UNIVERSIDAD COMPLUTENSE DE MADRID
FACULTAD DE INFORMÁTICA

TESIS DOCTORAL

Técnicas de Monitorización en Transmisiones Multimedia para Redes Definidas por Sofware

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Jesús Antonio Puente Fernández

Director

Luis Javier García Villalba

Madrid

Técnicas de Monitorización en Transmisiones Multimedia para Redes Definidas por Software

TESIS DOCTORAL

Memoria presentada para obtener el título de
Doctor por la Universidad Complutense de Madrid
en el Programa de Doctorado en Ingeniería Informática

Jesús Antonio Puente Fernández

Director

Luis Javier García Villalba

Facultad de Informática
Universidad Complutense de Madrid

Madrid, Septiembre de 2020

UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA

TESIS DOCTORAL

Técnicas de Monitorización en Transmisiones Multimedia para Redes Definidas por Software

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Jesús Antonio Puente Fernández

DIRECTOR

Luis Javier García Villalba

Tesis Doctoral presentada por el doctorando Jesús Antonio Puente Fernández en la Facultad de Informática de la Universidad Complutense de Madrid para la obtención del título de Doctor por la Universidad Complutense de Madrid en el Programa de Doctorado en Ingeniería Informática.

Terminada en Madrid el 21 de Septiembre de 2020.

Título:

Técnicas de Monitorización en Transmisiones Multimedia para Redes Definidas por Software

Doctorando:

Jesús Antonio Puente Fernández (jesusantoniopuente@ucm.es)
Departamento de Ingeniería del Software e Inteligencia Artificial
Facultad de Informática
Universidad Complutense de Madrid
28040 Madrid, España

Director:

Luis Javier García Villalba (javiergv@fdi.ucm.es)

Esta tesis doctoral ha sido realizada dentro del grupo de investigación GASS (Grupo de Análisis, Seguridad y Sistemas, grupo 910623 del catálogo de grupos reconocidos por la UCM) como parte de las actividades del proyecto de investigación SELFNET (Framework for Self-Organized Network Management in Virtualized and Software Defined Networks) financiado por la Comisión Europea dentro del Programa Marco de Investigación e Innovación Horizonte 2020 (H2020-ICT-2014-2/671672-SELFNET).

A mi padre Jesús, a mi madre Josefa y a mi hermana Rocío.

Índice General

Índice de Figuras

Índice de Tablas

Índice de Algoritmos

Lista de Acrónimos

2G	2^{th} *Generation*
3D	*3 Dimensions*
3G	3^{th} *Generation*
4G	4^{th} *Generation*
5G	5^{th} *Generation*
6G	6^{th} *Generation*
AA	*Active Applications*
ACL	*Access Control List*
AP	*Access Point*
API	*Application Programming Interface*
B5G	*Beyond 5G*
BGP	*Border Gateway Protocol*
BS	*Base Station*
CAPEX	*Capital Expenditure*
CLI	*Command Line Interface*
CN	*Core Network*
CPU	*Central Processing Unit*

D2D	*Device to Device*
DASN	*Demand-Addressable Sensor Network*
DDoS	*Distributed Denial of Service*
DDRP	*Data-Driven Routing Protocol*
DHCP	*Dynamic Host Control Protocol*
DNS	*Domain Name Server*
DoS	*Denial of Service*
DPI	*Data Packet Inspection*
DTM	*Digital Terrain Model*
E2E	*End to End*
EE	*Execution Environments*
ETSI	*European Telecommunications Standards Institute*
FC	*Fog Computing*
ForCES	*Forwarding and Control Element Separation Working Group*
FRP	*Functional Reactive Programming*
GPS	*Global Positioning System*
HD	*High Definition*
HoN	*Health of Network*
HTTP	*HyperText Transfer Protocol*
HTTPS	*HyperText Transfer Protocol Secure*

IDS	*Intrusion Detection System*
IETF	*Internet Engineering Task Force*
IoT	*Internet of Things*
IoT-A	*Internet of Things-Architecture*
IP	*Internet Protocol*
IPS	*Intrusion Prevention System*
LLDP	*Link Layer Discovery Protocol*
LoS	*Line-of-Sight*
MAC	*Medium Access Control*
MEC	*Multi-Access Edge Computing*
MEMS	*Micro-Electro-Mechanical Systems*
MFC	*MobileFlow Controller*
MFFE	*MobileFlow Forwarding Engine*
MITM	*Man In The Middle*
MME	*Mobile Management Entity*
NETCONF	*Network Configuration Protocol*
NFV	*Network Function Virtualization*
NIDS	*Network Intrusion Detection System*
NIPS	*Network Intrusion Prevention System*
NOS	*Network Operating Systems*
OF	*OpenFlow*
ONF	*Open Networking Foundation*

OPEX	*Operational Expenditure*
OS	*Operative System*
OSPF	*Open Shortest Path First*
OTT	*Over The ToP*
PAF	*Path Assignment Function*
PGW	*Packet Data Network Gateway*
QFF	*QoE Fairness Framework*
QMOF	*Quality of Service Matching and Optimization Function*
QoE	*Quality of Experience*
QoS	*Quality of Service*
RAN	*Radio Access Network*
RCP	*Routing Control Platform*
REST	*Representational State Transfer*
RFC	*Request For Comments*
RFID	*Radio Frequency IDentification*
SCS	*Sensor Control Server*
SDMN	*Software-Defined Mobile Network*
SDN	*Software Defined Network*
SDR	*Software Defined Radio*
SDSN	*Software Defined Sensor Network*
SDWCN	*Software Defined Wireless Cellular Network*
SE	*Security Enhanced*

SGW	*Serving Gateway*
SNMP	*Simple Network Management Protocol*
SOM	*Self Organizing Map*
SON	*Self Organizing Networks*
TCP	*Transmission Control Protocol*
TLS	*Transport Layer Security*
UAV	*Unmanned Aerial Vehicle*
VLAN	*Virtual Local Area Networks*
VM	*Virtual Machine*
VNF	*Virtual Network Function*
VNM	*Virtual Network Migration*
VoIP	*Voice over IP*
WAN	*Wide Area Network*
WCN	*Wireless Cellular Network*
WNFV	*Wireless Network Function Virtualization*
WSN	*Wireless Sensor Network*

Abstract

The development of new services and applications online, both in fixed landlines and in mobile devices, has led to an exponential growth in traffic circulating on the Internet. The continuous growth of video streaming demand should be noted as a significant contributor to the total amount of information circulating. This subsequently forces the network infrastructure to provide greater speed, flexibility and quality of service. In addition, the new concept of Quality of Experience (in short, QoE) in multimedia communications has experienced an important development in recent years, since it considers the expectations of the user to measure the quality with respect to a service or application.

On the other hand, traditional network equipment and protocols were not designed to support a high level of scalability and mobility. The correct operation of a network is linked to the ability to monitor and detect specific events such as delays, packet losses, etc. Consequently, the existing mechanisms do not provide the expected performance or require the deployment of new additional hardware. This not only generates an increase in the implementation and operation costs, but also in the deployment time of the services required by the user. For this reason, it is essential to create an open environment that enables rapid experimentation with new technologies and devices that are not limited to one manufacturer and that allows the behaviour of the network to be personalized.

Software Defined Networks, abbreviated to SDN (for its acronym in English), is a new paradigm that eliminates the rigidity present in current architectures and improves the flexibility and administration of networks. SDN proposes the centralized control of the network decoupling the control plane and the data plane in the routing devices and establishes an open communication interface between them. OpenFlow is the first SDN standard that has been widely used in different research projects.

Even though SDN and OpenFlow open new perspectives in the field of monitoring, they also introduce some challenges. The main one is related to the workload of both the data plane and the control plane. If the controller requests information to each switch continuously, it will provide more accurate data and can manage the network more effectively. However, this process can lead to excessive use of processing resources of the controller and, consequently, negatively affect the performance of priority tasks. In addition, switches will use more resources and energy to respond to the continuous requirements of the controller.

On the other hand, not all services require the same metrics. Some services (such as video streaming) are affected by the delay, while for others (such as email) this factor is indifferent. This diversity generates the need to organize the metrics according to the user's requirements, providing a scalable structure.

It is the intention of this Thesis to study the relationships which exist in concepts such as: Software Defined Network, Network Function Virtualization, Quality of Service and Network Management, considering Software Defined Network and the Network Function

Virtualization as keys for the development of a new generation of services and applications.

Firstly, this Thesis proposes a routing algorithm for multimedia transmissions in SDN networks where the purpose is to improve the QoE for the user. This routing algorithm uses Floodlight as a controller and OpenFlow as a communication protocol in the SDN network. The experimentation carried out shows the positives of the algorithm, improving the multimedia transmissions with respect to the current existing literature. This routing algorithm is the basis on which the different monitoring techniques proposed in this Thesis are based.

Similarly, this Thesis proposal is based on the previous contribution, a set of monitoring techniques using different architectures that allow for the reduction of the number of statistical queries and, therefore, reducing the load on the network controller related to the monitoring process. This reduction is carried out using an orchestrator system that determines the information polling time based on the controller load and the size of the network, whilst also providing a modular structure for the creation, updating and continuous improvement of monitoring algorithms. In this sense, the different monitoring techniques proposed in this Thesis are based on optimization algorithms that monitor three parameters: the data rate, the error rate and the delay. These algorithms have been implemented and simulated in the Floodlight controller using the Mininet simulation software with video transmissions from a server to a client computer.

Keywords: Controller, Data Rate, Error Rate, Floodlight, Mininet, Monitoring Techniques, OpenFlow, Optimization, Query, Routing Algorithm, Software Defined Networks, SDN, Switch, Video.

Resumen

El desarrollo de nuevos servicios y aplicaciones en línea, tanto en terminales fijos como en dispositivos móviles, ha dado lugar a un crecimiento exponencial del tráfico que circula por Internet. Mención especial merece el continuo crecimiento en la demanda de contenido multimedia (particularmente vídeo), que ocupa gran cantidad del total de la información que circula, lo que obliga a la infraestructura de red a brindar mayor velocidad, flexibilidad y calidad de servicio. Además, el nuevo concepto de Calidad de Experiencia (QoE) en comunicaciones multimedia ha tenido un importante desarrollo en los últimos años, ya que tiene en cuenta las expectativas del usuario para medir la calidad respecto a un determinado servicio o aplicación.

Por otro lado, los equipos de red y los protocolos tradicionales no fueron diseñados para soportar un alto nivel de escalabilidad y movilidad. La correcta operación de una red está ligada a la capacidad de monitorización y detección de eventos específicos como retardos, pérdidas de paquetes, etc. Consecuentemente, los mecanismos existentes no proporcionan el rendimiento esperado o requieren el despliegue de un nuevo hardware adicional. Esto no sólo genera un incremento en los costes de implementación y operación, sino también en el tiempo de despliegue de los servicios requeridos por el usuario. Por tal motivo resulta indispensable la creación de un ambiente abierto que posibilite la rápida experimentación con nuevas tecnologías y dispositivos que no estén limitados a un fabricante y que permita la personalización del comportamiento de la red.

Las Redes Definidas por Software, abreviadamente SDN (de sus siglas en inglés), es un nuevo paradigma que elimina la rigidez presente en las arquitecturas actuales y mejora la flexibilidad y administración de las redes. SDN propone el control centralizado de la red, desacoplando el plano de control y el plano de datos en los dispositivos de encaminamiento y estableciendo una interfaz abierta de comunicación entre ellos. OpenFlow es el primer estándar SDN que ha sido ampliamente utilizado en diferentes proyectos de investigación.

