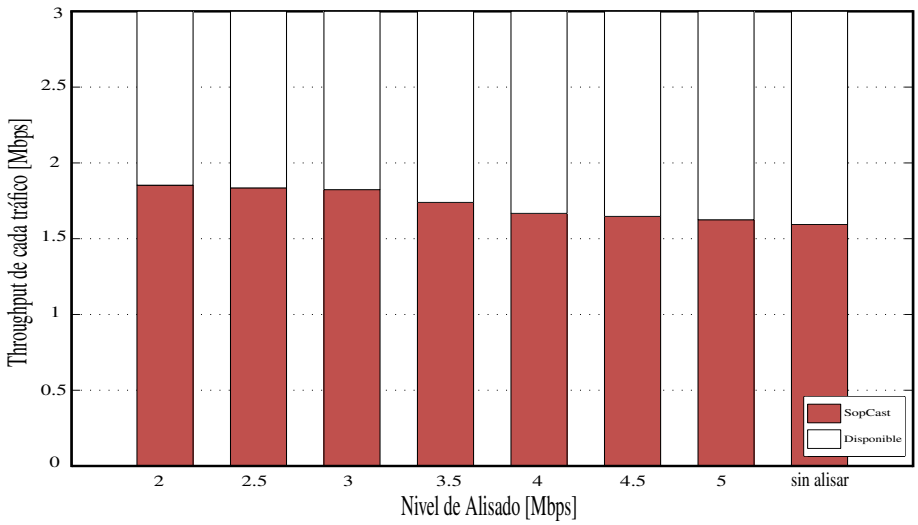

Figura 4.24: *Throughput* alcanzado por P2P-TV para una capacidad del enlace de 3 $Mbps$.

Aunque el enlace tiene suficiente capacidad, la aplicación no puede alcanzar el *throughput* necesario (1,84 $Mbps$) porque el tráfico a ráfagas satura el *buffer* de acceso y causa la pérdida de paquetes. Como se esperaba, si el umbral de alisado disminuye, la ocupación del enlace alcanzada por la aplicación aumenta. Además, el *throughput* logrado por la traza sin alisar es de aproximadamente 1,55 $Mbps$; sin

embargo, si se utiliza un umbral de alisado de 3 $Mbps$, se puede alcanzar alrededor de 1,84 $Mbps$. Se puede concluir que en este caso las técnicas de alisado mejoran la utilización del enlace.

4.2.2.3 **Resultados de la Prueba 2**

En esta prueba, se analiza cómo influye un tráfico de fondo (el cual no se alisa) en la aplicación P2P-TV (original y alisada) cuando comparten el enlace de subida. El tráfico de fondo seleccionado es un servicio FTP que utiliza el protocolo TCP-SACK para el transporte. En la Figura 4.25 y 4.26 se muestran las tasas de pérdida de *bytes* y de paquetes cuando un cliente P2P-TV y un servicio FTP comparten el mismo enlace de subida. De forma similar a las Figura 4.22 y Figura 4.23, cada serie corresponde a diferentes capacidades de enlace. Como se esperaba, la tasa de pérdida disminuye mientras más se alisa la traza. En este escenario, el servicio P2P-TV siempre presenta algunos niveles de pérdida, a diferencia de lo que se muestra en la Figura 4.22.

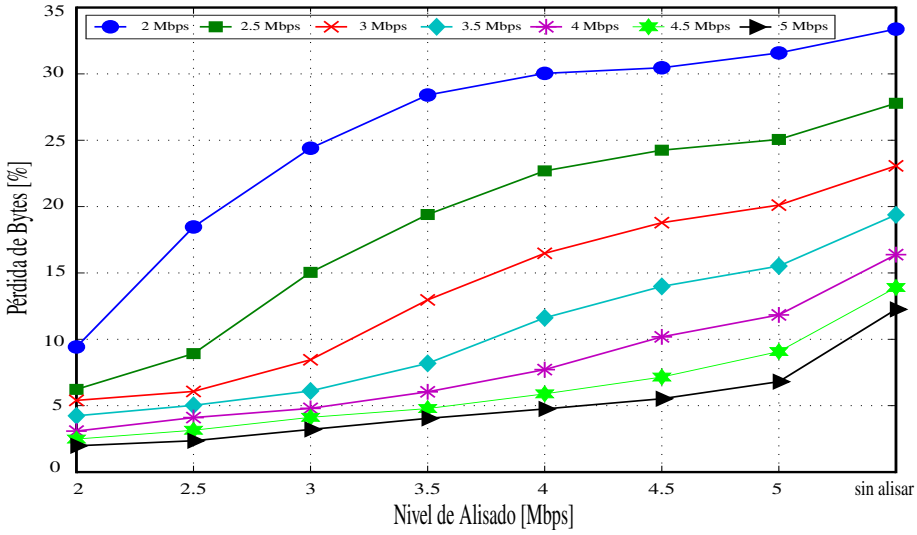

Figura 4.25: Pérdida de P2P-TV en *bytes* al compartir el enlace con un servicio FTP.

Como ejemplo se puede mencionar, que para una capacidad de enlace de 3,5 $Mbps$ en la Figura 4.25, y utilizando la traza sin alisar, se obtiene una tasa

de pérdida de *bytes* cercana al 20 %, que podría reducirse a un 8 % cuando se utilizan un nivel de alisado de 3,5 $Mbps$ y menos del 5 % para un nivel de alisado de 2 $Mbps$. Los niveles de pérdida de *bytes* en SopCast, independientemente del nivel de alisado y la capacidad de enlace utilizada, se deben a la combinación de flujos FTP y P2P-TV que llenan el *buffer* en el Nodo 0. Por lo tanto, la probabilidad de pérdida de *bytes* aumenta ligeramente cuando se comparan datos de la Figura 4.25 y 4.22.

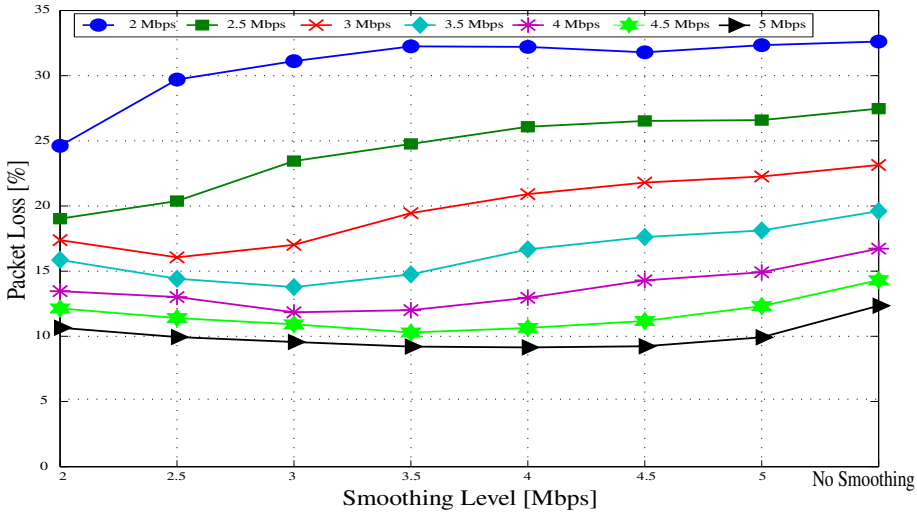

Figura 4.26: Pérdida de P2P-TV en número de paquetes al compartir el enlace con un servicio FTP.

La Figura 4.26 muestra el porcentaje de pérdida de paquetes para el flujo P2P-TV. La pérdida de paquetes disminuye a medida que aumenta el nivel de alisado, como sucedió en la Figura 4.23. La diferencia es que, en este caso, aparece un ligero aumento en la pérdida de paquetes para niveles más altos de alisado, lo que podría indicar que los paquetes pequeños tienen una mayor probabilidad de pérdida.

Las Figura 4.27 y Figura 4.28 muestran el comportamiento del *throughput* alcanzado por los flujos P2P-TV y FTP (respectivamente) para cada nivel de alisado de SopCast, cuando ambos servicios comparten el enlace de subida. En el eje X se

representa los diferentes niveles de alisado aplicados a SopCast así como también la traza de SopCast sin alisar. Analizando el caso del enlace de 2 $Mbps$, y utilizando el nivel de alisado más alto (2 $Mbps$), SopCast alcanza 1,65 $Mbps$ (Figura 4.27) que es un *throughput* mayor que el de la traza sin alisar, que solo alcanza 1,2 $Mbps$, es decir, la utilización de recursos de red para este tráfico se mejora en aproximadamente 450 Kbps por cada usuario, lo que representa más de un 35 %. Por otro lado, el tráfico de fondo FTP (Figura 4.28) disminuye su *throughput* en 400 $Kbps$ (cerca de un 33 %) cuando comparte el enlace con la traza de SopCast alisada a 2 $Mbps$. Por lo tanto, se podría decir que el FTP ha perdido el *throughput* que ha sido ganado por el servicio P2P-TV.

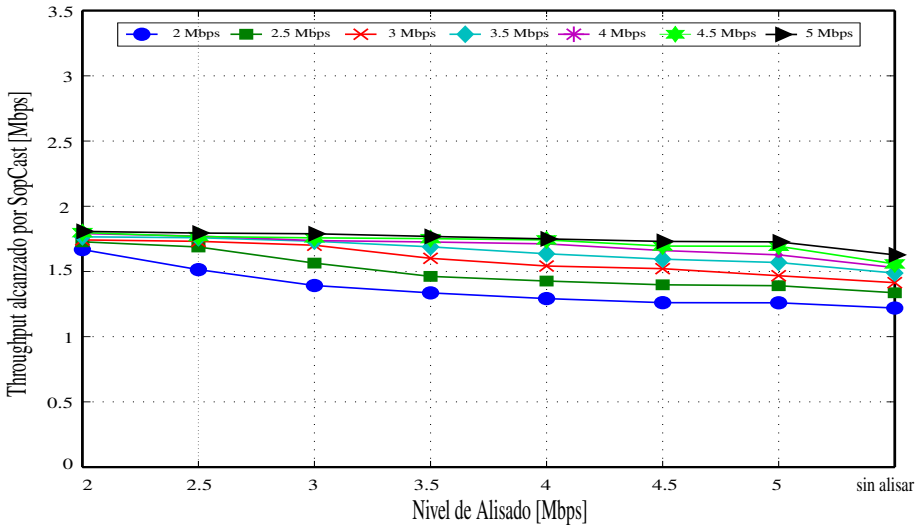

Figura 4.27: *Throughput* alcanzado por P2P-TV para cada nivel de alisado de SopCast.

Al igual que en la en la Prueba 1, se ha seleccionado el enlace de 3 $Mbps$ para representar el *throughput* logrado por P2P-TV y el servicio FTP, ya que presentan un comportamiento similar para todas las capacidades de enlace utilizadas. La Figura 4.29 muestra la distribución del *throughput* logrado por cada uno de los tráficos para cada nivel de alisado de la aplicación P2P-TV.