Si bien SDN y OpenFlow abren nuevas perspectivas en el campo de la monitorización, también introducen algunos desafíos. El principal está relacionado con la carga de trabajo tanto del plano de datos como del plano de control. Si el controlador solicita información de cada encaminador de forma continua, proporcionará datos más precisos y podrá gestionar de mejor manera la red. Sin embargo, este proceso puede generar una excesiva utilización de los recursos de procesamiento del controlador y, en consecuencia, afectar negativamente al rendimiento de tareas prioritarias en la red. Además, los encaminadores utilizarán mayor cantidad de recursos y energía para responder a los continuos requerimientos del controlador.

Por otro lado, no todos los servicios requieren las mismas métricas. Algunos servicios (como la transmisión de vídeo) se ven afectados por el retardo, mientras que para otros (como el correo electrónico) este factor es indiferente. Esta diversidad genera la necesidad de organizar las métricas en función de los requerimientos del usuario, proporcionando

una estructura escalable.

Con este propósito la presente tesis analiza la relación existente entre conceptos tales como: Redes Definidas por Software, Virtualización de Funciones de Red, Calidad de Servicio y Gestión de Red, considerando a las Redes Definidas por Software y a las Funciones de Red Virtualizadas tecnologías claves para el desarrollo de una nueva generación de servicios y aplicaciones.

La presente Tesis propone en primer lugar un algoritmo de encaminamiento para transmisiones multimedia en redes SDN cuya finalidad es mejorar la QoE al usuario. Este algoritmo de encaminamiento utiliza Floodlight como controlador y OpenFlow como protocolo de comunicación en la red SDN. La experimentación realizada demuestra la bondad del algoritmo mejorando las transmisiones multimedia respecto a la literatura existente hasta el momento.

La presente Tesis propone asimismo, a partir de la contribución anterior, un conjunto de técnicas de monitorización utilizando diferentes arquitecturas que permiten reducir el número de consultas estadísticas y, por tanto, reducir la carga del controlador de la red relativa al proceso de monitorización. Esta reducción se realiza mediante un sistema orquestador que determina el tiempo de petición de información en base a la carga del controlador y al tamaño de la red, proporcionando además una estructura modular para la creación, actualización y continua mejora de los algoritmos de monitorización. Estas técnicas de monitorización se basan en algoritmos de optimización que monitorizan tres parámetros: la tasa de transmisión de datos, la tasa de pérdidas y el retardo. Dichos algoritmos han sido implementados y simulados en el controlador Floodlight utilizando el software de simulación Mininet con transmisiones de vídeo desde un servidor a un ordenador cliente.

Palabras clave: Algoritmo de Encaminamiento, Controlador, Consultas, Floodlight, Mininet, Multimedia, OpenFlow, Optimización, Redes Definidas por Software, SDN, Tasa de Datos, Tasa de Error, Técnicas de Monitorización, Vídeo.

Parte I

Descripción de la Investigación

Capítulo 1

Introducción

Actualmente, el desarrollo de una nueva generación de servicios y aplicaciones se encuentra limitado por el creciente coste de las inversiones de capital *Capital Expenditure* (CAPEX) y de los gastos operativos *Operational Expenditure* (OPEX) en las redes de telecomunicaciones. El alto nivel de complejidad en los sistemas e infraestructuras de telecomunicaciones demandan una continua revisión y actualización. De igual manera, los procesos de mantenimiento se realizan únicamente mediante la reconfiguración manual de equipos. Los operadores invierten ingentes recursos en mitigar un gran número de problemas en la red, tales como fallos de los enlaces, ataques de seguridad, degradación de la calidad de servicio, actualizaciones de software, entre otros. En algunos casos, la única solución disponible es el reemplazo físico de los equipos ocasionando interrupción parcial o total del tráfico y violaciones en el nivel de servicio contratado.

Las arquitecturas de red tradicionales experimentan considerables limitaciones en el aprovisionamiento de nuevos servicios de red, especialmente en términos de operatividad, seguridad, calidad de servicio, costes de operación y mantenimiento. La íntima unión entre el plano de datos y el plano de control en los dispositivos de red, el creciente número protocolos y el acceso privativo y limitado al código fuente son las principales restricciones que impiden la innovación y personalización de servicios. Las actuales estrategias de mitigación de los problemas de red consisten principalmente en la reconfiguración manual de los equipos, los cuales causan interrupciones en la operación normal de la red. En algunos casos, la única solución factible incluye la instalación de equipamiento y funcionalidades adicionales, tales como routers, balanceadores, firewalls y *Data Packet Inspection* (DPI) entre otros.

La rápida proliferación del uso de dispositivos móviles ha revelado la falta de capacidad de las redes actuales para acomodar la gran cantidad de información que tendrán que gestionar. Esta situación ha dado lugar al desarrollo de una nueva generación de redes móviles no solo para proporcionar soluciones a tales problemas, sino también para mejorar muchas características de sus predecesoras. Capacidades mejoradas, relacionadas con la transferencia de datos masivos, baja latencia, interoperabilidad o la reducción en el

consumo de energía, permiten una mejor *Quality of Experience* (QoE) para los usuarios finales. Alcanzar estos objetivos requiere una gran capacidad de innovación, tales como velocidades de datos de alta velocidad o una mejor gestión de la información y métodos de análisis. La última parte tiene un impacto significativo en modelos de negocio basados en servicios y aplicaciones en tiempo real (e-health, e-security, *Voice over IP* (VoIP), transmisión de vídeo, etc). Sin embargo, en la actualidad, el desarrollo de estos servicios está limitado por el lento proceso de estandarización y el bajo desempeño en las estrategias de gestión y toma de decisiones.

Esta descomposición permite el control centralizado de la red con mayores capacidades de automatización y simplificación de tareas de gestión, mientras se acelera la innovación de nuevas aplicaciones de alto nivel. Por su parte, *Network Function Virtualization* (NFV) permite implementación de funciones de red como instancias virtualizadas, ejecutándose en un hardware genérico. El enfoque escalable proporcionado por NFV permite que *Virtual Network Function* (VNF) se puedan implementar en cualquier momento y en cualquier lugar, mientras que anteriormente llevaba más tiempo (en comparación con las funciones tradicionales). Desde el punto de vista técnico, *Software Defined Network* (SDN) y NFV son tecnologías complementarias, y juntas podrían facilitar la configuración y personalización de red. Además, conceptos como Cloud Computing, *Self Organizing Networks* (SON) e inteligencia artificial permiten el fácil despliegue de servicios y gestión mejorada del tráfico basada en decisiones inteligentes.

El término Monitorización de red describe el uso de un sistema que constantemente monitoriza una red de ordenadores en busca de componentes defectuosos o lentos, para luego informar a los administradores de redes mediante diferentes sistemas de notificación.

Llevar a cabo la monitorización de las redes informáticas es una solución muy eficaz para anticiparse a los problemas. Aun así, en muchas ocasiones se pueden dar fallos y errores, pero éstos se deben aprovechar para aprender de ellos y así no caer en lo mismo en el futuro. Por este motivo, una vez se detecten los fallos correspondientes es importante que se establezcan medidas preventivas para que así no vuelva a ocurrir lo mismo una y otra vez.

La presente tesis doctoral propone un estudio de diferentes técnicas de monitorización de una red SDN implementadas en un framework que permite monitorizar su estado. La investigación incluye un modelo de recolección y almacenamiento de métricas relacionadas con infraestructura física, elementos virtualizados y flujos de información.

De esta forma, esta investigación tiene como objetivo resolver el problema de reducción de consultas estadísticas obteniendo los resultados más precisos posible del estado de la red. Con este fin, el framework recolecta la información recibida de los dispositivos de red en un modelo genérico de datos apto para cualquier tipo de métrica. Esta arquitectura también permite la inclusión de métricas personalizadas. Los experimentos realizados verifican que las propuestas de optimización presentadas permiten recolectar métricas del plano de datos y también mejorar la calidad de servicio en comparación con el enfoque tradicional del

mejor esfuerzo sin sobrecargar con tráfico el controlador de la red.

Finalmente, aparte del objetivo principal de esta tesis, también se incluye un estudio extenso de la aplicación de las SDN sobre otras tecnologías como *Software Defined Sensor Network* (SDSN) y *Wireless Cellular Network* (WCN) para dar solución a sus problemas y desafíos. De la misma manera, esta tesis también analiza el ámbito de la seguridad sobre los diferentes planos de SDN proponiendo diferentes soluciones a dichos problemas de seguridad.

1.1. Motivación

A día de hoy, es habitual el uso de la virtualización de servidores, planteamientos de nuevos modelos de negocios, nuevas aplicaciones en dispositivos de todo tipo (ordenadores personales, teléfonos inteligentes, tabletas, etc.) y nuevos servicios online entre otros. Estos aspectos platean grandes demandas de términos de rendimiento y velocidad que las redes tienen que satisfacer en la experiencia del usuario. Este uso tan demandado ha provocado la aparición de tecnologías como SDN y NFV las cuales han focalizado el interés de la comunidad de investigación. Añadiéndose a la comunidad investigadora, los proveedores de servicios tienen varias oportunidades en ambas tecnologías ya que proporcionan la ruptura y rigidez que las arquitecturas de red tradicionales no tenían. Por tanto, SDN y NFV han dejado obsoleta la arquitectura de red tradicional con su alto coste de mantenimiento y procedimientos para su actualización de forma compleja. Además, estas dos nuevas tecnologías proporcionan la capacidad de aportar servicios e innovaciones que las tradicionales no eran capaces.

Por otro lado, el número de dispositivos conectados a la red ha crecido de forma exponencial en los últimos años y junto con el incremento de la capacidad de cómputo ha permitido la generación de nuevos servicios en tiempo real como por ejemplo la transmisión de vídeo bajo demanda. En este ámbito, entra el rol de la monitorización de redes para proporcionar la mejor QoE y *Quality of Service* (QoS). Para contribuir al estudio de las causas de dichos inconvenientes, se necesita de la adaptación de estrategias de monitorización y elaboración de nuevas metodologías capaces de hacerles frente. De igual forma, los mecanismos de análisis de datos y de inteligencia de red son fundamentales para resolver o mitigar problemas potenciales. Debido a esto, otro de los temas clave para la comunidad de investigación es la provisión de capacidades de análisis de datos e inteligencia en redes SDN.

En este contexto, en el presente trabajo se proponen distintas técnicas de monitorización que permitan coleccionar estadísticas del estado de los dispositivos de redes SDN y por tanto, tener una visión global de toda la red. Además, otro objetivo de estas técnicas de monitorización es poder ser adaptadas en entornos de redes reales. De esta forma se podrá tener una visión completa monitorizada y precisa de la red en tiempo real para ser utilizada en varios aspectos posteriores.

1.2. Objetivos de la Investigación

Teniendo en cuenta las necesidades del estado de los dispositivos en redes SDN, el objetivo principal de esta tesis se centra en estudiar y proporcionar distintos métodos de monitorización y su eficiencia en entornos de redes reales además de identificar ventajas, inconvenientes y riesgos derivados de los entornos a monitorizar. Por lo tanto, se ha resuelto el problema de optimización del número de peticiones de

Tomando en cuenta la necesidad de conocer el contexto de la situación en una red SDN, se han cubierto los siguientes objetivos durante esta investigación:

- En primer lugar, se revisa el estado del arte relacionado con SDN, NFV y los diferentes tipos de monitorización para definir qué elementos deben ser tomados en cuenta en el proceso de diagnóstico y cómo las tecnologías actuales permiten cubrir dichas necesidades. Los requerimientos y las ventajas de incorporar capacidades de análisis de datos son también discutidos.

- Como segundo paso, se identifica el problema a tratar: el objetivo de esta tesis doctoral es resolver el problema de la optimización de la monitorización construyendo una herramienta que optimice el número de consultas estadísticas a los switches encargados del enrutamiento del tráfico.

- Más adelante, se extrae y evalúan las características más comunes de las técnicas aplicadas para la monitorización de la red.

- A continuación, se procede a la selección de un conjunto de estrategias de referencia para la monitorización de los dispositivos de red basado en su eficacia en escenarios experimentales.

- Una vez que el contexto situacional y los requerimientos para monitorizar redes SDN son definidos, se propone la implementación de una herramienta con varias técnicas de monitorización para la obtención de estadísticas sobre técnicas de colección convencionales.

- Después de ser implementadas dichas técnicas de monitorización, se procede a la evaluación a través de simulaciones para comprobar la precisión de las diferentes propuestas de monitorización, aplicando y sin aplicar las optimizaciones. Dichas simulaciones se realizan para obtener el número de consultas estadísticas sobre los switches mediante la transmisión de video de un servidor a un cliente a través de una red SDN.

- Por último, se valora y discuten los resultados de las propuestas implementadas en busca de si se ha resuelto el problema de optimización del número de consultas estadísticas.

1.3. Identificación del Problema

Vivimos en un mundo donde la tecnología está a la orden del día, en el que la evolución y los cambios están cada vez más presentes. Cada día sumamos nuevos dispositivos a nuestras redes, bien sea en nuestro hogar, negocio u otros entornos. Sin darnos cuenta, utilizamos un gran número de aparatos electrónicos y, si no hacemos un buen uso, podríamos tener graves consecuencias. Es aquí donde aparece la figura de la monitorización de redes.

El aumento de dispositivos conectados a una misma red incrementa el riesgo de mal funcionamiento, ralentización o fallos. Esto puede perjudicar el estado de nuestros equipos e incluso comprometer la seguridad de nuestra red.

Para reducir estos riesgos y prevenir fallos existen los sistemas de monitorización de redes, que se encargan de analizar los datos del funcionamiento de todos los equipos conectados a la red para garantizar su control, lo que permite optimizar su desempeño.

La optimización de redes utiliza la monitorización del tráfico de la plataforma como una manera fácil de identificar problemas de rendimiento y, en el mejor de los casos, prevenirlos. Cuando monitorizamos un segmento de red, podemos observar el tráfico Unicast, como el tráfico Broadcast y Multicast, siendo, por regla general, mucho mayor la cantidad de paquetes Unicast, convirtiéndose en un elemento vital de análisis.

Sin embargo, somos conscientes de que la presencia de tráfico Broadcast y Multicast implica un riesgo en el rendimiento global de la red. La generación de una cantidad excesiva de este tipo de paquetes puede llevar incluso al colapso total de la plataforma, por lo que hay que prestar especial interés.

El efecto de este tráfico no debería ser evaluado por la cantidad total de paquetes, sino por el efecto que estos paquetes tienen y pudieran tener en un futuro sobre el rendimiento global de la plataforma.

Recordemos que cada paquete Broadcast en particular, implica que todos los switches del segmento de red harán una copia de dicho paquete y lo colocarán en cada uno de sus puertos. Además, cada dispositivo deberá ejecutar una interrupción de entrada/salida para analizarlo.

Si partimos de que el paquete es innecesario, todo el proceso anterior se hará para que los dispositivos finalmente descarten el paquete.

Por lo tanto, la identificación del problema de esta tesis doctoral es la optimización de la monitorización de redes SDN para construir una herramienta que optimice el número de consultas estadísticas a los switches encargados del enrutamiento del tráfico de dicha red.

1.4. Resumen de las Contribuciones

Los resultados de la investigación realizada en esta tesis comprenden diversas contribuciones que han sido publicadas en revistas internacionales indexadas en el JCR y en congresos especializados del área. Estas contribuciones se enmarcan en el área de la monitorización en redes definidas por software. Como se representa en la Figura 1.1, estas contribuciones se enmarcan en los siguientes dominios de conocimiento: i) SDN, ii) QoE y QoS y iii) Monitorización.

En primer lugar, se presenta un algoritmo de encaminamiento para transmisiones multimedia en redes SDN cuya finalidad es mejorar la QoE al usuario que utiliza Floodlight como controlador y OpenFlow como protocolo de comunicación en la red SDN. Asimismo, a partir de esta contribución, se presenta un conjunto de tres técnicas de monitorización utilizando diferentes arquitecturas que permiten reducir el número de consultas estadísticas y, por tanto, reducir la carga del controlador de la red relativa al proceso de monitorización.

En los Capítulos 2 y 3 se hace una revisión exhaustiva de la literatura en el campo de la seguridad de las redes SDN [PFVCGV16] [PFGVK18b].

En el área de técnicas de monitorización se proponen tres arquitecturas que utilizan algoritmos de optimización las cuales monitorizan tres parámetros: la tasa de transmisión de datos, la tasa de error (pérdidas) y el número de consultas estadísticas a los switches de la topología. Dichos algoritmos han sido implementados y simulados en el controlador Floodlight utilizando el software de simulación Mininet con transmisiones de vídeo desde un servidor a un ordenador cliente. Todas estas contribuciones se especifican en el Capítulo 4 [PFGV17a] [PFGV17b] [PFGVK18a] [PFGV20].

Figura 1.1: Esquema de las Contribuciones

1.5. Estructura del Trabajo

Esta Tesis doctoral se estructura como sigue:

El Capítulo 2 describe las SDN, analizando su evolución en los últimos años, sus características junto con las oportunidades y retos que presenta dicha tecnología. Además, se describe la aplicación que tiene esta tecnología en diferentes ámbitos y su inclusión con otras tecnologías para resolver distintos inconvenientes.

El Capítulo 3 resume el estado del arte de las principales técnicas de monitorización en las SDN. Comienza con las principales aportaciones de monitorización acompañado de los trabajos que usan los valores de las monitorizaciones realizadas buscando optimizar la QoS y QoE. Además, se describe los principales problemas de seguridad en los diferentes planos de SDN con las soluciones propuestas hasta el momento. Por último, se revisa el estado de las redes de sensores.

El Capítulo 4 presenta las contribuciones de esta tesis doctoral. En él se presenta la descripción, implementación y pruebas de los algoritmos utilizados para optimizar la monitorización de SDN. Se presenta dos contribuciones principales que son la búsqueda de la calidad en la transmisión de vídeo y por otro lado las distintas técnicas propuestas para monitorizar redes SDN.

El Capítulo 5 reúne las principales conclusiones de este trabajo de investigación.

Por último, el Capítulo 5.1 expone posibles líneas futuras de investigación.

El Apéndice A presenta la arquitectura SDN denominada OpenFlow. OpenFlow es el estándar SDN más utilizado por la comunidad científica que ofrece un protocolo abierto de comunicación entre el controlador y el switch. Se muestran los elementos de la arquitectura OpenFlow, haciendo énfasis en el switch OpenFlow y en el protocolo OpenFlow como canal de comunicación entre el switch y el controlador.

El Apéndice B analiza el Sistema Operativo de Red, sus características, ventajas y las principales herramientas disponibles en la actualidad. En otras palabras, se analizan los principales tipos de controladores que actualmente se utilizan en las SDN. Se clasifican en grupos en cuanto al lenguaje en el que son implementados acompañados de las características propias de cada uno.

Capítulo 2

Redes Definidas por Software

El objetivo general de este capítulo es proporcionar una visión general de la arquitectura de las Redes Definidas por Software (SDN) y la evolución de las redes tradicionales hasta llegar a dicha arquitectura. Además, se proporciona la limitación existente en los planos de control y datos y cómo se han separado para llegar a las Redes (SDN). Del mismo modo, en este capítulo se lista y describe las diferentes aplicaciones de las resdes (SDN) en diferentes campos. Por otro lado, se describen otras arquitecturas como son las redes de sensores y redes celulares inalámbricas y cómo las redes (SDN) contribuyen a mitigar sus desafíos. Por último, se describen los retos y desafíos de esta de las redes (SDN).

2.1. Introducción

El nacimiento de nuevos servicios y aplicaciones on-line, tanto en terminales fijos como en dispositivos móviles han hecho de las redes de comunicación un punto estratégico, tanto en empresas, instituciones y hogares. La continua evolución de estos servicios y la creciente información que circula en internet han traído retos imprevistos a los desarrolladores y empresas. En especial, los nuevos dispositivos que, gracias a los avances en *Micro-Electro-Mechanical Systems* (MEMS), automáticamente guardan, procesan y envían información con datos relevantes relacionados con las actividades humanas a través de la red. Este tipo de dispositivos, principalmente constituidos por sensores y actuadores (*Radio Frequency IDentification* (RFID), dispositivos Bluetooth, redes de sensores, sistemas embebidos, ...) han dado lugar al nacimiento de nuevos conceptos y paradigmas como es el de *Internet of Things* (IoT).

En 2011 el número de dispositivos conectados en el planeta sobrepasó al número de habitantes. Actualmente, existen 9 billones de dispositivos conectados y se espera que para el año 2021 se alcance un número de dispositivos conectados generando tráfico que será tres veces la población global [GBMP13]. Estos dispositivos utilizan diferentes formas de conectarse a la red; entre otras, la infraestructura de red tradicional. Sin embargo,

los equipos y protocolos de red tradicionales no fueron diseñados para soportar un alto nivel de escalabilidad, alta cantidad de tráfico y movilidad. Las actuales arquitecturas resultan poco eficientes y presentan limitaciones importantes para satisfacer estos nuevos requerimientos.

La infraestructura encargada de transmitir la información procedente de dispositivos IoT (routers, switches, redes 3G-4G, puntos de acceso) tiene que adaptarse a nuevos servicios post-PC (VoIP, Virtualización, QoS, Computación en la Nube, Aplicaciones de IoT) y, al mismo tiempo, brindar seguridad, escalabilidad, rapidez y disponibilidad, entre otros. Algunos esfuerzos como SENSE [SEN17], *Internet of Things-Architecture* (IoT-A) [IoT17] o Cognitive Management Framework for IoT [VGS+13], así como nuevos protocolos como el *Data-Driven Routing Protocol* (DDRP) [SZMM13] han tratado de obtener una conectividad más inteligente entre los elementos de red. Sin embargo, es posible que no sean la mejor opción para cada uno dominios de aplicación y dispositivos en particular (Smart Grid, Intelligent Transportation, Smart Home, Health Care, Environmental Monitoring, ...). Por esta razón, en los últimos años ha surgido la idea de personalizar el comportamiento de la red y dar flexibilidad a los usuarios para utilizar los recursos de red según sus necesidades. Más aún, el desarrollo de algoritmos para la toma de decisiones en redes IoT requiere que diferentes métodos (algoritmos genéticos, redes neuronales, algoritmos evolutivos y otras técnicas de inteligencia artificial) puedan ser implementados rápidamente en los equipos de red de forma dinámica sin necesidad de esperar un estándar.