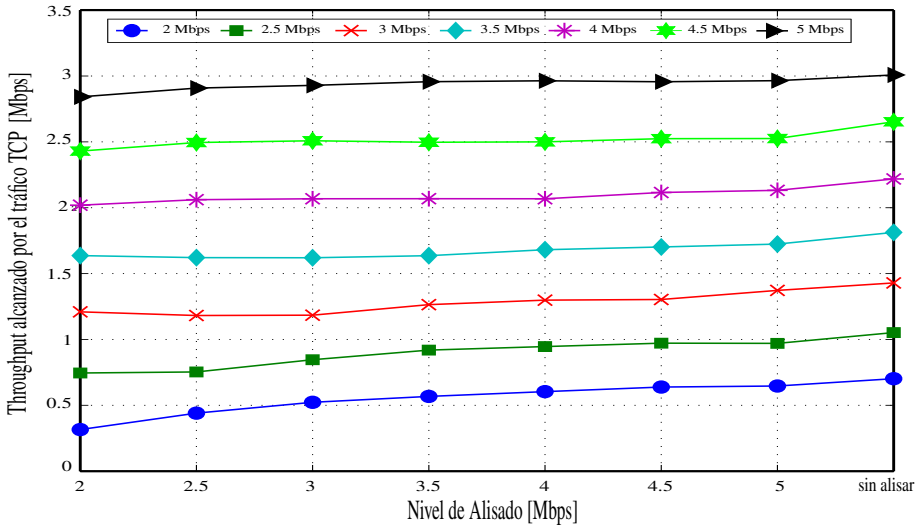

Figura 4.28: *Throughput* alcanzado por FTP para cada nivel de alisado de SopCast.

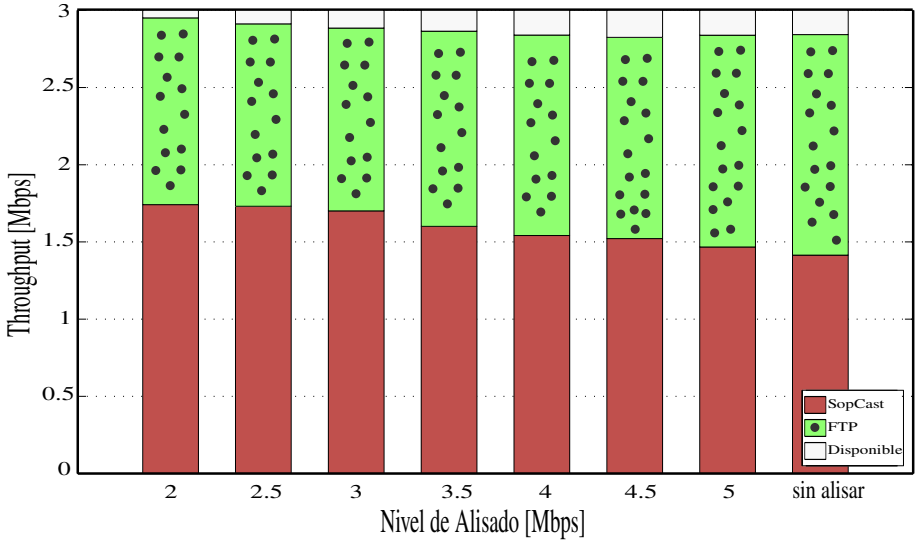

Figura 4.29: *Throughput* alcanzado por el tráfico de SopCast (alisado) y por el FTP (no alisado), para cada nivel de alisado de SopCast en un enlace con capacidad de 3 $Mbps$.

Con el objetivo de determinar los niveles de alisado más apropiados para aplicar al flujo P2P-TV (UDP), las pruebas se repiten para todos los niveles de alisado aplicados a la traza de SopCast. En teoría, la aplicación P2P-TV debería lograr su máximo *throughput* y el resto de la capacidad disponible debería ser utilizado por el servicio FTP. Sin embargo, los resultados representados en la Figura 4.29 muestran que SopCast no alcanza el mismo *throughput* en todos los casos, y el FTP no ocupa toda la capacidad del enlace disponible.

También se puede ver que cuando el nivel de alisado es menor o igual que la capacidad del enlace, el *throughput* de la aplicación P2P-TV mejora. Por ejemplo, en el caso de la traza sin alisar, SopCast alcanza aproximadamente $1{,}45\ Mbps$ y el FTP cerca de $1{,}40\ Mbps$, dejando inutilizado $0{,}15\ Mbps$. Por otro lado, si se considera el nivel de alisado más alto ($2\ Mbps$), el *throughput* alcanzado por P2P-TV es de $1{,}75\ Mbps$, el mayor valor de *throughput* obtenido por SopCast en esta prueba.

Como se ha estudiado en [30], en función de la implementación del protocolo TCP, se conseguirá una mayor o menor ocupación de la capacidad del enlace. Los valores de *throughput* alcanzados por el servicio FTP podrían ser diferentes según la variante de TCP que se utilice (SACK, FACK, Reno, New Reno, Vegas, entre otras), dependiendo de si la implementación de TCP es más o menos drástica a la hora de dejar de transmitir tráfico cuando detecta congestión. Otras implementaciones como New Reno y SACK utilizan la información de pérdida de paquetes para reducir su tasa de envío. Por otro lado, Vegas utiliza el aumento del *Round-Trip Time* (RTT) para prevenir la pérdida de paquetes.

Sólo puedes analizar los datos que tienes. Sé estratégico sobre qué reunir y cómo almacenarlo.

Marie Curie

CAPÍTULO

5

Integración y gestión de servicios

En este capítulo, se presenta una solución de orquestación de servicios de red y la puesta en marcha de una plataforma de virtualización para una red $5G$. Además, se describen dos escenarios que han sido implementados en aras de corroborar la viabilidad y funcionalidad de esta plataforma: *(1)* un concierto en tiempo real entre artistas localizados en diferentes ciudades europeas y *(2)* una prueba de concepto para un sistema de videovigilancia en sistemas de transporte público utilizando dispositivos IoVT. Se ha tenido en cuenta que dicha plataforma consiste en un sistema extremo a extremo que se compone por varios elementos habilitados para $5G$, pero que al mismo tiempo integra arquitecturas de sistemas heredados. De ahí que se deban tener en cuenta las limitaciones en las redes de acceso y considerar diferentes técnicas de optimización del tráfico para mejorar la QoS.

Primeramente, se describe los componentes desarrollados como parte de la plataforma de gestión y orquestación propuesta, incluyendo la infraestructura física de red desplegada, los elementos de *software* desarrollados para garantizar la correcta orquestación de servicios utilizando OSM, la seguridad de las diferentes tecnologías de nube empleadas (específicamente, *OpenStack* y el CEE del proveedor) y la monitorización del estado de la red a través del uso de varios *plugin*.

Por último, se detalla una prueba de concepto de un sistema de videovigilancia para transporte público, fundamentada en la arquitectura de gestión y orquestación para $5G$ expuesta. El caso de uso propuesto se aborda con dos enfoques diferentes, por un lado se realiza una implementación real de dispositivos IoT y un proveedor

de nube privado, y por otro lado, se realizan simulaciones con el tráfico proveniente de una aplicación P2P-TV (la misma que ha sido objeto de estudio en capítulos anteriores), cuando comparte el mismo enlace de acceso con un tráfico de fondo basado en diferentes implementaciones del protocolo de transporte TCP. Además, se aplica una técnica de conformado de tráfico con el fin de hacer un estudio de algunos parámetros de QoS, lo cual permitirá evaluar el rendimiento de la red. Concretamente, se analiza el retardo introducido a los paquetes del tráfico P2P-TV y la pérdida de paquetes tanto para el *video streaming* como para el tráfico TCP utilizado de fondo; y también la influencia entre ambos tráficos en el comportamiento de las aplicaciones.

5.1 Orquestación en sistemas 5G

En el capítulo 3 se detallaron los elementos de arquitectura de alto nivel que componen la plataforma de gestión y orquestación de extremo a extremo para $5G$ propuesta en este trabajo, Figura 3.1, en la cual se integran elementos de sistemas $5G$ así como sistemas heredados ($4G$, Wi-Fi, entre otros). Uno de los aspectos más importantes abordados ha sido la orquestación de servicios en la nube, ya que habilita el carácter inteligente de la plataforma propuesta, y al mismo tiempo garantizar una gestión dinámica de aplicaciones dentro de un conjunto de nodos IoT que forman parte de un entorno *Edge Cloud*.

La plataforma propuesta integra diversas nubes, específicamente una nube privada utilizando *OpenStack* y el CEE de un fabricante de de redes móviles. Esto conlleva a la implementación de un sistema heterogéneo en cuanto a proveedores, tecnologías, infraestructuras y servicios. Es por ello que, a continuación se presenta la implementación de la orquestación para múltiples proveedores ($4G$/$5G$), los mecanismos para garantizar la seguridad, monitorizar, controlar y gestionar la infraestructura de red y los recursos de la red a través de diferentes *plugin*, así como también el montaje del equipamiento físico que da soporte a esta plataforma.

5.1.1 Orquestación de múltiples VIMs

Para que la orquestación de las diferentes VIMs ocurra de manera escalable y flexible, se ha de garantizar que las variadas tecnologías de nubes puedan coexistir dentro de un mismo sistema de gestión. En la implementación de red utilizada, los sistemas integrados son una combinación de la tecnología de nube desplegada en el *King's College London* (KCL) (utilizando *OpenStack*) y diversos sistemas propietarios, como por ejemplo el CEE (un sistema precomercial $5G$ de Ericsson). Dicho CEE es un entorno similar a *OpenStack* que añade un conjunto de funcionalidades que le permite cumplir con tareas específicas de servicios móviles. Por lo tanto, para integrar con éxito múltiples nubes en una plataforma de orquestación común, se requiere implementar las APIs necesarias que posibiliten la comunicación entre diferentes VIMs. Sobre la base del conjunto de *plugin* proporcionado por OSM, se pueden orquestar y administrar múltiples VIMs, siempre y cuando los *plugin* apropiados sean creados e integrados dentro del orquestador de recursos. OSM admite las siguientes VIMs: *OpenVIM*, *OpenStack*, *VMware vCloud Director*, *VMware Integrated OpenStack* y AWS.

Para el *OpenStack* implementado en el campus de KCL se utilizaron mecanismos de interfaz ya integrados en OSM. Sin embargo y como se mencionó en capítulos anteriores, para el CEE se diseñó un nuevo *plugin* que garantiza el acceso solamente a un pequeño subconjunto de APIs, debido a los requerimientos de seguridad exigidos por las aplicaciones y sistemas propietarios en una arquitectura de orquestación compartida con varios proveedores. Los *plugin* de OSM permiten, en este caso, personalizar cómo se accederá desde el orquestador al sistema propietario. En particular, con las restricciones de acceso se impide la conectividad directa entre OSM y el CEE. Este enfoque ofrece la flexibilidad de manipular múltiples *OpenStack* con diferentes niveles de autorización (o permisos de acceso) y un subconjunto diferente de VNFs y funciones desplegadas por la VIM. También muestra cómo las APIs de comunicación se puede personalizar en OSM y cómo es posible integrar nuevas APIs para interactuar con otras plataformas que actualmente no son compatibles.

El *plugin* de la VIM del sistema propietario se basa en el *plugin* genérico OSM-*OpenStack*. Se requirieron varias modificaciones al mismo, puesto que el proceso de autenticación en el CEE difiere del proceso de autenticación estándar entre OSM y *OpenStack*. Para lograr una autenticación eficiente entre estos dos sistemas, varios componentes de *software* del orquestador de recursos tuvieron que modificarse para pasar correctamente las solicitudes no autenticadas al CEE. Con este fin, y para poder orquestar sin dificultades en los restantes subsistemas, se implementaron las APIs correspondientes y los componentes de integración necesarios dentro de OSM y de esta manera, poder agregar exitosamente dicha VIM.

El desarrollo de *plugin* específicos para una VIM genera inherentemente un entorno de orquestación compartimentado. Esta compartimentación proporciona a los administradores la capacidad de adaptarse específicamente a las funcionalidades de las VIMs (en este caso, *OpenStack* y CEE) según sea necesario. Dicha segmentación también refuerza la seguridad dentro de la arquitectura, ya que los usuarios finales sólo tienen acceso a funciones específicas, a discreción del desarrollador del *plugin* y del administrador del sistema. En consecuencia, la arquitectura general es más escalable, ya que se pueden agregar funciones adicionales a cada *plugin* según sea necesario.