SDN es una arquitectura de red que elimina la rigidez presente en las redes tradicionales. Su estructura permite que el comportamiento de la red sea más flexible y adaptable a las necesidades de cada organización, campus o grupo de usuarios. Además, su diseño centralizado permite recopilar información importante de la red y usarla para mejorar y adaptar sus políticas dinámicamente. El desarrollo de SDN en los últimos años ha impulsado nuevos conceptos, como es el sistema operativo de red (Network Operating System, *Network Operating Systems* (NOS)), tratando de emular el avance que se ha tenido en sistemas de computación. Gracias a esta herramienta se ha logrado probar SDN en múltiples proyectos (Home Networking, Data Centers, Multimedia, entre otras iniciativas). De igual manera, SDN ha impulsado el diseño de modelos que finalmente integran y logran convergencia entre arquitecturas que tradicionalmente son independientes (WiFi – 4G – LTE). Sin embargo, todas estas oportunidades están aún lejanas de ser implementadas globalmente en equipos de producción. Temas importantes como la convergencia con redes actuales, escalabilidad, rendimiento, seguridad, etc., son retos importantes que deben superarse para ser posicionados en el mercado.

2.2. Evolución

La idea de transmitir información entre dos puntos a través de una red hizo necesario el diseño de protocolos de comunicación (*Transmission Control Protocol* (TCP)/*Internet*

Protocol (IP), *HyperText Transfer Protocol Secure* (HTTPS), *Domain Name Server* (DNS)) y la fabricación de equipos especializados en la transmisión de información. Dichos equipos han evolucionado dando lugar a una gran variedad de dispositivos (hub, switch, router, firewall, middlebox, ...). Esto ha causado un incremento exponencial en el número de dispositivos conectados.

Todos estos dispositivos encargados de transmitir información tienen características similares en su diseño y fabricación. En primer lugar, existe un hardware especializado en el tratamiento de paquetes (plano de datos) y, sobre el hardware, funciona un sistema operativo (generalmente Linux) que recibe la información del hardware y ejecuta una aplicación de software (plano de control). El software contiene miles de líneas de código y su objetivo es determinar el siguiente salto que debería tomar un paquete para llegar a su destino. El programa sigue las reglas definidas por un protocolo específico (actualmente existen unos 7000 *Request For Comments* (RFC)s) o alguna tecnología propia del fabricante. Los equipos modernos también analizan la información de los paquetes en búsqueda de información maliciosa o intrusiones (cortafuegos, sistemas de detección de intrusos). Sin embargo, todo el software o tecnología que se utiliza en la fabricación de estos dispositivos es rígido o simplemente cerrado para el administrador de red.

El administrador está limitado a configurar únicamente algunos parámetros, generalmente a través de comandos de bajo nivel usando una interfaz de comandos (*Command Line Interface* (CLI)). Por otro lado, cada nodo es un sistema autónomo que busca el siguiente salto que debe tomar un paquete para llegar a su destino. Algunos protocolos (*Open Shortest Path First* (OSPF), *Border Gateway Protocol* (BGP)) permiten que los nodos compartan información de control entre sí, pero únicamente con sus vecinos inmediatos y de manera muy limitada, con el fin de evitar carga adicional en el tráfico de red. Esto significa que no existe una visión global de la red como un todo. Si el administrador necesita controlar y modificar un camino determinado, el administrador tiene que jugar con parámetros, prioridades o utilizar artilugios para lograr el comportamiento esperado en la red. Cada cambio en la política de red requiere la configuración individual, ya sea directa o de forma remota de cada uno de los equipos. Esta rigidez hace muy complicada la implementación de políticas de red de alto nivel que sean adaptativas, es decir, que sean flexibles y reaccionen dinámicamente según las condiciones de la red.

Al igual que los sistemas operativos evolucionan y se adaptan a las nuevas necesidades y tendencias tecnológicas (soporte multi-CPU, multi-GPU, 3D, soporte pantalla táctil, entre otras), la adaptabilidad de la red a nuevos requerimientos (*Virtual Local Area Networks* (VLAN), IPv6, QoS, VoIP) se materializa por medio de protocolos o RFC. Sin embargo, a diferencia del sistema operativo que, gracias a su separación hardware, permite la continua actualización de aplicaciones o directamente su actualización completa, en el área de redes el período de diseño de una nueva idea hasta su publicación en un protocolo y posterior instalación en los equipos puede durar algunos años. Algunos servicios propietarios de los

fabricantes requieren que toda la infraestructura de la red sea de la misma firma para funcionar apropiadamente. Esta limitación favorece la dependencia de una tecnología o fabricante específica.

2.3. Separación de los Planos de Control y Datos

El concepto de SDN no es nuevo y completamente revolucionario, sino que más bien surge como el resultado de contribuciones, ideas y avances en la investigación en redes. En [ONF17] se determinan 3 estados importantes en la evolución de SDN: redes activas (de mediados de los 90 a principios de 2000), separación de los planos de datos y de control (2001-2007) y el *Application Programming Interface* (API) *OpenFlow* (OF) y NOS (2007-2010). Todos estos aspectos se analizan a continuación.

La dificultad de los investigadores para probar nuevas ideas en una infraestructura real y el tiempo, el esfuerzo y los recursos necesarios para estandarizar estas ideas en la *Internet Engineering Task Force* (IETF) hizo necesario dar cierta programabilidad a los dispositivos de red. Las redes activas proponen una interfaz programable o network API que abre al usuario los recursos individuales de cada nodo como procesamiento, recursos de memoria, procesamiento de paquetes y permitían incluir funcionalidades personalizadas a los paquetes que circulaban a través del nodo. La necesidad de utilizar diferentes modelos de programación en los nodos dio el primer paso para la investigación en virtualización de las redes, así como el desarrollo de frameworks o plataformas para el desarrollo de aplicaciones en el nodo. El framework de arquitectura de las redes activas v1.0 [ONF17] [Cal99] contiene un sistema operativo de nodo (Node Operating System, NodeOS) compartido, un grupo de ambientes de ejecución *Execution Environments* (EE) y aplicaciones activas (*Active Applications* (AA)s). The NodeOS administra los recursos compartidos, mientras que los EE definen a una máquina virtual para las operaciones de paquetes. Las AA operan dentro de un EE y brindan el servicio extremo a extremo. La separación de paquetes a cada EE depende de un patrón en la cabecera de los paquetes entrantes al nodo. Este modelo fue utilizado en la plataforma PlanetLab [Pla17], en donde los investigadores realizaban experimentos en ambientes virtuales de ejecución y los paquetes eran demultiplexados a cada ambiente virtual en función su cabecera. Estos avances resultaron importantes, especialmente en la investigación de arquitecturas, plataformas y modelos de programación en redes. Sin embargo, su aplicabilidad en la industria fue limitada y criticada principalmente por sus limitaciones en rendimiento y seguridad. El trabajo presentado en [WT01] es un esfuerzo para brindar un mayor rendimiento a las redes activas; el Secure Active Network Environment Architecture [AAKS98] intentó mejorar su seguridad.

El crecimiento exponencial de los volúmenes de tráfico que circulan por la red acarreó la necesidad de mejorar la gestión y de utilizar mejores funciones de administración como es el manejo de los caminos o enlaces que circulan en la red (ingeniería de tráfico), predicción

de tráfico, reacción y recuperación rápida a problemas en la red, entre otros. Sin embargo, el desarrollo de estas tecnologías se ha visto fuertemente limitado por la estrecha unión entre el hardware y software de los equipos de red. Además, el continuo incremento en las velocidades de enlace (backbones) hizo que todo el mecanismo de transmisión de paquetes (packet forwarding) fuese concentrado en el hardware, separando el control o la administración de red a una aplicación de software. Dichas aplicaciones funcionarían mejor en un servidor, ya que presenta mayores recursos de procesamiento y memoria que los disponibles en un solo dispositivo de red. En este sentido, el proyecto ForCES (Forwarding and Control Element Separation) [YDAG04] estandarizado por la IETF (RFC 3746) estableció una interfaz entre los planos de datos y de control en los nodos de red. El software SoftRouter [LNR+04] utilizaba esta interfaz para instalar forwarding tables en el plano de datos de los routers. Asimismo, el proyecto *Routing Control Platform* (RCP) [CCF+05] propuso un control lógico centralizado de la red. De esta manera se facilitaba la administración y se daba capacidad de innovación y programación de red. RCP tuvo una aplicabilidad inmediata, ya que aprovechó un protocolo de control existente, BGP, para instalar entradas en las tablas de encaminamiento de los routers.

Con la separación de los planos de datos y control se desarrollaron arquitecturas "clean-slate" como es el proyecto 4D [GHM+05a] o Ethane [CFP+07]. La arquitectura 4D propone una arquitectura de 4 capas según su funcionalidad: data plane, discovery plane, dissemination plane y decision plane. Por su parte, el proyecto Ethane [CFP+07] propone un sistema de control centralizado de enlaces para redes empresariales. Sin embargo, la necesidad de switches personalizados basados en Linux, OpenWrt, NetFPGA con soporte para el protocolo Ethane, hizo difícil su aplicabilidad. Actualmente, el protocolo OF [MAB+08] es el más utilizado en la comunidad científica y ha sido la base para la realización de diferentes proyectos. Empresas como Cisco también han presentado una propuesta de nueva arquitectura denominado Cisco Open Network Environment (Cisco ONE). Simplificando el análisis previo, el término SDN propone algunos cambios a las redes de hoy en día. En primer lugar, establece la separación o desacople de los planos de datos y de control, permitiendo su independiente evolución y desarrollo. En segundo lugar, propone que el plano de control sea lógicamente centralizado teniendo de esta manera una visión global de la red. Finalmente, se instauran interfaces abiertas entre los planos de control y de datos. Las diferencias entre estas arquitecturas se presentan en la Figura 2.1.

La programabilidad que ofrece SDN a la red puede compararse como las aplicaciones móviles que hoy en día son ejecutadas sobre un sistema operativo (Android, Windows Mobile). Dichas aplicaciones utilizan los recursos del móvil (*Global Positioning System* (GPS), acelerómetro, memoria) gracias al API que ofrece el sistema operativo. De la misma forma, el administrador de red gracias a las API disponibles (propietarias o abiertas) en el controlador, puede gestionar y programar los recursos de la red según las necesidades de los usuarios.

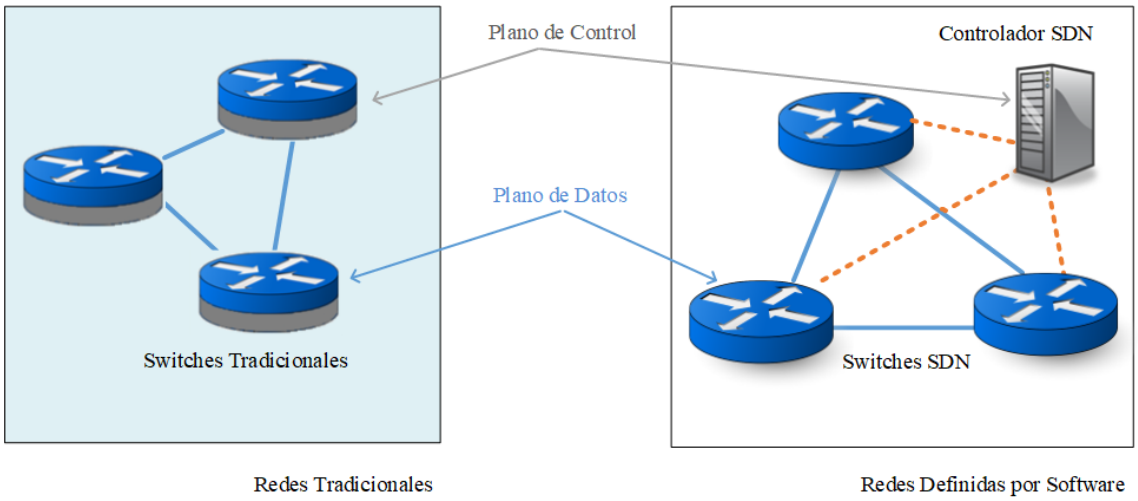


Figura 2.1: Comparación entre la Arquitectura Tradicional y Arquitectura SDN

2.4. Aplicabilidad

SDN brinda la capacidad de modificar el comportamiento de la red según las necesidades del usuario. Es decir, SDN por sí misma no resuelve ningún problema en concreto, sino que brinda una herramienta más flexible para gestionar de mejor manera las redes. Con el fin de probar las ventajas de esta arquitectura, la comunidad investigadora ha presentado múltiples proyectos de interés. A continuación, se resumen algunas de estas aplicaciones.