Para preservar la integridad del ecosistema presentado, los sistemas $4G$ y $5G$ fueron modificados para poder implementarlos e integrarlos en el *plugin* de la VIM. De esta manera la capa superior de la orquestación se mantuvo al margen de los detalles de dicha implementación, sin verse afectada la orquestación de los subsistemas no heterogéneos.

Para la orquestación de diferentes VNFs, OSM requiere de una interfaz de comunicación con cada VIM, de tal manera que se permita la implementación de los VNFs. Opcionalmente, se necesitará implementar operaciones primitivas de configuración en los VNFs para cada servicio de red, mediante una conexión directa a la interfaz de administración de los mismos. Además, un conjunto de políticas de administración en la VIM asegura la adecuada asignación de recursos computacionales en los casos en los que OSM no ofrezca las opciones de configuración en sus APIs. Cada *plugin* en la VIM expone una interfaz norte (*Northbound Interface* (NBI)) definiendo la disponibilidad de la red, la capacidad de almacenamiento y

de cómputo, así como el uso dado por cada *tenant* y la información de facturación. La interfaz en dirección sur (*Southbound Interface* (SBI)) de la VIM recopila y clasifica esta información en un catálogo de servicios, detallando los servicios de red disponibles. Cada entrada de este catálogo (también se conoce como *Virtual Network Function Descriptor* (VNFD)) es la composición de un conjunto de elementos de información que definen el tiempo de vida, los *Central Processing Unit* (CPU)s asignados y los requisitos de almacenamiento para el proceso de creación de instancias.

5.1.2 Fundamentos para una orquestación inteligente

Una de las contribuciones más importantes en la comunidad MANO es la capacidad de monitorizar en tiempo real la infraestructura y los recursos de la red. La monitorización permite el uso eficiente de los recursos, tanto en la infraestructura física a cargo de los recursos y almacenamiento de los servicios de red así como en la red de transporte subyacente [91], que a menudo se le asignan recursos en exceso para evitar la degradación del rendimiento en caso de congestión.

Las herramientas de monitorización en OSM se encuentran en una etapa de desarrollo temprana y sólo recuperan información básica de cada VIM [92]. Sin embargo, su naturaleza modular permite integrar mecanismos personalizados (Figura 5.1). Como se ha explicado en la sección 3.1, el *plugin* desarrollado para comunicar el *OpenStack* con el CEE, se conecta a OSM a través del controlador personalizado *OpenDaylight*, el cual es capaz de medir la latencia, el *jitter*, la pérdida de paquetes, el *throughput* y el ancho de banda en la red. Esto se logra mediante el uso de un servidor apto para solicitar y almacenar datos de red provenientes de diversas fuentes, incluidas *OpenDaylight* y *OpenStack*; sin embargo, cualquier otra fuente de datos se puede agregar fácilmente por medio de una REST API. También es posible que otras aplicaciones distintas a OSM puedan utilizar estos datos y se sustenten en ellos.

La ampliación y la seguridad son puntos clave en el diseño de un sistema de monitorización, ya que no se limita solamente a *OpenDaylight* y/u *OpenStack*,

sino que también debe ofrecer la posibilidad de conectar cualquier otro controlador SDN y/o cualquier otra VIM.

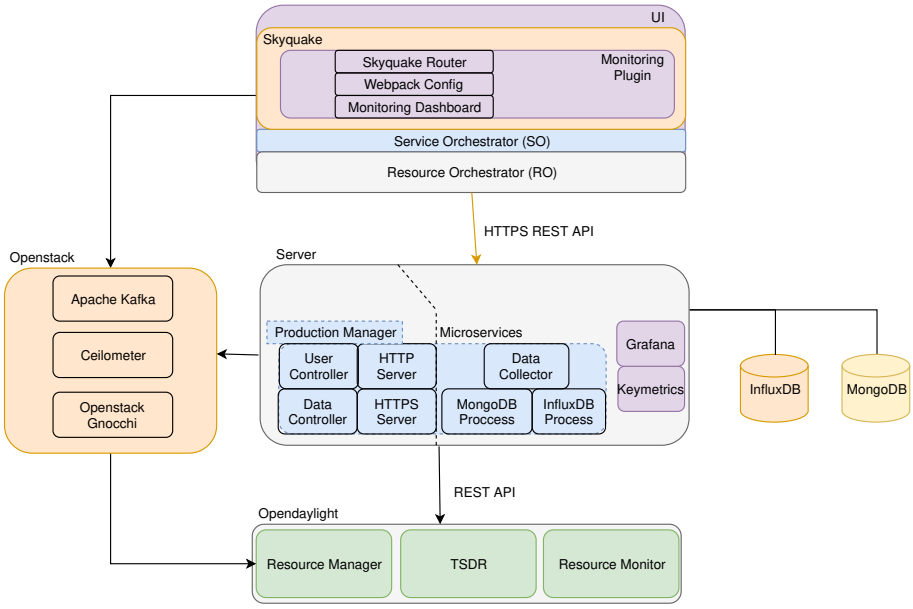

Figura 5.1: Conjunto de *plugin* utilizados para monitorizar la red por medio de un controlador SDN.

En la Figura 5.1 se muestra el conjunto de *plugin* utilizados para monitorizar la red por medio de un controlador SDN, facilitando una orquestación inteligente para la plataforma $5G$ presentada, basándose en parámetros de red. Cada aplicación externa al registrarse, recibe un *token* con un tiempo limitado, el cual debe pasarse en cada llamada para la autenticación. El servidor puede verse como el punto central de agregación de datos que ejecuta muchos microservicios para recopilar datos de red de muchos lugares y almacenarlos en una serie temporal utilizando *InfluxDB*. Además, permite que otras aplicaciones utilicen los datos a través de servicios HTTP y HTTPS, así como el almacenamiento de la información del usuario en *MongoDB* para facilitar la separación entre los datos de gestión y los datos de la red. Por último, el *Production Manager* es el responsable de monitorizar los microservicios y reiniciarlos en caso de fallo, manteniendo un registro

de todos los eventos. Al mismo tiempo, todo esto también se integra con la UI de *Keymetrics* que muestra el estado físico de los servidores así como el estado de los microservicios. Para monitorizar los datos de forma gráfica, se utiliza *Grafana*, una herramienta para generar gráficos en tiempo real directamente desde *InfluxDB*.

La inclusión de los módulos antes mencionados permite que las políticas de QoS, como la priorización y el aislamiento de recursos a través de la segmentación virtual de la red física, se apliquen a través del administrador de recursos en el controlador SDN. Las herramientas de desarrollo en OSM incluyen mecanismos para expandir la integración SDN y permitir la configuración automática de reglas de tráfico y reenvío de flujos, de acuerdo con los requisitos del servicio. De esta manera, la orquestación garantizará una correcta coordinación de políticas y reglas de QoS, tanto en la infraestructura de nube y como en la red física.

5.1.3 **Diseño para IoT y tecnologías de nube**

Para demostrar el verdadero potencial de las nuevas tecnologías emergentes, se ha implementado la arquitectura para una red $5G$ detallada en la sección 3.1; dicho sistema está integrado por tecnologías de radio $5G$ (en este caso, pre-comerciales), $4G$ y Wi-Fi. El sistema $5G$ se basa en dos sistemas *Massive Multiple-Input Multiple-Output (MIMO)* a 3,5 GHz y un sistema *millimeter-wave* (mmWave) a 28 GHz. Las unidades de radio están conectadas a unidades de banda base de $5G$, equipamiento de red de $5G$, una *virtualized RAN* (vRAN) de $4G$ y un *virtualized Core* (vCore). Se han interconectado siete plataformas, como se ilustra en la Figura 5.2. La primera de ellas se corresponde a un conjunto de antenas ubicadas en la azotea del KCL, compuesta por los *Massive MIMO* a 3,5G GHz y el mmWave a 28 GHz antes mencionados. La segunda es la instalación de un $4G$ *E-UTRAN NodeB* (eNB) en un laboratorio de la misma universidad. La tercera plataforma son pico celdas comerciales de $4G$, utilizándose *dots* $4G$ pertenecientes al fabricante de telecomunicaciones con el cual se ha colaborado para el montaje de la red $5G$. La cuarta se basa en la infraestructura de *Software Defined Radio* (SDR) desplegada por el KCL, mientras que la quinta hace uso de puntos de acceso Wi-Fi. La sexta es un

sistema al aire libre desplegado en un espacio público en el centro de Londres y, finalmente, la séptima es un escenario $5G$ en vivo que conecta al KCL con un centro de arte y ceremonias perteneciente al Ayuntamiento de Londres, conocido como GuildHall.

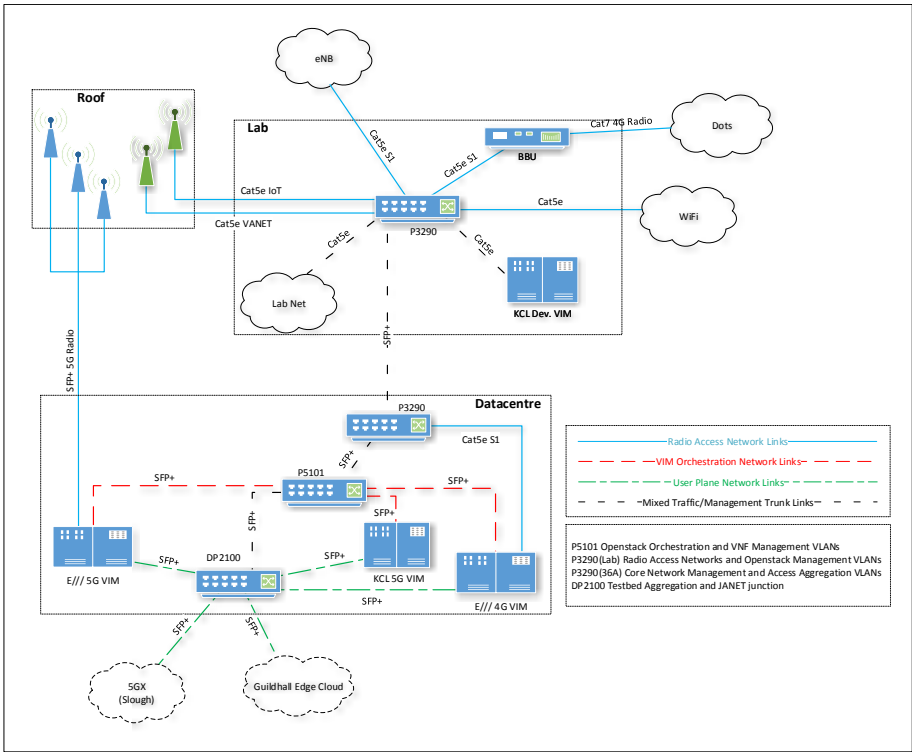

Figura 5.2: Sistema de integración para la red 5G en el KCL.

Con esta implementación, se pretende probar cada una de las plataformas de manera independiente según las KPIs de rendimiento de $5G$ definidas por la *Next Generation Mobile Networks* (NGMN) y probar las capacidades de comunicación de un extremo a otro en los diferentes sitios de prueba mediante la exploración de nuevas soluciones para el almacenamiento en el borde de la red, así como también la orquestación de servicios en tiempo real basada en las condiciones de la red, los requisitos de las aplicaciones y la movilidad del usuario.

Con el fin de proporcionar un entorno más flexible y dinámico, y ofrecer más

opciones para la experimentación, es necesario garantizar la existencia de un conjunto de características en la implementación final: *(a)* soporte de aplicaciones de baja latencia, *(b)* funcionalidades de *pass-through*, *(c)* dispositivos *hot-plug* y *(d)* migración de VNFs.

A continuación se explica detalladamente cómo se ha implementado la plataforma KCL-*OpenStack* (representada como KCL $5G$ VIM) y sus características. Sin embargo, en la Figura 5.2 se muestra además que existen un total de seis sistemas *OpenStack* en la infraestructura $5G$ presentada: uno que aloja la implementación KCL-*OpenAPI* (OAI) y otros servicios (KCL $5G$ VIM), otro para una plataforma VNF de prueba (KCL Dev. VIM), otro tercero para un sistema $4G$ (E/// $4G$ VIM) y los restantes tres para el *core* y las redes de acceso de radio ($3{,}5\ GHz$ y $28\ GHz$) del sistema $5G$ (E/// $5G$ VIM).

La implementación de KCL-*OpenStack* consta de ocho nodos de cómputo, de los cuales uno actúa como controlador y siete están dedicados a cómputo y almacenamiento. Los nodos se dividen en dos grupos: hipervisores de *Virtual Machine* (VM)s y basados en contenedores. Además, un nodo de cada grupo está ejecutando un kernel de baja latencia para aplicaciones con restricciones en el tiempo de ejecución, por ejemplo funciones $3GPP$. Los nodos están habilitados para la funcionalidad de *Single-Root I/O Virtualization* (SRIOV), lo que permite que las VNFs pasen directamente a una interfaz de red física con el fin de mejorar su rendimiento. La capacidad de migración de VNFs entre los nodos en el borde se garantiza a través de un bloque de almacenamiento de red dedicado a $10\ Gbps$, donde cada nodo de cómputo actúa como miembro del grupo de almacenamiento.

El despliegue de la red expone los VNFs a la infraestructura física por medio de enlaces a $10\ Gbps$ dedicados para la red del proveedor y SRIOV. En este contexto, el componente *Neutron* de *OpenStack* gestiona todo lo relacionado con la implementación de las redes de los VNFs dentro del propio *OpenStack*, mientras que *OpenDaylight* gestiona todo lo relacionado con el enrutamiento entre los diferentes *testbed* externos a *OpenStack*. *Open vSwitch* (OvS) se usa en todos los nodos como puente principal que conecta los VNFs a las redes físicas cuando no se usa SRIOV. Para lograr un mayor rendimiento de la red y una mejor eficiencia en

el uso de recursos, es también posible utilizar OvS en conjunto con el *Data Plane Development Kit* (DPDK) de Linux.

Por otro lado, *OpenDaylight* se utiliza en el *switch* de red proporcionado por el fabricante, gestionando una plataforma flexible de conmutación y enrutamiento ($DP2100$ en la Figura 5.2), donde todos los VNFs están conectados a la red física. Normalmente, para la integración entre OSM y *OpenDaylight*, el primero toma control total del controlador SDN e instala flujos estáticos para los VNFs cuando los mismos son instanciados, este escenario se conoce como *SDN-assist* y requiere que los VNFs usen SRIOV, pero se vuelve redundante en escenarios tipo *MAC-learning*. Sin embargo, en aras de proporcionar un entorno más flexible y dinámico, *SDN-assist* no se ha tenido en cuenta en la implementación del sistema propuesto. En su lugar, *OpenDaylight* se ejecuta de forma autónoma en el *switch* de capa 2 ($L2$) para ejecutar la función de *MAC-learning* y el reenvío de paquetes. Lo antes mencionado provee la capacidad de ejecutar funciones estándar de *MAC-learning* en la red del proveedor y permite conectar dispositivos *hot-plug* según se requiera sin la necesidad de implementar nuevamente los servicios de red con descriptores actualizados a través de OSM.

Uno de los principales objetivos del sistema $5G$ E2E presentado y desplegado como parte de este trabajo ha sido demostrar la viabilidad y utilidad de las tecnologías $5G$ en aplicaciones en tiempo real. Con este fin, se organizó un evento cultural, entre la Escuela de Música y Drama del Guildhall en Londres [93] y un músico al piano desde el Museo de la Puerta de Brandeburgo en Berlín, ofreciendo un concierto totalmente inmersivo, conectando artistas en diferentes localizaciones del planeta [94, 95, 96]. La latencia ultrabaja y el gran ancho de banda proporcionado por la plataforma $5G$ desplegada, han garantizado una sensación de inmediatez e inmersión durante esta experiencia cultural.

5.2 Integración de un servicio IoVT

A continuación se presenta la prueba de concepto propuesta en la sección 3.2, para un sistema de videovigilancia inteligente en vehículos de transporte público basado en dispositivos IoVT. La arquitectura para el VSS presentada (Figura 3.3)

se fundamenta en el sistema de gestión y orquestación de tecnologías $5G$ descrita en las secciones 3.1 y 5.1. El objetivo de esta implementación es proporcionar la gestión dinámica de aplicaciones dentro de un conjunto de nodos de borde ubicados en una flota de vehículos de transporte público, como pueden ser autobuses.

Por un lado, se realiza una implementación real de las aplicaciones o servicios utilizados en estos sistemas inteligentes de videovigilancia que serán gestionados como si se tratase de distintas implementaciones de VNFs (a modo de ejemplo). Por otro lado, se realizan simulaciones para obtener una medida de la eficiencia que se podría esperar al implementar técnicas de optimización de tráfico en este entorno, para ello se ha utilizado para las pruebas el tráfico de una aplicación P2P en tiempo real y un tráfico de fondo basado en TCP. En particular, se analizará del retardo introducido y la pérdida de paquetes, permitiendo realizar una evaluación del rendimiento esperado por la red.

En general, se puede decir que esta prueba de concepto aborda la necesidad de proveer videovigilancia inteligente en vehículos de transporte público de una manera eficiente en términos de congestión de red para dispositivos de bajas capacidades.

5.2.1 Implementación de la arquitectura

La Figura 5.3 presenta la implementación de la prueba de concepto de la arquitectura de un VSS inteligente para dos ejemplos de aplicaciones: *Video Streaming* y *Detección de Objetos*. La misma se centra fundamentalmente en la capa *Nodos Edge* y su interacción con la capa *Servicios en la Nube*, ya que las capas *Red de Acceso* y *Servicios en la Nube* son transparentes desde el punto de vista de diseño, y por lo general, son seleccionadas en función de estrategias comerciales.

Existen ciertos requerimientos mínimos para que un dispositivo pueda ser utilizado en la capa *Nodos Edge*, los cuales dependen del CSP utilizado, en general se puede resaltar: soportar un sistema operativo Linux, un procesador con al menos $1\ GHz$, $128\ MB$ de memoria y la potencia requerida por las aplicaciones que se desean utilizar. Para los nodos de cómputo se utilizaron dos *Raspberry Pi* y la

conexión a Internet se proporcionó a través de una interfaz de comunicación Wi-Fi estándar como *Red de Acceso*.

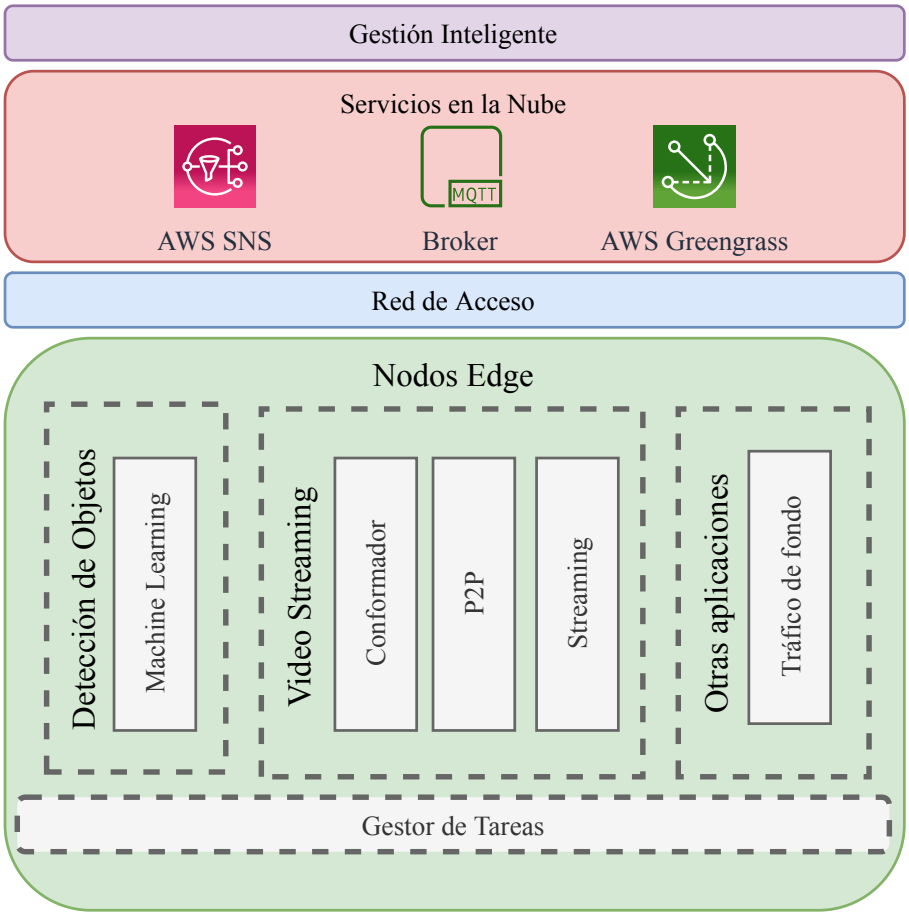

Figura 5.3: Implementación de la prueba de concepto para un VSS basada en dispositivos IoVT.

La capa *Gestión Inteligente* se basa en OSM, ya presentado y analizado en secciones anteriores. Se toma como punto de partida el sistema de gestión y orquestación de infraestructuras $5G$ para múltiples CSP, donde además se incluye una solución SDN basada en *OpenDaylight*, para gestionar de forma dinámica flujos de tráfico. Para ello se desarrollaron dos *plugin*, uno de los cuales posibilita la comunicación con la tecnología de nube privada de un fabricante de redes $5G$,

permitiendo la conexión con redes de este tipo. El segundo *plugin* se implementa en la interfaz gráfica de OSM con el fin de monitorizar, en tiempo real, la latencia, el *jitter*, el *throughput*, el ancho de banda y la pérdida de paquetes del estado de la red SDN y de los microservicios gestionados.

En la actualidad casi todos los CSPs ofrecen soporte para infraestructuras IoT, por lo que la selección óptima del mismo debería tener en cuenta, además de aspectos técnicos, estrategias de negocios y modelos de precios. En la capa *Servicios en la Nube*, se utilizó AWS como CSP, específicamente el servicio *Greengrass* que permite ejecutar contenedores y aplicaciones *serverless* en algunos dispositivos IoT y proporciona una comunicación segura entre el dispositivo y la nube. En la nube se implementó un *Broker* de mensajes basado en un modelo de publicación/suscripción para la entrega de mensajes, de modo que tanto los nodos como la capa de *Gestión Inteligente* puedan acceder a cualquier información enviada desde el *Edge Cloud*. Para este fin, se usó *Simple Notification Service* (SNS) de AWS al integrarse fácilmente con *Greengrass*.

Con el fin de probar el diseño de la arquitectura con un ejemplo de videovigilancia inteligente, se han implementado dos aplicaciones independientes (*Detección de Objetos* y *Video Streaming*) que pueden verse como una cadena de servicios. Por un lado, la *Detección de Objetos* es una aplicación *serverless* que captura imágenes de la cámara integrada en el dispositivo IoT y realiza tareas de inferencia *Machine Learning* (ML) utilizando MXNet. Se ha utilizado un modelo de ML preentrenado como ejemplo para facilitar la implementación de la prueba de concepto. No obstante, la eficiencia de dicho algoritmo está fuera del alcance de este trabajo.