2.4.1. Redes Domésticas

En el incipiente campo de la IoT, la gestión de los dispositivos y los recursos de red en redes residenciales resulta todo un desafío debido al número de usuarios y dispositivos conectados a un mismo punto (usualmente, un punto de acceso). En [KSC+11] [VGS+13] se presenta una implementación de un sistema basado en OF que permite la monitorización y administración del acceso de usuarios a Internet basados en usage caps, es decir, una capacidad limitada de datos por usuario o dispositivo. El sistema permite visibilidad sobre los recursos de red, administración de acceso a nivel de usuario, grupo de usuarios, dispositivo, aplicación u hora del día e, incluso, el intercambio de capacidad de acceso con otro usuario. El control y monitorización de la red se realiza a través de una interfaz amigable de usuario Kermit y la administración de la capacidad y políticas de red por medio del language Resonance [NRFC09]. En este sentido, debido al bajo volumen de tráfico de las redes IoT, puede que la aplicación de soluciones SDN no sea totalmente necesaria y se tenga como posible alternativa.

2.4.2. Seguridad

La seguridad también puede ser mejorada debido a la visión global de la red. La seguridad no puede basarse únicamente en la seguridad de los hosts (antivirus), ya que cuando éstos se encuentran comprometidos dichas defensas no son efectivas. En [RMTF09] se presenta el sistema Pedigree como alternativa de seguridad en el tráfico que circula por la red corporativa. Este sistema, basado en OF, permite al controlador analizar y autorizar el tráfico y conexiones que circula en la red. Los hosts tienen un módulo de seguridad a nivel de kernel (tagger) que no se encuentra bajo el control del usuario. Este módulo etiqueta las conexiones que solicitan enviar información a través de la red (procesos, archivos, etc.). Dicha etiqueta se envía hacia el controlador (arbiter) al inicio de la comunicación. El controlador analiza y acepta o rechaza la conexión según sus políticas. Una vez que se autoriza la conexión, las tablas de flujo correspondientes se instalan en el switch. Pedigree presenta mayor resistencia a una variedad de ataques de evasión como los gusanos polimórficos. El sistema agrega una mayor carga al tráfico de red y al host. Sin embargo, esta carga no es mayor al de un software antivirus común.

2.4.3. Redes Móviles

Los dispositivos de la infraestructura de redes portadoras móviles (mobile carrier networks) comparten similares limitaciones que las redes de computadores. Las redes portadoras de igual forma siguen estándares y protocolos, por ejemplo, los propuestos por el Third Generation Partnership Project (3GPP), así como implementaciones propietarias específicas de los vendedores. En este punto el paradigma SDN y su modelo basado en flujos (flow-based forwarding model) puede aplicarse a este tipo de infraestructura ofreciendo mejores herramientas. *Software-Defined Mobile Network* (SDMN) [PWH13] es una arquitectura que permite a los operadores apertura, innovación y programabilidad sin depender de un fabricante exclusivo o proveedores de servicios *Over The ToP* (OTT). Este modelo consta de 2 elementos: *MobileFlow Forwarding Engine* (MFFE) y el *MobileFlow Controller* (MFC). MFFE es el plano de datos simple, estable y de alto desempeño. Tiene una estructura más compleja que un switch OF ya que soporta funcionalidades adicionales de portadoras como son la tunelización de capa 3 (por ejemplo GTP-U y GRE), funcionalidades de nodos de redes de acceso y de carga flexible. El MFC corresponde al plano de control de alta capacidad, en donde se desarrollan las aplicaciones de redes móviles. De igual manera, se establecen interfaces 3GPP para interconectarse con diferentes tipos de *Mobile Management Entity* (MME), *Serving Gateway* (SGW) o *Packet Data Network Gateway* (PGW).

2.4.4. Multimedia

Los múltiples servicios on-line multimedia como, por ejemplo, transmisión de contenido en tiempo real, requieren altos niveles de eficiencia y disponibilidad por parte de la infraestructura de red. Según estudios presentados por CISCO [Cis17], en el 2017 el 73 % de todo el tráfico IP (público y privado) era tráfico de vídeo IP (en 2012 era del 60 %). Además, en los últimos años ha tomado fuerza el término de QoE [LCMP12a], que intenta redefinir la QoS tomando en consideración el nivel de aceptación del usuario a un determinado servicio o aplicación multimedia. En este sentido, SDN permite optimizar las tareas de administración multimedia. Por ejemplo, en [KSKD+12] se mejora la experiencia QoE a través de la optimización de rutas. Esta arquitectura consiste de los elementos: el *Quality of Service Matching and Optimization Function* (QMOF) que lee los parámetros multimedia y determina la configuración apropiada para el enlace y el *Path Assignment Function* (PAF) que mantiene actualizada la topología de la red. En el caso de una degradación de la calidad en los enlaces, el sistema automáticamente modifica los parámetros de los enlaces tomando en cuenta las prioridades de los usuarios. Asimismo, el proyecto OpenFlow-assisted *QoE Fairness Framework* (QFF) [GEB+13] busca las transmisiones multimedia que se encuentran en la red y ajusta dinámicamente las características de la transmisión en función de los dispositivos terminales y los requerimientos de la red.

2.4.5. Confiabilidad y Recuperación

Uno de los problemas comunes en las redes tradicionales es la dificultad para recuperarse cuando falla un enlace. El tiempo de convergencia se ve afectado por la limitada información que posee el nodo para recalcular una ruta. En algunos casos, se requiere inevitablemente la intervención del administrador para que manualmente restablezca los enlaces en la red. En este punto SDN, gracias a su visión global, permite la personalización de los algoritmos de recuperación. En [SSC+12] se propone un sistema basado en OF que utiliza los mecanismos de restauración y protección para buscar un camino alternativo. En restauración el controlador busca otro camino cuando recibe la señal de caída de enlace. Por su lado, el método de protección se anticipa a un fallo y calcula previamente un camino alternativo. Por otro lado, al igual que el mal funcionamiento de un switch o router puede afectar gravemente la disponibilidad de la red, en SDN el mal funcionamiento del controlador (fallo del NOS, ataque *Distributed Denial of Service* (DDoS), error de la aplicación) puede ocasionar un colapso de toda la red. En este sentido, la confiabilidad de la red puede garantizarse por medio de controladores de respaldo (backup). Sin embargo, es necesario que tanto el controlador principal como el secundario tengan actualizada y coordinada la misma información de control y configuración. El componente CPRecovery [FBMP12] es un mecanismo de backup primario que permite la replicación de información entre el controlador principal y de respaldo. El sistema usa la fase de replicación para mantener actualizado el controlador backup y la fase de recuperación que inicia el

controlador de respaldo al momento de detectar un error en el controlador principal.

2.4.6. Virtualización

El concepto de virtualización en redes tiene similitud con virtualización en sistemas de cómputo, donde diferentes sistemas operativos pueden compartir recursos hardware, es decir, en virtualización de redes se intenta que múltiples redes virtuales puedan operar sobre una misma infraestructura, cada una con una topología y lógica de encaminamiento propia. Inicialmente, las tecnologías VLAN y redes privadas virtuales permiten que varios usuarios compartan recursos de la red. Sin embargo, la separación se controla sólo por el administrador de red con parámetros limitados (puerto del switch) y únicamente opera con protocolos de red conocidos. Con la separación de los planos de control y de datos que soporta SDN, las posibilidades de crear redes virtuales más avanzadas es prometedora. Por ejemplo, Flowvisor [SGY+09] es una plataforma de virtualización que utiliza OF [MAB+08] y se ubica lógicamente entre las capas de control y encaminamiento. Flowvisor [SGY+09] actúa como un proxy transparente entre los controladores y switches. Luego crea un plano virtual y transparente según las políticas establecidas por el administrador, asegurando aislamiento en términos de ancho de banda, flowspace y carga en el *Central Processing Unit* (CPU) del switch. El usuario puede observar y controlar únicamente su propio slice. Adicionalmente, es posible volver a dividir un slice virtual y tener de esta manera una jerarquía de redes virtualizadas, tal y como se muestra en la Figura 2.2.

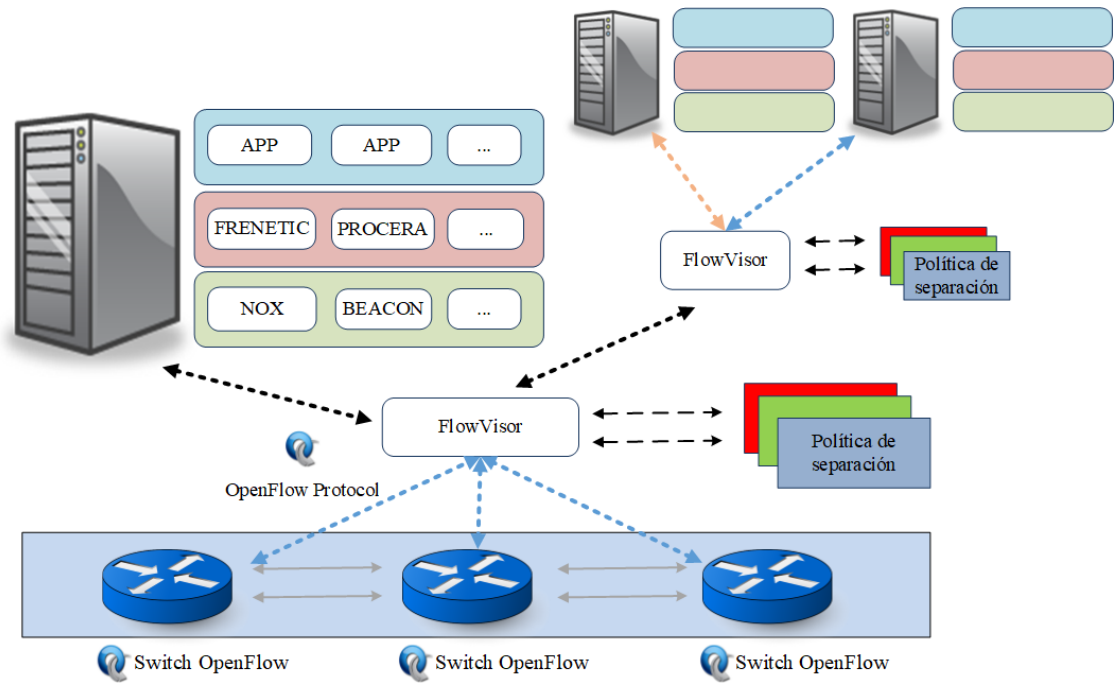


Figura 2.2: Protocolo OF, Virtualización y Sistemas Operativos de Red

En [SGY+10] hay una demostración de cuatro exitosos experimentos usando Flowvisor [SGY+09] (balanceador de carga, transmisión de vídeo streaming, ingeniería de tráfico y experimentos de hardware experimental), cada uno con su propio slice. Sin embargo, los experimentos muestran algunos problemas por resolver: interacción inesperada con otros dispositivos de red instalados, incremento del tráfico de broadcast emitidos por dispositivos no OF y algunas violaciones del aislamiento en CPU, especialmente cuando un slice añade una regla de encaminamiento que es enviado por el switch a través de un camino lento.