Para proporcionar cierto nivel de privacidad en entornos de transporte público, las imágenes son capturadas y procesadas cada 1 segundo en los nodos y se eliminan una vez que se realiza la predicción correspondiente, de tal manera que no se envían imágenes a través de Internet ni se almacenan en la nube. Para cada predicción, la *Detección de Objetos* envía un mensaje al *Broker* a través de una conexión segura. Dicho mensaje contiene la lista de objetos detectados en formato *JavaScript Object Notation* (JSON). El *Broker* de mensajes es un servicio administrado en la nube, donde los mensajes con objetos detectados se almacenan y

comparten con la capa de *Gestión Inteligente*. Una vez que un objeto peligroso es detectado (por ejemplo, un arma), se ejecuta la aplicación *Video Streaming*. Esto permite hacer un uso eficiente del ancho de banda disponible, ya que se envía una pequeña cantidad de datos (un mensaje JSON) desde el dispositivo IoT, mientras que la transmisión de video se iniciará sólo en el caso que se detecte un objeto peligroso. A pesar de que un análisis de eficiencia energética no ha sido desarrollado, al utilizar este mecanismo se garantiza una reducción en el consumo de energía, ya que a diferencia de otras arquitecturas [97], el video se transmite sólo en caso de ser necesario. También, es posible ejecutar el *Video Streaming* bajo demanda en caso que sea requerido.

Por otro lado, la aplicación *Video Streaming* es una cadena de servicios que consta de tres componentes: *Streaming*, *P2P* y *Conformador*. En tiempo real, el *Streaming* usa la cámara y transmite el video, el cual es utilizado por el componente *P2P* para crear una red P2P con otros nodos y la capa de *Gestión Inteligente*, reduciéndose el tráfico enviado por cada nodo. Antes de enviar el tráfico a través de la *Red de Acceso*, el *Conformador* permite realizar una generación de paquetes más eficiente en cuanto a la utilización del ancho de banda, limitando las ráfagas y por tanto reduciendo la pérdida de paquetes, haciendo uso de algunas de las técnicas de conformado de tráfico discutidas en los capítulos 3 y 4. La aplicación *Video Streaming* se activa cuando se detecta cierto objeto y consiste en una cadena de contenedores: *VideoLAN Client* (VLC) se usa para transmitir el video desde la cámara y SopCast (aplicación P2P-TV estudiada a lo largo de este trabajo) se ejecuta como un cliente P2P. El *Conformador* no se ha implementado como parte de la prueba de concepto, en su lugar, se proporcionan resultados de simulación significativos utilizando NS [98].

Para dar una idea más clara, el objetivo del *Conformador* es minimizar el impacto causado por el tráfico a ráfagas en los *buffer* de los dispositivos de acceso y enviar a la capa *Gestión Inteligente* métricas de QoS para decidir qué parámetros optimizar en la generación de tráfico. Uno de los motivos por los cuales estas ráfagas se producen es debido a la concurrencia de diversos flujos de datos en la interfaz de los dispositivos de acceso. Los dispositivos IoVT producen diferentes

flujos de datos, por ejemplo video, datos de sensores y datos de control. En este caso en particular, se transmiten los datos de las aplicaciones que se ejecutan en los nodos (*Detección de Objetos*, *Video Streaming* y *Otras aplicaciones*) y la comunicación entre el *Gestor de Tareas* y *Servicios en la Nube*, donde existe una confluencia de tráfico que además, involucra diferentes protocolos de transporte. Por ejemplo, la comunicación entre el *Gestor de Tareas* y la nube se transmite sobre HTTPS utilizando TCP, mientras que el *Streaming* y el P2P pueden utilizar una combinación de TCP y/o UDP.

La función principal del *Conformador* en este caso de uso, es limitar los picos de *throughput* que a veces aparecen cuando se envía tráfico de red. El objetivo es evitar la pérdida de paquetes en el *buffer* durante los períodos de congestión utilizando un algoritmo óptimo de control de congestión TCP [30] y alisado de tráfico [99]. Este es un método de bajo consumo de recursos que un dispositivo IoT podría permitirse, a costa de introducir ciertos niveles de retardo, que sean tolerables por las aplicaciones. Estos retardos adicionales se han seleccionado para controlar la pérdida de paquetes causada por la llegada del tráfico en ráfagas y para mejorar la calidad de la comunicación. En este caso, la tasa de salida de cada paquete varía, pero el comportamiento promedio del tráfico sería el mismo.

5.2.2 Optimización del tráfico

Para las simulaciones se utiliza el tráfico de la captura de la aplicación P2P-TV estudiada en la sección 4.1. Además, se ha seleccionado un escenario representativo, donde un dispositivo IoT ejecuta la aplicación de transmisión de video en conjunto con otros servicios o aplicaciones TCP en segundo plano, lo cual sucedería cuando el nodo IoT envía información de control a la nube o datos de otros sensores. En la Figura 5.4 se representa el escenario de simulación en el que una aplicación de *video streaming* y un servicio FTP comparten el mismo enlace de acceso a Internet, muy similar al escenario presentado en la Figura 4.21. La capacidad del enlace de acceso varía entre 2 y 5 $Mbps$ y al igual que en el capítulo anterior, las capacidades de enlace inferiores a 1,84 $Mbps$ (que se corresponde

con el *throughput* promedio de la traza de la aplicación P2P-TV empleada) no se han tenido en cuenta para evitar el deterioro del rendimiento del *video streaming*.

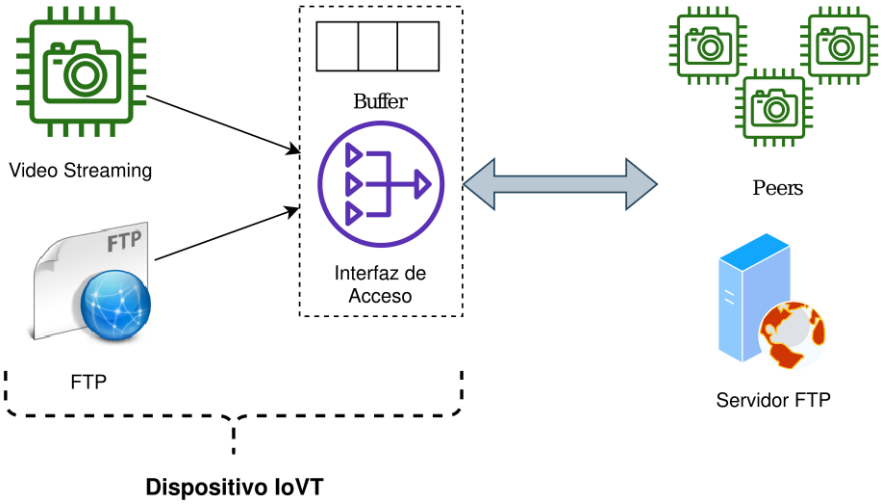

Figura 5.4: Escenario de simulación utilizado para las pruebas, en el cual se transmiten dos tipos de tráfico diferentes.

Para mostrar un ejemplo en concreto, se han realizado las simulaciones utilizando el algoritmo de alisado presentado en la sección 3.4, como técnica de conformado del tráfico. De igual manera que las simulaciones realizadas previamente, se han seleccionado múltiples niveles de alisado, que varían de 2 a 5 $Mbps$, evitándose así congestionar los *buffer*, y por lo tanto, la pérdida de paquetes. Niveles de alisado inferiores al *throughput* medio (1,84 $Mbps$) no se deben usar para evitar el deterioro de la aplicación. Se ha comprobado que para este flujo de datos en particular, si se alisa por encima de 5 $Mbps$, el tráfico resultante presenta un comportamiento en términos de ráfagas y tiempos entre paquetes muy similares a cuando se alisa a 5 $Mbps$.

Las pruebas se repetirán para seis distintas capacidades de enlace, para cada nivel de alisado y para seis algoritmos diferentes de control de congestión (*Sack*, *Reno*, *Fack*, *Linux*, *Vegas* y *New Reno*) utilizados por el servicio TCP. Además, cada valor mostrado corresponde a la media de 100 iteraciones de cada simulación.

En cada caso, se obtiene la pérdida de paquetes para las dos aplicaciones (P2P y FTP). Con la finalidad de realizar las simulaciones en condiciones similares a las pruebas del capítulo 4, se utilizan los parámetros de *buffer* descritos en [100] en una conexión Wi-Fi, ya que son valores ampliamente utilizados en dispositivos comerciales: en el enlace ascendente se implementa un *buffer* de 50 paquetes con una política de gestión de tipo *drop-tail*, mientras que el enlace descendente está dimensionado con un *buffer* de 500 paquetes.

En el capítulo 3 se ha hecho alusión a que las técnicas de conformado de tráfico introducen ciertos niveles de retardo, por lo tanto, se debe encontrar un balance entre los retardos añadidos y los niveles de alisado. Este nivel de retardo introducido dependerá de las características del tráfico de la aplicación y su tolerancia al mismo; en este caso, la aplicación P2P-TV utilizada, puede tolerar un retardo de hasta 60 segundos, según [43]. Si se llegara a utilizar otra aplicación, dicha tolerancia al retardo debería verificarse ya que podría impactar el rendimiento y la QoS de diferentes maneras.

Con el objetivo de determinar el umbral de alisado apropiado para este tipo de tráfico, se obtiene el retardo añadido a cada paquete para cada nivel de alisado utilizado (de 2 a 5 $Mbps$). La Figura 5.5 presenta el porcentaje de paquetes a los cuales se les ha añadido un determinado retardo. Cada barra representa un nivel de alisado y se han agrupado los paquetes dentro de ciertos umbrales para su posterior análisis. Para el nivel más alto de alisado (2 $Mbps$), se introducen retardos intolerables para los servicios en tiempo real, por encima de los (10 segundos). Esto se debe a que el nivel de alisado está muy cerca del *throughput* medio que necesita la aplicación utilizada ($1{,}84$ $Mbps$). Sin embargo, a medida que disminuye el umbral de alisado (hacia valores cercanos a 5 $Mbps$), el valor de los retardos añadidos también disminuyen. Se puede observar una cantidad significativa de paquetes que no cambian su tiempo de envío para todos los niveles de alisado (0 segundos). Los retardos añadidos ocurren cuando los paquetes llegan en ráfagas. Siempre existe una disparidad entre el nivel de alisado, los retardos añadidos y la pérdida de paquetes. Por ejemplo, el retardo máximo introducido (y sólo en algunos paquetes) en la traza alisada a 3 $Mbps$, no supera los 20 segundos (tolerable por SopCast) y aproximadamente el $18{,}3\,\%$ de los paquetes no se ven afectados.

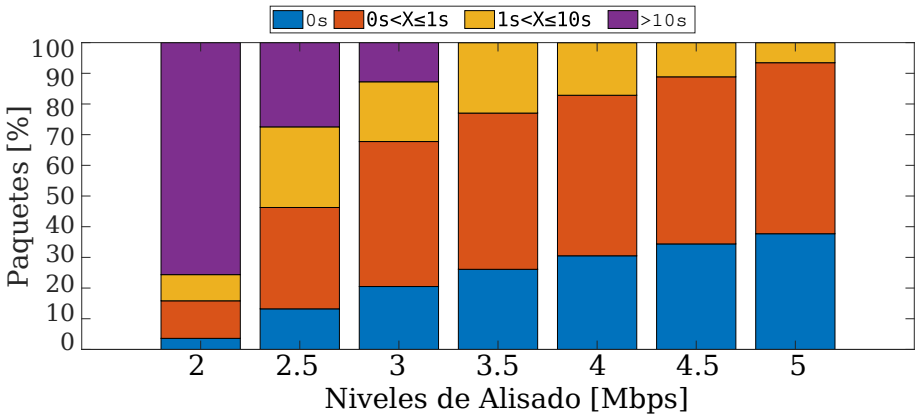

Figura 5.5: Porcentaje de paquetes a los cuales se les ha introducido un determinado retardo, según cada nivel de alisado.