Otro aspecto importante es la integración entre las diferentes operaciones de red y virtualización de sistemas operativos *Operative System* (OS). En virtualización de OS, las diferentes máquinas virtuales *Virtual Machine* (VM)s requieren una capa de acceso de red que permita interconexión entre VMs y fuera de él y además soporte funciones de red comunes a una capa física tradicional. El modelo común es establecer comunicación entre nodos virtuales y el NIC físico implementando un típico encaminamiento de capa L2 (switching) o L3 (routing). Esto dificulta la administración de la red en ambientes virtuales, por ejemplo al momento de migrar VMs entre diferentes servidores físicos. En este enfoque, SDN y virtualización de redes puede ayudar a lograr estos objetivos.

Open vSwitch [PPA+09] es un switch basado en software diseñado para ambientes virtuales. Este switch exporta una interfaz para un minucioso control de la red. Adicionalmente, tiene una partición lógica para el plano de encaminamiento basado en un flexible motor de encaminamiento basado en tablas. El plano de encaminamiento tiene una interfaz externa y puede ser administrado por ejemplo a través de OF [MAB+08]. Con esta abstracción, el controlador puede obtener una vista lógica de múltiples Open vSwitches ejecutándose en servidores separados físicamente.

Otra aplicación interesante en virtualización es la *Virtual Network Migration* (VNM). En redes tradicionales, la migración o el cambio en un nodo de la red requiere la re-configuración y re-sincronización de los protocolos de encaminamiento. Esto causa altos retardos y pérdida de paquetes. En este ámbito, el uso de nodos virtuales puede reducir significativamente el tiempo de inactividad.

En el sistema VNM propuesto en [PFC+10], el controlador SDN crea nuevas entradas de flujo para el nuevo switch y redirecciona el camino del nodo inicial hacia el siguiente. Luego, el controlador elimina las entradas flow del switch antiguo permitiendo ser retirado con seguridad. Los resultados de experimentos muestran un tiempo total de migración de 5 ms sin aparente pérdida de paquetes. Más aún, el sistema podría ser reconfigurado dinámicamente para ubicar redes virtuales en diferentes nodos físicos según la hora del día o la demanda de tráfico para ahorrar energía (green networks). Este trabajo de investigación se diferencia del propuesto en la presente Tesis doctoral ya que analiza las ventajas y desventajas de aplicar diferentes tipos de migración de tráfico en base a la virtualización que utiliza.

2.4.7. Fiabilidad y Recuperación

Uno de los problemas más comunes en las redes tradicionales es el tiempo de recuperación ante un fallo de enlace. El tiempo de convergencia se ve afectado por la información limitada del nodo para recalcular la ruta. En algunos casos, es necesaria la intervención del administrador de la red para restablecer la ruta de los datos de red. En este punto, la visión global de SDN permite la personalización de algoritmos de recuperación. [SSC+12] propuso un sistema basado en OpenFlow que utiliza un mecanismo de restauración y protección para calcular un camino alternativo. En el mecanismo de restauración, el controlador busca una ruta alternativa cuando se recibe la señal de fallo. Mientras tanto, en cuanto a la protección, el sistema anticipa un fallo calcula previamente una ruta alternativa. De forma similar a un fallo en el switch o routers, el mal funcionamiento del controlador SDN (fallo del NOS, ataque DDoS y error de aplicación) pueden causar un colapso de toda la red. Por lo tanto, la fiabilidad de la red se puede garantizar a través de controladores de respaldo. Sin embargo, es necesario coordinar y actualizar la información de control y configuración entre el controlador principal y de respaldo. El componente CPRecovery [FBMP12] es un mecanismo de respaldo primario que permite la replicación de información entre el controlador primario y de respaldo. El sistema utiliza la fase de replicación para mantener la copia de respaldo del controlador y la fase de recuperación que inicia la copia de seguridad del controlador en el momento que detecta un fallo del controlador principal.

2.5. Redes de Sensores Inalámbricas

Esta sección revisa la arquitectura general de las *Wireless Sensor Network* (WSN) y sus características principales. Además, se presenta las aplicaciones de este tipo de redes en diferentes campos de la vida real y finalmente se describen algunos desafíos que se producen como consecuencia de esta tecnología.

Las WSN son generalmente, un conjunto distribuido de sensores autónomos que son usados para monitorizar condiciones físicas o ambientales como temperatura, sonido y humedad entre otras. Las WSN están compuestas de un conjunto que comprende desde pocos hasta un gran número de dispositivos llamados nodos los cuales contienen un transceptor de radio con una antena interna, un microcontrolador y una fuente de energía. Una WSN también está compuesta de una estación base que almacena los datos recogidos y un servidor de aplicaciones como se muestra en la Figura 2.3.

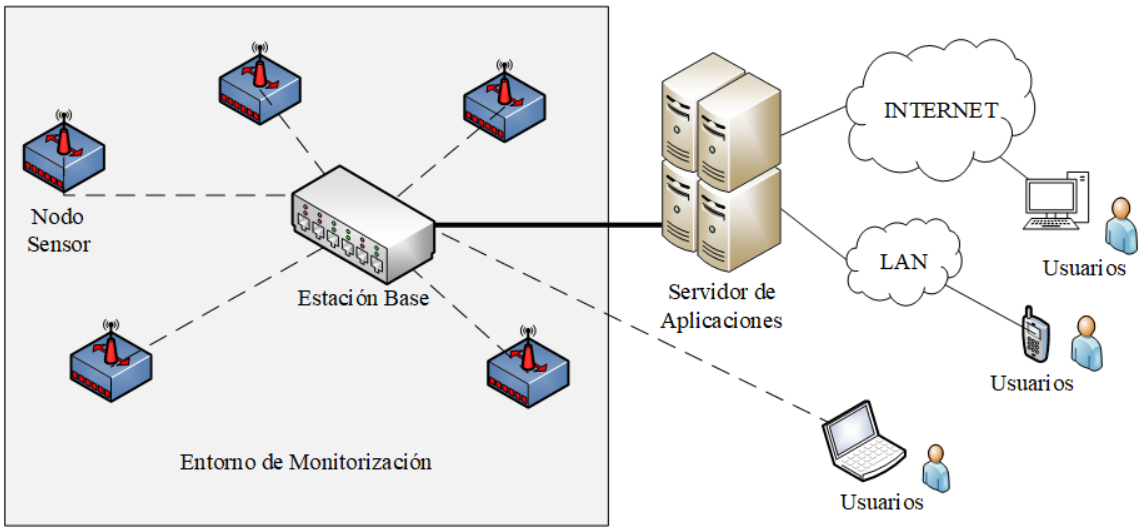


Figura 2.3: Arquitectura de las Wireless Sensor Network

Las WSN incluyen varias características que se enumeran a continuación:

- Tolerancia a fallos: los nodos tienen la capacidad de hacer frente a los fallos de los nodos.
- Heterogeneidad de nodos: los nodos no tienen que compartir la misma marca o las mismas características.
- Optimización de energía: restricciones de consumo de energía para nodos que usan baterías.
- Durabilidad: capacidad de resistencia en condiciones ambientales extremas.
- Escalabilidad: grandes implementaciones de nodos no interfieren en el rendimiento de la red.

Para probar estos tipos de redes, hay algunos simuladores disponibles para verificar su comportamiento. Por ejemplo, simuladores como OPNET [Riv], NetSim [Tet97] o NS-3 [NS306] son los más conocidos en la actualidad.

2.5.1. Aplicabilidad

Las WSN son muy útiles en varios campos de investigación, sobre todo en la informática y telecomunicaciones. Dichas redes proporcionan una gran aplicabilidad desde el punto de vista de la monitorización para obtener información importante de un sistema o entorno. En este sentido, la mayoría de los dispositivos IoT no se conectan directamente a redes IP ya que no implementan la arquitectura TCP-IP. La Figura 2.4 proporciona un árbol organizado de Aplicabilidad de las WSN.

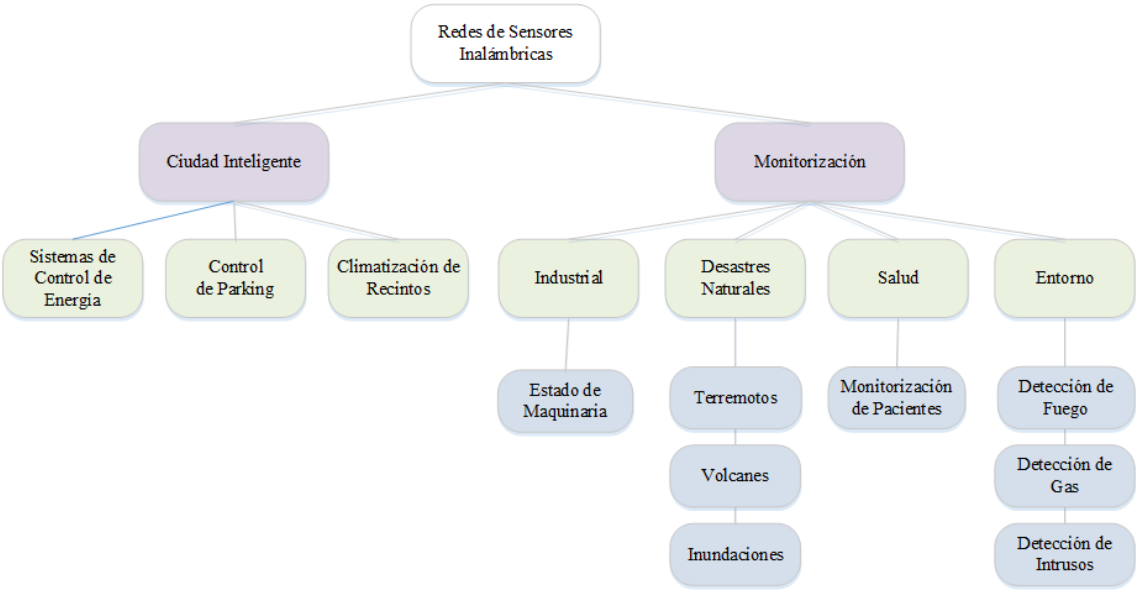


Figura 2.4: Aplicabilidad de las Wireless Sensor Network

La aplicación más común de las WSN es la monitorización de áreas, que consiste en el despliegue de sensores en un lugar para detectar algún fenómeno que quiera ser monitorizado. Un ejemplo de monitorización de áreas es el uso de este tipo de redes para detectar fuego, gas, intrusión enemiga entre otros.

Las WSN también se utilizan en redes inteligentes y sistemas de control de energía [LMZ+16] [SZ16] [YCEHH17] [BDC18] para cuantificar la energía que un usuario o una aplicación está utilizando y, al mismo tiempo, poder regular dicho consumo de energía. En este sentido, según determinadas condiciones, las WSN también son capaces de tomar decisiones como consecuencia de dichas condiciones. De esta manera, las WSN son extremadamente útiles en los edificios inteligentes para controlar el clima interior, control de plazas de aparcamiento, encendido de bombillas en pasillos, etc.

Este tipo de redes son el principio de una Ciudad Inteligente y la base de su funcionamiento. En una ciudad de este tipo, los sensores inteligentes están repartidos por toda la ciudad y proporcionan la información a las administraciones públicas o directamente al ciudadano a través de wifi y a tiempo real.[RR16].