En la Figura 5.6 y la Figura 5.7 se muestran las pérdidas de paquetes obtenidas para la aplicación P2P y el servicio TCP utilizado como tráfico de fondo respectivamente. Los resultados se presentan para cada capacidad del enlace del canal (cada subfigura) y los diferentes algoritmos de control de congestión para el tráfico de fondo TCP. Por ejemplo, la Figura 5.6c muestra los resultados de pérdida de paquetes obtenida por la aplicación P2P cuando la capacidad del canal es de $3\ Mbps$, observándose que *Sack* y *Reno* presentan valores similares de pérdida cuando se alisa a $4\ Mbps$.

De las gráficas presentadas, se puede notar que existe una diferencia en la pérdida de paquetes según la variante de control de congestión utilizada. Las pérdidas en el TCP dependen en gran medida de la implementación de los mecanismos de control de congestión. Además, no sólo el servicio FTP tiene pérdidas de paquetes, sino que también se afecta el tráfico de la aplicación P2P, que presenta los mayores niveles de pérdidas, lo cual se aprecia al comparar el nivel de porcentaje de pérdidas mostrado en la Figura 5.6 con respecto a la Figura 5.7 para una misma capacidad del enlace.

Por otro lado, se observa que existen algunas diferencias notables en cuanto a las pérdidas entre los diferentes algoritmos de control de congestión. Por ejemplo,

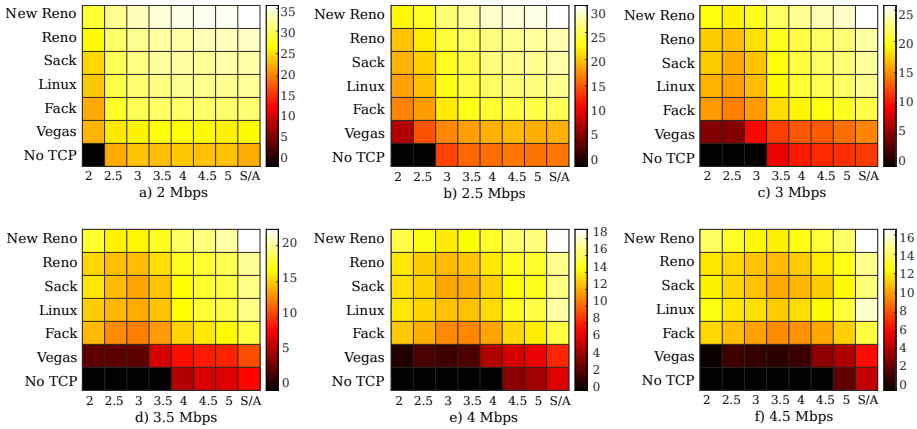

Figura 5.6: Pérdida de paquetes UDP para la aplicación P2P cuando comparte el enlace con un tráfico de fondo TCP.

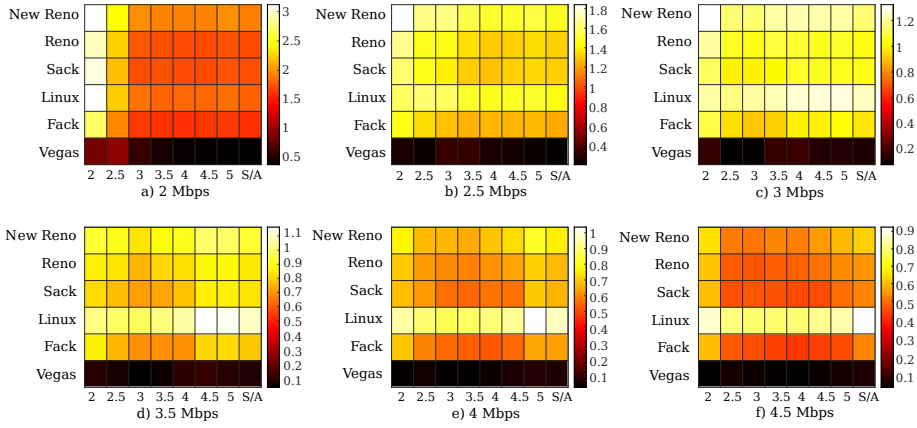

Figura 5.7: Pérdida de paquetes para el tráfico de fondo TCP cuando comparte el enlace con una aplicación P2P (UDP).

Vegas es el que menor cantidad de pérdidas presenta (tanto para SopCast como para el propio FTP). Esto se debe al comportamiento drástico de su control de flujo, ya que cuando detecta congestión en la transmisión detiene el envío de paquetes, lo que hace que no sea la variante más difundida en Internet. Por otra parte, *New Reno* presenta el peor valor de pérdidas. Esta variante puede detectar múltiples pérdidas de paquetes y admite múltiples retransmisiones pero la detección de una pérdida requiere demasiado tiempo y este retardo provoca la pérdida de gran cantidad de paquetes de información. El resto de las variantes (*Sack*, *Reno*, *Fack* y *Linux*) tienen un comportamiento similar, a medida que detectan congestión van disminuyendo el tamaño de su ventana y por tanto se van adaptando al entorno. En general, las pruebas realizadas muestran que el algoritmo de control de congestión del tráfico de fondo repercute en la QoS de otras aplicaciones que comparten el enlace, y por lo tanto, el uso del *Conformador* podría ser una implementación viable en sistemas VSS basados en dispositivos IoVT.

Uno nunca se da cuenta de lo que se ha hecho; uno sólo puede ver lo que queda por hacer.

Marie Curie

CAPÍTULO

6

Conclusiones y líneas futuras

6.1 Conclusiones

6.1.1 Integración y gestión de servicios

En esta tesis se ha presentado la puesta en marcha de un sistema $5G$ extremo a extremo. La arquitectura $5G$ se centró en la integración de varios componentes, entre ellos: $5G$ móvil, $4G$, SDR y Wi-Fi, así como también en las capacidades de softwarización, gestión y orquestación ofrecidas por el paradigma NFV y SDN. En este sentido, se implementó una plataforma capaz de proporcionar mecanismos de gestión y orquestación, por medio de OSM, de múltiples tecnologías de nubes que coexisten dentro de un mismo dominio, siendo este uno de los desafíos vigentes en las redes IoT. Además, se desarrolló un *plugin* en *OpenDaylight* que hizo posible la monitorización en tiempo real de la infraestructura y recursos de la red, como la latencia, el *jitter* y el rendimiento, lo que facilitó la orquestación inteligente del sistema $5G$ propuesto. Otro de los resultados relevantes presentados ha sido la baja latencia y el gran ancho de banda proporcionados en esta implementación, logrando la sensación de inmediatez y total inmersión durante una demostración realizada en colaboración con la Escuela de Música y Drama del Guidlhall en Londres y el Museo de la Puerta de Brandeburgo en Berlín, en la cual se realizó un concierto en tiempo real ofrecido entre artistas en las dos ubicaciones físicas.

En cuanto a la prueba de concepto, se presentó una arquitectura heurística de un VSS inteligente para vehículos de transporte público basado en dispositivos IoVT, que se utilizaron como nodos *Edge-Cloud*, una aplicación ML que detecta

objetos en conjunto con una transmisión de vídeo P2P. Los resultados muestran que la solución propuesta es viable en dispositivos de bajas capacidades. Por otro lado, es claro que los algoritmos de control de congestión para TCP tienen un gran impacto, no sólo para las aplicaciones que los utilizan, sino también para el tráfico que comparte el mismo enlace.

6.1.2 Métodos de optimización del tráfico

A la hora de optimizar el tráfico de una aplicación es necesario realizar un estudio previo de cómo la misma genera paquetes a la red y la influencia que podría tener en los dispositivos de las redes de acceso. Por este motivo, se presentó una comparativa del comportamiento del tráfico de una aplicación P2P-TV, en función de las políticas de *buffer*. Se ha mostrado que el *buffer* puede influir de manera negativa en el tratamiento dado a los paquetes de vídeo, provocando que los *peer* no contribuyan a la distribución de paquetes de vídeo. Se hace necesario por tanto tener en cuenta el tamaño de los paquetes que se generan, ya que algunas políticas penalizan los paquetes grandes. Teniendo en cuenta la gran variedad de tecnologías de acceso que se pueden encontrar en el escenario considerado, las medidas presentadas ilustran que independientemente del ancho de banda de salida a Internet, el comportamiento para cada tipo de *buffer* es similar en cuanto a retardo. Sin embargo, se observa que las pérdidas obtenidas en los *buffer* limitados en *bytes* son mucho menores que en los limitados en número de paquetes.

La metodología presentada permite conformar el tráfico de aplicaciones en tiempo real, con el propósito de ahorrar ancho de banda y utilizar eficientemente los recursos de la red. En el primer método estudiado se utilizó la técnica de compresión de cabeceras y dos políticas de multiplexión de tráfico generado por una aplicación P2P-TV con base en el protocolo UDP. Teniendo en cuenta la cantidad de paquetes generados, se ha mostrado que se pueden obtener valores importantes de ahorro de ancho de banda en redes residenciales. Se han desarrollado simulaciones con el propósito de estudiar el ahorro de ancho de banda para las distintas políticas de multiplexión propuestas. La primera se basa en la selección de un *período*, de forma que todos los paquetes que llegan durante ese intervalo de

tiempo son multiplexados y enviados juntos. La segunda, define un *umbral* para el tiempo de llegada entre paquetes, con el fin de multiplexar los que correspondan a la misma ráfaga. Los resultados muestran que con el empleo de las políticas propuestas se alcanzan ahorros significativos: el *uplink* puede reducirse entre un $26\,\%$ y un $33\,\%$ para la política basada en un *período* y entre un $33\,\%$ y un $35\,\%$ si se emplea un *umbral* para el tiempo entre paquetes. Se consigue además, una importante reducción del número de paquetes por segundo. De igual manera, se ha comprobado que los retardos añadidos por el proceso de multiplexión no perjudican la experiencia del usuario con la aplicación.

En el segundo método estudiado se utilizó una técnica de alisado con el fin de restringir el *throughput* instantáneo exigido por la aplicación a niveles razonables dadas las condiciones de la red de acceso, con el objetivo de prevenir la pérdida de paquetes y así lograr un aumento de la QoS. Las pruebas han demostrado que niveles más altos de alisado de tráfico pueden reducir los niveles de pérdida y aumentar el *throughput*. Como contrapartida, se introducen ciertos niveles de retardo en algunos paquetes que llegan en ráfagas al *buffer*. Estos niveles de retardo son tolerados por la aplicación utilizada; para estos casos se recomienda establecer un balance entre el retardo añadido y el *throughput* alcanzado.

6.2 Líneas futuras

Como una línea futura de este trabajo se propone mejorar la eficiencia en la transmisión de datos en las últimas versiones de la tecnología Wi-Fi (por ejemplo, Wi-Fi 5 y Wi-Fi 6). Con la rápida evolución de las tecnologías inalámbricas, se debe satisfacer la creciente demanda de tráfico en las redes, garantizando una mejor y más amplia cobertura, con mayores velocidades. Es por ello que, la eficiencia en la descarga de datos en Wi-Fi 6 podría mejorarse utilizando mecanismos de optimización de tráfico, basados en la agregación de tramas de diferentes aplicaciones, utilizando transmisión *multicast* a distintas velocidades y añadiendo nuevo métodos de seguridad.

De igual manera, se sugiere seguir investigando en el concepto de *network slicing* y cómo podría mejorarse su implementación en la RAN, fundamentalmente

en *Wireless LAN* (WLAN). Como se ha mencionado, las redes $5G$ integran diferentes tecnologías de radio (sistemas *Long Term Evolution* (LTE), $5G$ y tecnologías Wi-Fi), por lo que se precisa que en cada una de ellas se pueda implementar el *network slicing* de una manera eficiente.

Bibliografía

[1] A. M. Escolar, J. M. Alcaraz-Calero, P. Salva-Garcia, J. B. Bernabe, and Q. Wang, "Adaptive network slicing in multi-tenant 5g iot networks," *IEEE Access*, vol. 9, pp. 14 048–14 069, 2021. 1

[2] S. S. N. Perala, I. Galanis, and I. Anagnostopoulos, "Fog computing and efficient resource management in the era of internet-of-video things (iovt)," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018, pp. 1–5. 2

[3] H. Li, K. Ota, and M. Dong, "Learning iot in edge: Deep learning for the internet of things with edge computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, 2018. 2

[4] T. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through vm migration and transmission power control," *IEEE Transactions on Computers*, vol. PP, pp. 1–1, 10 2016. 2

[5] J. Liu, H. Guo, Z. M. Fadlullah, and N. Kato, "Energy consumption minimization for fiwi enhanced lte-a hetnets with ue connection constraint," *IEEE Communications Magazine*, vol. 54, no. 11, pp. 56–62, 2016. 2

[6] S. Sansó, C. Guerrero, I. Lera, and C. Juiz, "A platform for lightweight deployment of iotapplications based on a function-as-a-servicemodel," *IEEE Latin America Transactions*, vol. 17, no. 07, pp. 1155–1162, 2019. 2

[7] O. Nassef, L. Sequeira, E. Salam, and T. Mahmoodi, "Building a lane merge coordination for connected vehicles using deep reinforcement learning," *IEEE Internet of Things Journal*, pp. 1–1, 2020. 2

[8] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, 2016. 2

[9] L. Sequeira, J. Navajas, and J. Saldana, "The effect of the buffer size in qos for multimedia and bursty traffic: When an upgrade becomes a downgrade," *KSII Transactions on Internet and Information Systems*, vol. 8, pp. 3159–3176, Sept. 2014. 3, 11, 14

[10] J. Saldana, J. Fernandez-Navajas, J. Ruiz-Mas, and L. Casadesus, "Online games traffic multiplexing: Analysis and effect in access networks." *KSII Transactions on Internet & Information Systems*, vol. 6, 2012. 3, 15, 31

[11] I. Alhassan, M. Adu-Gyamfi, B. Kassim, and B. Abdallah, "Factors affecting the adoption of ict by administrators in the university for development studies tamale: Empirical evidence from the utaut model," *International Journal of Sustainability Management and Information Technologies*, vol. 4, 01 2018. 7

[12] M. Condoluci, F. Sardis, and T. Mahmoodi, "Softwarization and virtualization in 5g networks for smart cities," in *International Internet of Things Summit*. Springer, 2015, pp. 179–186. 7

[13] A. A. Barakabitze, N. Barman, A. Ahmad, S. Zadtootaghaj, L. Sun, M. G. Martini, and L. Atzori, "Qoe management of multimedia streaming services in future networks: A tutorial and survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 526–565, 2019. 8

[14] A. A. Barakabitze, I.-H. Mkwawa, L. Sun, and E. Ifeachor, "Qualitysdn: Improving video quality using mptcp and segment routing in sdn/nfv," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*. IEEE, 2018, pp. 182–186. 8

[15] X. Zhou, R. Li, T. Chen, and H. Zhang, "Network slicing as a service: enabling enterprisesówn software-defined cellular networks," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 146–153, 2016. 8

[16] I. Alam, K. Sharif, F. Li, Z. Latif, M. M. Karim, S. Biswas, B. Nour, and Y. Wang, "A survey of network virtualization techniques for internet of things using sdn and nfv," *ACM Comput. Surv.*, vol. 53, no. 2, apr 2020. 8

[17] Y. Jararweh, M. Al-Ayyoub, A. Darabseh, E. Benkhelifa, M. Vouk, and A. Rindos, "Sdiot: A software defined based internet of things framework," *Journal of Ambient Intelligence and Humanized Computing*, vol. 1, pp. 453–461, 08 2015. 8

[18] F. Z. Yousaf, M. Gramaglia, V. Friderikos, B. Gajic, D. von Hugo, B. Sayadi, V. Sciancalepore, and M. R. Crippa, "Network slicing with flexible mobility and qos/qoe support for 5g networks," in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2017, pp. 1195–1201. 8

[19] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "Nfv and sdn—key technology enablers for 5g networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2468–2478, 2017. 8

[20] A. De Domenico, Y.-F. Liu, and W. Yu, "Optimal virtual network function deployment for 5g network slicing in a hybrid cloud infrastructure," *IEEE Transactions on Wireless Communications*, vol. 19, no. 12, pp. 7942–7956, 2020. 9

[21] B. Omoniwa, R. Hussain, M. A. Javed, S. H. Bouk, and S. A. Malik, "Fog/edge computing-based iot (feciot): Architecture, applications, and research issues," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4118–4149, 2018. 9

[22] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1125–1142, 2017. 9

[23] P. Sethi and S. R. Sarangi, "Internet of things: architectures, protocols, and applications," *Journal of Electrical and Computer Engineering*, 2017. 9

[24] M. Kalverkamp and C. Gorldt, "Iot service development via adaptive interfaces: Improving utilization of cyber-physical systems by competence based user interfaces," in *2014 International Conference on Engineering, Technology and Innovation (ICE)*. IEEE, 2014, pp. 1–8. 9

[25] Y. Yang, "Fa2st: Fog as a service technology," in *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, vol. 1. IEEE, 2017, pp. 708–708. 10

[26] Y. Shi, G. Ding, H. Wang, H. E. Roman, and S. Lu, "The fog computing service for healthcare," in *2015 2nd International Symposium on Future Information and Communication Technologies for Ubiquitous HealthCare (Ubi-HealthTech)*. IEEE, 2015, pp. 1–5. 10

[27] "Ieee approved draft standard for low-rate wireless networks," *IEEE P802.15.4-REVd/D06, March 2020*, pp. 1–945, 2020. 10

[28] A. Betzler, C. Gomez, I. Demirkol, and J. Paradells, "Coap congestion control for the internet of things," *IEEE Communications Magazine*, vol. 54, no. 7, pp. 154–160, 2016. 10

[29] T. Sultana and K. A. Wahid, "Choice of application layer protocols for next generation video surveillance using internet of video things," *IEEE Access*, vol. 7, pp. 41 607–41 624, 2019. 10

[30] J. Saldana, M. Suznjevic, L. Sequeira, J. Fernandez-Navajas, M. Matijasevic, and J. Ruiz-Mas, "The effect of tcp variants on the coexistence of mmorpg and best-effort traffic," in *International Conference on Computer*

Communications and Networks (ICCCN), July 2012, pp. 1–5. 11, 58, 69, 85

[31] A. B. Vieira, P. Gomes, J. A. M. Nacif, R. Mantini, J. M. Almeida, and S. V. A. Campos, "Characterizing sopcast client behavior." *Computer Communications*, vol. 35, no. 8, pp. 1004–1016, 2012. 11, 12, 35

[32] T. Silverston and O. Fourmaux, "Measuring p2p iptv systems," in *Proceedings of ACM NOSSDAV*, vol. 7, 2007, p. 2. 11, 12, 43

[33] W.-C. Feng, F. Chang, W.-C. Feng, and J. Walpole, "Provisioning on-line games: a traffic analysis of a busy counter-strike server." in *Internet Measurement Workshop*. ACM, 2002, p. 151–156. 11, 14

[34] I. Quintana, J. Ruiz-Mas, J. Fernandez-Navajas, L. Casadesus, L. Sequeira, and J. Saldana, "Influencia del buffer del router en la distribución de vídeo p2p-tv," in *Actas del XXVII Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2012)*, Sept. 2012. 11

[35] T. Silverston, O. Fourmaux, K. Salamatian, and K. Cho, "Measuring P2P-TV systems on both sides of the world." in *ICME*. IEEE, 2010, p. 1321–1326. 11

[36] S. Tang, Y. Lu, J. M. Hernández, F. A. Kuipers, and P. V. Mieghem, "Topology Dynamics in a P2PTV Network." in *Networking*, ser. Lecture Notes in Computer Science, vol. 5550. Springer, 2009, p. 326–337. 11

[37] P. Eittenberger, U. R. Krieger, and N. M. Markovich, "Measurement and analysis of live-streamed p2ptv traffic," *Performance Modelling and Evaluation of Heterogeneous Networks, Proc. HET-NETs*, p. 14–16, 2010. 11, 12, 13

[38] M. Cha, P. Rodriguez, S. Moon, and J. Crowcroft, "On next-generation telco-managed p2p tv architectures," in *IPTPS'08 Proceedings of the 7th international conference on Peer-to-peer systems*, 2011. 11

[39] Y. Lu, B. Fallica, F. A. Kuipers, R. E. Kooij, and P. V. Mieghem, "Assessing the Quality of Experience of SopCast." *IJIPT*, vol. 4, no. 1, p. 11–23, 2009. 11

[40] B. Fallica, Y. Lu, F. Kuipers, R. Kooij, and P. V. Mieghem, "On the Quality of Experience of SopCast," in *Next Generation Mobile Applications, Services and Technologies, 2008. NGMAST '08. The Second International Conference on*, sept. 2008, p. 501–506. 11, 12, 13

[41] U. Krieger and R. Schwessinger, "Analysis and quality assessment of peer-to-peer IPTV systems," in *Consumer Electronics, 2008. ISCE 2008. IEEE International Symposium on*, april 2008, p. 1–4. 11

[42] Y. Liu, Y. Guo, and C. Liang, "A survey on peer-to-peer video streaming systems," *Peer-to-Peer Networking and Applications*, vol. 1, no. 1, p. 18–28, March 2008. 11

[43] A. Sentinelli, G. Marfia, M. Gerla, S. Tewari, and L. Kleinrock, "Will IPTV ride the peer-to-peer stream?" *IEEE Communications Magazine*, vol. 45, no. 6, p. 86, 2007. 12, 13, 35, 55, 59, 87

[44] L. Vu, I. Gupta, J. Liang, and K. Nahrstedt, "Measurement and modeling of a large-scale overlay for multimedia streaming," in *The Fourth International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness; Workshops*, ser. QSHINE-07. ACM, 2007, pp. 3:1–3:7. 12

[45] D. R. Choffnes and F. E. Bustamante, "Taming the torrent: a practical approach to reducing cross-isp traffic in peer-to-peer systems," in *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*, ser. SIGCOMM '08, 2008, pp. 363–374. 12

[46] D. Moltchanov, Y. Koucheryavy, and B. Moltchanov, "The effect of biased choice of peers on quality provided by p2p file sharing." in *Consumer Communications and Networking Conference (CCNC), 2012 IEEE*. IEEE, 2012, pp. 608–613. 12

[47] "http://www.sopcast.org/." 12

[48] W. Rutherford, L. Jorgenson, M. Siegert, P. Van Epp, and L. Liu, "16000–64000 b pmtu experiments with simulation: The case for super jumbo frames at supercomputing '05," *Optical Switching and Networking*, vol. 4, no. 2, pp. 121–130, 2007. 13

[49] R. Stanojevic and R. Shorten, "Trading link utilization for queueing delays: An adaptive approach." *Computer Communications*, vol. 33, no. 9, p. 1108–1121, 2010. 14