Otra gran aplicación de estas redes se encuentra en el campo del cuidado de la salud [GGW+06] [LMFJ+04]. Las aplicaciones médicas pueden ayudar a los pacientes para detectar y tratar posibles enfermedades con tiempo suficiente para aplicar un tratamiento eficaz. Está dividido en dos tipos: los dispositivos implantados y los dispositivos portátiles. Respecto a los dispositivos implantados, están compuestos por artefactos que se introducen en el cuerpo humano. Además, monitorizan información humana como la tensión y niveles de azúcar entre otros. Hoy en día, la información sobre entrenamientos está ganando importancia ya que la cantidad de personas que practican deporte está aumentando. De

esta forma, otros tipos de redes de sensores han aparecido para ser aplicadas en la asistencia sanitaria, éstos son los dispositivos portátiles. Estos dispositivos se utilizan en la superficie del cuerpo humano o cerca de órganos vitales, para medir datos del cuerpo (número de pasos en un día, calorías consumidas, frecuencia cardíaca, etc..).

Además, otros campos en los que se aplican las redes de sensores son en la detección ambiental o en los movimientos de tierra. Hay varios ejemplos de este tipo de redes de sensores, pero las más importantes son las siguientes: en primer lugar, las WSN de monitorización de la contaminación del aire se implementan en algunas ciudades para medir la concentración de gases peligrosos para la población. Estos gases peligrosos provienen de automóviles y fábricas entre otras fuentes. La ventaja de los sensores inalámbricos con respecto a los sensores cableados es la movilidad para probar los datos leyéndolos en diferentes lugares o zonas de la ciudad. En segundo lugar, una WSN se puede configurar en un bosque para detectar si se ha iniciado un incendio. En este caso, los nodos de red se han desplegado con sensores para medir el porcentaje de gases, temperatura y humedad de dicho bosque. La ventaja en este campo es la capacidad para detectar prematuramente un incendio y cómo se está expandiendo a través del bosque. En tercer lugar, una WSN es muy útil para detectar movimientos leves en el suelo. Los sensores dentro de los nodos de la red pueden recopilar datos, hecho que es muy importante para obtener un patrón o una ocurrencia de deslizamientos de tierra. La ventaja en esta aplicación es la reducción en el tiempo de reacción antes de que ocurra un movimiento del suelo. Siguiendo este tema, la WSN se puede usar eficazmente en la prevención de desastres naturales como inundaciones [CEQM+04], huracanes, erupciones de volcanes, etc. De esta manera, Werner et al. [WALR+06] proponen una WSN para estudiar señales sísmicas en un volcán. El despliegue propuesto consiste en 16 redes de sensores con un sismómetro y un micrófono que recopilan datos sísmicos y acústicos del volcán. Estos sensores envían los datos recogidos a un nodo pasarela que también se encuentra cerca del volcán y, después de eso, transmite los datos a través de radio de onda libre a un observatorio. En cuarto lugar, un factor en el que se involucra directamente al ser humano es en la calidad del agua. Las WSN se despliegan en océanos y ríos, así como reservas de aguas subterráneas. El uso de redes de sensores proporciona un estado preciso del estado del agua para usarlo como entrada en un sistema que determina si el agua es potable.

Otra aplicación importante de las WSN es la monitorización industrial. Para empezar, la preservación del estado de las máquinas es una tarea importante para la cual una WSN puede ayudar a recopilar datos sobre el estado de tal maquinaria. Ofrece un importante ahorro de coste y permite nuevas funcionalidades debido a su capacidad de ser configurado en lugares difíciles o inaccesibles en los que una red cableada no podría ofrecer. Por otra parte, las WSN se utilizan para recopilar información ambiental, como la temperatura de un frigorífico y el nivel de agua en tanques de desbordamiento en centrales nucleares, entre otros. A continuación, la monitorización de la calidad y el nivel de agua incluye algunas tareas, como verificar la calidad de las aguas subterráneas o superficiales en para

proteger el desperdicio de agua. Finalmente, una WSN se puede usar para monitorizar las condiciones de infraestructura civil y procesos geofísicos en tiempo real.

Las transmisiones multimedia en WSN incluyen enormes requisitos de ancho de banda, consumo de energía, QoS, tratamiento de datos recopilados y protocolos de enrutamiento [GVSOTCBA09], entre otros. Cuando un usuario requiere que se envíe un vídeo, se necesita un ancho de banda elevado durante la transmisión. Consecuentemente, este elevado requisito se traduce en un incremento de consumo energético. Algunos factores como el retraso y la capacidad del canal, afecta directamente la QoS de la red y, por lo tanto, la entrega el vídeo debe alcanzar un nivel mínimo para garantizar una buena recepción del vídeo. Debido a esto, la transmisión de vídeo en las WSN no es adecuada en las tecnologías de transmisión usadas actualmente.

2.5.2. Desafios

Las WSN presentan algunos desafíos generales [CK03] en términos de gestión de sensores y unificación en el procesamiento de datos. Uno de los principales problemas de WSN es el descubrimiento de redes, ya que la topología de red tiene que ser construida en tiempo real y debe actualizarse permanentemente en el caso que se agreguen nuevos sensores o se produzcan fallos en los sensores de la topología. Otro desafío es el control de la red ya que cada nodo de la topología cambia dinámicamente en términos de energía, ancho de banda, etc. El enrutamiento también es un aspecto importante de WSN en contraste con el enrutamiento IP, ya que los caminos de enrutamiento se establecen a partir de información (por ejemplo, datos extraídos de geoinformación) en función de las necesidades. Además, las señales colaborativas y el procesamiento de la información requieren la comunicación entre nodos, y al mismo tiempo, que no pierda ninguna señal durante la transmisión. De esta forma, las WSN proporcionan métodos para consultar los datos recogidos de la red. Finalmente, el reto más importante es el de la seguridad, ya que la red se puede desplegar en un entorno hostil y, por lo tanto, estaría expuesta a ataques físicos.

Como se ha descrito anteriormente, varios estudios han planteado y discutido los aspectos sobre las WSN. Baronti et al. [BPC+07] proporcionan una revisión exhaustiva sobre los desarrollos y desafíos más recientes que las WSN deben superar planteando algunas soluciones para mitigarlos. En particular, [BPC+07] también se ocupa del estándar IEEE 802.15.4 en las WSN, el cual ha dejado de ser usado actualmente. Dicho estándar define las características de las capas física y de control de acceso al medio (MAC) en una red inalámbrica de área personal de corto alcance proveyendo datos confiables, operaciones de corto alcance y una duración razonable de la vida de la batería mientras se mantiene un apilado de protocolos flexibles. Se deben cumplir cuatro requisitos clave para un despliegue de WSN acreditable en entornos industriales: eficiencia energética, escalabilidad, fiabilidad y la puntualidad. El primer requisito, eficiencia energética, es extremadamente importante

ya que los nodos de los sensores suelen ser alimentados por baterías que algunos de ellas no pueden ser reemplazadas o recargadas debido a las circunstancias ambientales. En este caso, el estándar 802.15.4 incluye un mecanismo de administración de energía, basado en el ciclo de trabajo, para minimizar la actividad de los nodos sensoriales. La segunda característica, la escalabilidad, es otro factor importante que tiene que ser considerado en una WSN ya que el número de nodos de sensores desplegados puede ser alto, especialmente cuando los grandes despliegues cubren grandes áreas geográficas. Finalmente, la fiabilidad y la puntualidad son factores muy críticos en el ámbito industrial. Si un conjunto dado de paquetes de datos no se entrega correctamente en el destino, el comportamiento correcto del sistema puede verse comprometido. En estos términos, el estándar IEEE 802.15.4 proporciona una confiabilidad muy baja en términos de relación de entrega de paquetes cuando la administración de energía está habilitada. Este comportamiento es causado por el protocolo 802.15.4 MAC y, por lo tanto, nos referiremos a esto como un problema de la fiabilidad 802.15.4 MAC. Anastasi et al. [ACDF10] proponen una solución para este problema aprovechando la flexibilidad en la elección del acceso múltiple en la detección de portadora / en los parámetros de prevención de colisiones (CSMA / CA) que ofrece el estándar IEEE 802.15.4.

Perrig et al. [PSW04] resumen los principales desafíos y problemas de seguridad en las WSN. Se afirma que un sistema seguro es el que cada uno de sus componentes es seguro ya que los que no proporcionan seguridad se consideran como puntos de ataque potenciales. Además, se afirma que un sistema seguro tiene que cumplir con algunas características tales como establecimiento de claves, configuración de confianza, autenticación, privacidad, robustez en la comunicación, enrutamiento seguro y resistencia a la captura de nodos. Respecto al establecimiento de claves, la solución más sencilla es el establecimiento de una clave compartida en toda la red, pero, desafortunadamente, todo el tráfico de la red puede ser descifrado si la clave es interceptada. La opción más utilizada para proporcionar seguridad entre nodos es la criptografía de clave pública. Con el fin de proporcionar protección contra los atacantes que quieran inyectar códigos maliciosos o escuchar la comunicación, las WSN deben proporcionar autenticación mediante criptografía. Usándolo, los emisores y los receptores se identifican en la red como entidades confiables. Hoy en día, la privacidad es el factor más importante en las comunicaciones. El riesgo de ser espiado es elevado ya que existen muchas razones para hacerlo, pero, al mismo tiempo, un conjunto de reglas y leyes sociales no permiten que esto suceda. Además, la solidez de la comunicación puede afectar el rendimiento de la red a través de ataques de denegación de servicio [WS02]. Este reto aborda el empleo de comunicaciones de amplio espectro. El enrutamiento seguro también es un servicio esencial de WSN, ya que los protocolos de enrutamiento existentes experimentan vulnerabilidades debido a la alta sensibilidad de capturar de nodos de la red [KW03]. Consecuentemente, la resistencia a la captura de nodos es también un reto para mantener la seguridad de la red. Las soluciones a estos problemas son métodos que dejen evidencia de manipulación

incluidos en los nodos sensoriales, recopilando datos del entorno y verificándolos de forma cruzada para entre otros.

Las WSN abren una amplia gama de posibilidades en términos de creación de nuevas aplicaciones, creación o mejora de protocolos, introduciendo nuevos conceptos de diseño y desarrollando nuevos algoritmos. Ye et al. [YHE02] presentan un nuevo diseño de protocolo que trata de reducir el consumo de energía, pero al mismo tiempo, proporciona alta escalabilidad evitando colisiones. Este protocolo está compuesto por tres componentes principales entre otros: un componente que está escuchando y durmiéndose periódicamente, un componente para prevenir colisiones y escuchas, y, finalmente, un componente para enviar mensajes. La escucha periódica y el componente durmiente intentan disminuir el tiempo de escucha a un valor periódico. Entonces, si el tiempo de escucha se reduce a la mitad, la energía ahorrada también se reduce a la mitad. Para prevenir la colisión y el exceso de audiencia, se utiliza el campo de duración dentro de cada paquete, que indica el tiempo restante para la próxima transmisión y, entonces, evita la colisión de paquetes. El manejo de mensajes propuesto divide el mensaje en pequeños fragmentos que serán enviados en ráfaga. Finalmente, los ataques físicos relacionados con la captura de nodos de red también son un gran problema. En contraste con las redes IP tradicionales que eran inaccesibles para los posibles atacantes, las WSN se colocan en entornos que no están totalmente garantizados.