[50] L. Sequeira, J. Fernandez-Navajas, J. Saldana, J. R. Gallego, and M. Canales, "Describing the access network by means of router buffer modelling: a new methodology," *The Scientific World Journal*, 2014. 14, 61

[51] A. Vishwanath, V. Sivaraman, and M. Thottan, "Perspectives on router buffer sizing: recent results and open problems." *Computer Communication Review*, vol. 39, no. 2, p. 34–39, 2009. 14

[52] A. Dhamdhere and C. Dovrolis, "Open issues in router buffer sizing." *Computer Communication Review*, vol. 36, no. 1, p. 87–92, 2006. 14

[53] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden, "Part III: routers with very small buffers." *Computer Communication Review*, vol. 35, no. 3, p. 83–90, 2005. 14

[54] J. Saldana, J. Fernandez-Navajas, J. Ruiz-Mas, E. Viruete, and L. Casadesus, "Influence of online games traffic multiplexing and router buffer on subjective quality." in *Consumer Communications and Networking Conference (CCNC), 2012 IEEE*. IEEE, 2012, p. 462–466. 14

[55] J. Saldana, J. Murillo, J. Fernandez-Navajas, J. Ruiz-Mas, E. Viruete, and J. Aznar, "Evaluation of multiplexing and buffer policies influence on voip conversation quality," in *Consumer Communications and Networking Conference (CCNC), 2011 IEEE*, 2011, p. 378–382. 14

[56] J. Saldana, J. Fernandez-Navajas, J. Ruiz-Mas, J. I. Aznar, E. Viruete, and L. Casadesus, "Influencia del buffer del router en la multiplexión de juegos online," in *Actas del XXVI Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2011)*, Sept. 2011. 14

[57] J. Saldana, J. Murillo, J. Fernandez-Navajas, J. Ruiz-Mas, J. Aznar, and E. Viruete, "Bandwidth efficiency improvement of online games by the use of tunneling, compressing and multiplexing techniques," in *Performance Evaluation of Computer Telecommunication Systems (SPECTS), 2011 International Symposium on*, 2011, p. 227–234. 14

[58] B. Thompson, T. Koren, and D. Wing, "Tunneling Multiplexed Compressed RTP (TCRTP)," RFC 4170 (Best Current Practice), Internet Engineering Task Force, November 2005. 14, 15

[59] A. Vakili and J.-C. Grégoire, "QoE management for video conferencing applications," *Computer Networks*, vol. 57, no. 7, p. 1726–1738, 2013. 15

[60] J. Saldana, J. Fernandez-Navajas, J. Ruiz-Mas, J. I. Aznar, E. Viruete, and L. Casadesus, "First person shooters: can a smarter network save bandwidth without annoying the players?" *IEEE Communications Magazine*, vol. 49, no. 11, p. 190–198, 2011. 15, 16

[61] J. Saldana, L. Sequeira, J. Fernandez-Navajas, and J. Ruiz-Mas, "Traffic optimization for tcp-based massive multiplayer online games," in *Performance Evaluation of Computer and Telecommunication Systems (SPECTS), 2012 International Symposium on*, July, pp. 1–8. 15, 55

[62] V. Jacobson. (1990, Feb.) RFC1144: Compressing TCP/IP headers for low-speed serial links. 15, 34

[63] M. Degermark, B. Nordgren, and S. Pink, "IP Header Compression," February 1999. 15

[64] G. Pelletier and K. Sandlund, "RObust Header Compression Version 2 (ROHCv2): Profiles for RTP, UDP, IP, ESP and UDP-Lite," RFC 5225, April 2008. 16

[65] E. Ertekin and C. Christou, "Internet protocol header compression, robust header compression, and their applicability in the global information grid," *IEEE Communications Magazine*, vol. 42, no. 11, pp. 106–116, 2004. 16

[66] R. S. Prasad, C. Dovrolis, and M. Thottan, "Router buffer sizing for TCP traffic and the role of the output/input capacity ratio," *IEEE/ACM Trans. Netw.*, vol. 17, no. 5, p. 1645–1658, Oct. 2009. 16

[67] A. M. AlAdwani, A. Gawanmeh, and S. Nicolas, "A demand side management traffic shaping and scheduling algorithm," in *2012 Sixth Asia Modelling Symposium*, 2012, pp. 205–210. 17

[68] M. Marcon, M. Dischinger, K. P. Gummadi, and A. Vahdat, "The local and global effects of traffic shaping in the internet," in *2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011)*, 2011, pp. 1–10. 17

[69] N. Duffield, K. Ramakrishnan, and A. Reibman, "Save: an algorithm for smoothed adaptive video over explicit rate networks," *IEEE/ACM Transactions on Networking*, vol. 6, no. 6, pp. 717–728, 1998. 17

[70] T. Lakshman, A. Ortega, and A. Reibman, "Vbr video: tradeoffs and potentials," *Proceedings of the IEEE*, vol. 86, no. 5, pp. 952–973, 1998. 18

[71] S. S. Lam, S. Chow, and D. K. Y. Yau, "An algorithm for lossless smoothing of mpeg video," in *Proceedings of the Conference on Communications Architectures, Protocols and Applications*, ser. SIGCOMM '94. New York, NY, USA: Association for Computing Machinery, 1994, p. 281–293. 18

[72] B. Vandalore, W. chi Feng, R. Jain, and S. Fahmy, "A Survey of Application Layer Techniques for Adaptive Streaming of Multimedia." *Real-Time Imaging*, vol. 7, no. 3, p. 221–235, 2001. 18

[73] J. Rexford, S. Sen, J. Dey, W.-c. Feng, J. Kurose, J. Stankovic, and D. Towsley, "Online smoothing of live, variable-bit-rate video," in *Proceedings of 7th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'97)*. IEEE, 1997, pp. 235–243. 18

[74] 3GPP, "Evolved universal terrestrial radio access (e-utra); nb-iot; technical report for bs and ue radio transmission and reception," 3rd Generation Partnership Project (3GPP), Technical Report (TR) 36.802, 2016, release 13. 22

[75] G. Mountaser, M. L. Rosas, T. Mahmoodi, and M. Dohler, "On the feasibility of mac and phy split in cloud ran," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, 2017, pp. 1–6. 23

[76] 3GPP, "Study on new radio access technology: Radio access architecture and interfaces," 3rd Generation Partnership Project (3GPP), Technical Report (TR) 38.801, 2017, release 14. 23

[77] ——, "System architecture for the 5g system," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 23.501, 2017, release 15. 24

[78] A. C. Baktir, A. Ozgovde, and C. Ersoy, "How can edge computing benefit from software-defined networking: A survey, use cases, and future directions," *IEEE Communications Surveys Tutorials*, vol. 19, no. 4, pp. 2359–2391, 2017. 24

[79] B. Forum, "Hybrid access broadband network architecture," Broadband Forum, Technical Report (TR) 348, 2016, issue 1. 24

[80] 3GPP, "Study on access traffic steering, switch and splitting support in the 5g system architecture," 3rd Generation Partnership Project (3GPP), Technical Report (TR) 23.793, 2018, release 16. 24

[81] M. Condoluci, S. Johnson, V. Ayadurai, M. Lema, M. Cuevas, M. Dohler, and T. Mahmoodi, "Fixed-mobile convergence in the 5g era: From hybrid access to converged core," *IEEE Network*, vol. PP, pp. 1–8, 02 2019. 24

[82] R. Bryant *et al.*, "Accelerating nfv delivery with openstack," *OpenStack Foundation Report - White Paper*, 2016. 26

[83] D. Bonfiglio, M. Mellia, M. Meo, N. Ritacca, and D. Rossi, "Tracking down skype traffic," in *IEEE INFOCOM 2008 - The 27th Conference on Computer Communications*, April 2008, pp. 261–265. 31

[84] M. Amiri, H. Osman, and S. Shirmohammadi, "Datacenter traffic shaping for delay reduction in cloud gaming," in *2016 IEEE International Symposium on Multimedia (ISM)*, Dec 2016, pp. 569–574. 31

[85] "Network-aware p2p-tv application over wise networks." [Online]. Available: https://cordis.europa.eu/project/id/214412/ 40, 41, 48

[86] E. Alessandria, M. Gallo, E. Leonardi, M. Mellia, and M. Meo, "P2p-tv systems under adverse network conditions: A measurement study," in *IEEE INFOCOM*, April 2009, pp. 100–108. 42

[87] I. Quintana, A. Tsiopoulos, M. Lema, F. Sardis, L. Sequeira, J. Arias, A. Raman, A. Azam, and M. Dohler, "The making of 5g: Building an end-to-end 5g-enabled system," *IEEE Communications Standards Magazine*, vol. 2, no. 4, pp. 88–96, 2018. 46

[88] P. Braun, S. Pandi, R. Schmoll, and F. Fitzek, "On the study and deployment of mobile edge cloud for tactile internet using a 5g gaming application," in *Consumer Communications and Networking Conference (CCNC), 2017 IEEE*, January 2017, pp. 154–159. 46

[89] J. Manner, "[jtg] jugi's traffic generator." [Online]. Available: http://www.netlab.tkk.fi/~jmanner/jtg.html 48

[90] "[caida] cooperative association for internet data analysis." [Online]. Available: https://www.caida.org/ 48

[91] Q. Chen, F. R. Yu, T. Huang, R. Xie, J. Liu, and Y. Liu, "An integrated framework for software defined networking, caching, and computing," *IEEE Network*, vol. 31, no. 3, pp. 46–55, 2017. 75

[92] A. Israel *et al.*, "Osm release three - a technical overview," *ETSI OSM Community White Paper*, October 2017. 75

[93] "Guildhall school of music & drama." [Online]. Available: https://www.gsmd.ac.uk/ 80

[94] "City of london to host the world's first 5g connected theatre," 2018. [Online]. Available: https://news.cityoflondon.gov.uk/city-of-london-to-host-the-worlds-first-5g-connected-theatre/ 80

[95] "King's and city of london host world's first 5g connected theatre performance," 2018. [Online]. Available: https://www.kcl.ac.uk/news/kings-and-city-of-london-host-worlds-first-5g-connected-theatre-performance-2 80

[96] "World's first 5g distributed concert london-berlin," Youtube, 2018. [Online]. Available: https://www.youtube.com/watch?v=mB_w-ml-dZY 80

[97] A. Sammoud, A. Kumar, M. Bayoumi, and T. Elarabi, "Real-time streaming challenges in internet of video things (iovt)," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2017, pp. 1–4. 84

[98] "https://www.nsnam.org/." 84

[99] I. Quintana, L. Sequeira, J. Fernandez, J. Ruiz, and J. Saldana, "Minimizing the impact of p2p-tv applications in access links." *IEEE Latin America Transactions*, vol. 17, no. 02, pp. 183–192, 2019. 85

[100] L. Sequeira, J. L. de la Cruz, J. Ruiz-Mas, J. Saldana, J. Fernandez-Navajas, and J. Almodovar, "Building an sdn enterprise wlan based on virtual aps," *IEEE Communications Letters*, vol. 21, no. 2, pp. 374–377, 2016. 87

Declaración

Por la presente declaro que he producido esta obra sin la prohibición de terceros y sin hacer uso de los medios que no sean los especificados. Además, que no existe ningún conflicto de intereses en relación con la publicación de esta tesis. El autor no trabaja con ninguna de las empresas cuyos productos se citan en el presente trabajo, ni tiene ninguna relación comercial o asociación con dichas empresas.

Este trabajo de tesis se llevó a cabo a partir del año 2011 hasta el 2022 bajo la supervisión de Dr. José Ruíz Más en la Universidad de Zaragoza.

Londres, 14 de enero de 2022

Idelkys Quintana

Esta tesis se terminó de escribir en Londres el
14 de enero de 2022