2.6. Redes de Sensores Definidas por Software

El concepto de Redes de Sensores Definidas por Software SDSN [KAMH17] [NHAM17] [MLMAM18] ha aparecido debido a la necesidad y la demanda en redes específicas de aplicaciones y también por la disminución en el precio de los sensores. Las SDSN consisten en nodos sensores (descritos en la Sección 3), incluyendo la función de cambio dinámico de acuerdo con las necesidades de las aplicaciones de software (igual que SDN). Esta sección describe la arquitectura SDSN y los principal retos que presenta esta tecnología.

2.6.1. Arquitectura

Físicamente, las arquitecturas SDN y SDSN son similares en términos de gestión ya que la primeras están controlados por un controlador y las segundas por un servidor de control de sensores. El procedimiento de despliegue de una nueva tarea o aplicación es a través de la reprogramación de nodos sensores con el código que un usuario necesita ejecutar dentro de la red. Además de la alta aplicabilidad de WSN (detección de temperatura, movimiento sísmico, medición de la contaminación, etc.), SDSN incorpora un mecanismo para cambiar la configuración de la red con roles. El Servidor de Control de Sensores *Sensor Control Server* (SCS) es responsable de configurar el valor del rol actual. Por otro lado, la entidad llamada compilador de escenarios (también situado en el servidor) genera

el rol del programa de acuerdo con la aplicación que se utilizará [CEQM+04] [WALR+06], y después de eso, será enviado a los sensores a través de la comunicación inalámbrica.

Lógicamente, las SDSN se basan en tres capas definidas de forma software para cumplir su funcionalidad como muestra la Figura 2.5. La capa física (plano de datos) está compuesta por nodos sensores y además contiene la *Software Defined Radio* (SDR) que se utiliza para controlar el acceso al medio. La capa de red (plano de control) es responsable de la transmisión de datos a través de la red y también el concepto SDN (controlador) se encuentra dentro de él. Finalmente, la capa de aplicación gestiona los roles de los programas, tareas de detección y el sistema operativo que controla los nodos sensores.

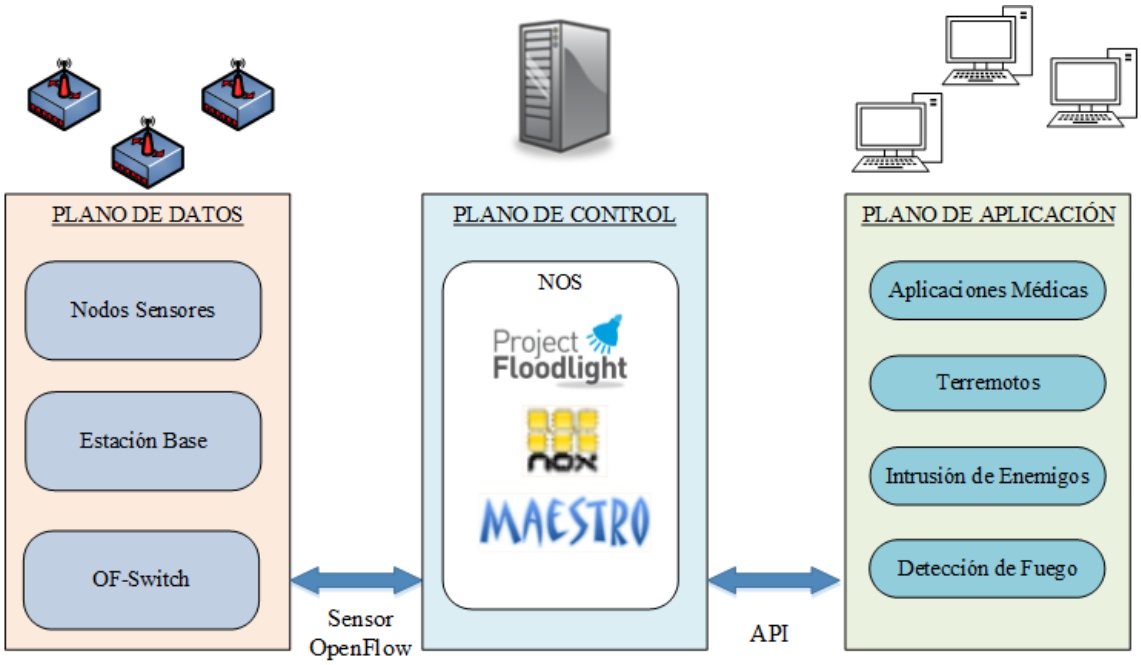


Figura 2.5: Arquitectura SDSN

2.6.2. Desafíos

Por lo tanto, SDSN obtiene las mejores características de la unión de WSN y SDN [MM18] debido al diseño del Sensor OpenFlow [LTQ12]. Una de las ventajas logradas es la versatilidad, ya que son capaces de soportar varias aplicaciones sin instalar ningún software adicional (plug-and-play). Otro beneficio es la flexibilidad en términos de cambios rápidos en el comportamiento de la red debido al control centralizado. Además, la facilidad para gestionar la red a través de una Interfaz de programación de aplicaciones API abierta es un factor muy importante en este tipo de redes.

Las SDSN han inspirado varias aplicaciones. Zeng et al. [ZMG+13] proponen el

proyecto *Demand-Addressable Sensor Network* (DASN). El propósito de este proyecto es construir una red de sensores que pueda realizar tareas de detección en función de los requisitos del usuario. Estas demandas son buscadas dentro de los nodos sensores y, si se satisfacen las solicitudes, dichas solicitudes se mezclan con más datos en la red y finalmente se muestran al usuario a través de un terminal.

Los SDSN presentan algunos desafíos técnicos [LTQ12] en que la comunidad de investigación trata de resolver proponiendo varias soluciones. El primer problema pertenece a los flujos de datos (plano de datos), que se componen de paquetes y su direccionamiento. A diferencia de las SDN, las WSN no utilizan direcciones IP, sino que emplean direccionamiento a través de atributos (por ejemplo, nodos sensores en N atmósferas). El problema se resuelve incluyendo direccionamiento IP (IPv4 [P+81] e IPv6 [DH98]) dentro de las WSN. De la misma manera que en el anterior problema, el canal OF [PLHea11] del sensor proporciona conectividad TCP/ IP en el envío de datos, y, como se indicó anteriormente, las WSN no incorporan este direccionamiento. Un enfoque para dar una solución a tal problema es el uso de protocolos de transporte a través de WSN por lo que el tráfico se entregará en su destino. Además, dado que las WSN son sensibles a cada cambio en el entorno, la sobrecarga del tráfico de paquetes control en la red supone un gran problema. Desarrollar un algoritmo que solo envíe el primer paquete cuando se produce un fallo en la tabla de rutas, reduce la sobrecarga de la red hasta que llega el mensaje de modificación de flujo. Finalmente, la generación de tráfico es el problema más importante en las WSN, ya que los paquetes de datos que se envían desde los nodos sensores aumentan el tráfico a través de la red. Añadiendo un módulo que genere tráfico en la red que utilice métodos síncronos, asíncronos y round-robin propone una solución para controlar la cantidad de tráfico en la red.

Mahmud et al. [MR11] proponen un enfoque para utilizar la tecnología OF dentro de una WSN. Además, exponen cuatro ideas a desarrollar en redes de sensores para lograr un incremento en confiabilidad: en primer lugar, presentan el concepto de sensor de flujo en lugar del sensor típico que se está utilizando actualmente. En segundo lugar, el uso del protocolo OF en la comunicación entre los sensores de flujo y el controlador. En tercer lugar, el flujo de tráfico que pertenece a los mensajes de control (plano de control) no interfiere en el tráfico de datos (plano de datos). Por último, se pretende lograr un enrutamiento infalible en ambos sentidos de tráfico. Su principal característica es el procedimiento de enrutamiento para reenviar el tráfico de datos que se describe a continuación. Asigna un valor (coste) para todas las rutas disponibles en la red y, después, guarda cada ruta calculada en una estructura de árbol. Una vez que se ha calculado el árbol de rutas, el optimizador de rutas elige la mejor ruta en términos del número de saltos que está compuesta. Concretamente, elegirá como la mejor ruta la que tenga el mínimo número de saltos. Sin embargo, si se produce un problema en un nodo sensor y se vuelve inactivo, el optimizador de ruta elegirá la segunda mejor ruta siguiendo los mismos parámetros de elección.

Otro desafío es la asignación de recursos y su gestión. De esta manera, Costanzo et al. [CGMP12] proponen algunas ideas para administrar los recursos de red de manera eficiente: la agregación de datos solo cuando sea necesario, flexibilidad en las rutas de enrutamiento y ahorro de energía en los nodos sensores cuando la radio no está funcionando. Para obtener estas características, esta arquitectura está dividida en capas incluyendo: una capa de adaptación (formateo de paquetes), capa de virtualización (parte de la red según los términos de la topología) y un controlador (que trabaja con reglas de la tabla de flujo) [KAMH18]. Dado que hay agregaciones de datos obligatorias como los mensajes de intercambio entre los sensores y el controlador cuando se crea la topología de la red, no está claro si esta arquitectura optimiza el uso de energía en los nodos sensores [XWZ16].

Galluccio et al. proponen SDN-WISE [GMMP15b] [GMMP15a], una extensión de Sensor OpenFlow [LTQ12]. Utiliza SDN con el fin de simplificar las políticas de reconfigurar las WSN que se componen de nodos sensores de proveedores distintos. SDN-WISE se basa en varios controladores (controladores locales) y uno adicional (controlador global) que funciona como un proxy entre el plano de datos y ellos. Su principal característica es los sensores solo participan en tareas de control local, eliminando la interacción con el controlador global. Por lo tanto, un paquete puede ser tratado de diferentes maneras, ya que las diferentes reglas de flujo se programan dentro de los controladores.

Continuando en esta línea, Gante et al. [DGAM14] proponen una solución para incrementar la gestión de las WSN. El controlador SDN se compone de una pila de cinco capas: las tres capas inferiores son la capa física, la capa *Medium Access Control* (MAC) y finalmente la capa NOS. El controlador se coloca en la cuarta capa, y la quinta es la capa de aplicación donde se gestionan tareas tales como enrutamiento, QoS y localización. Los autores afirman que esta arquitectura en capas y centralizada logra un mejor rendimiento en las tareas de administración, y también ofrece un ahorro de energía en los nodos sensores proveniente de la liberación de la carga de comunicación.

Como se indicó en la sección anterior, la unión entre las arquitecturas WSN y SDN permite la multitarea en WSN. De esta manera, Zeng et al. [ZLG+15] [ZMG+13] proponen una arquitectura compuesta por WSN y SDN que permite la multitarea compartiendo los mismos recursos de red. Por lo tanto, cada sensor contiene varios programas que se ejecutan dentro de ellos, los cuales pertenecen a las aplicaciones basadas en los requisitos del usuario. Con el fin de soportar la multitarea de una manera eficiente, ofrece una optimización (en términos de carga y energía) en todas las aplicaciones administradas por un controlador global con un proceso de programación de éstas. Este proceso de optimización se define como un problema de energía mínima de activación del sensor que tiene en cuenta las restricciones de almacenamiento y cobertura. La Tabla 2.1 reúne las soluciones basadas en SDN para las WSN y sus características. Las cuatro columnas del lado derecho describen las características de cada solución. La columna llamada OF refleja si el modelo propuesto es compatible con el protocolo OF. A continuación, la siguiente

columna, llamada Agregación de Datos, determina si la propuesta agrega datos a la red. La columna Gestión muestra si la solución gestiona los recursos de red de manera eficiente. Finalmente, la última columna señalada como Asignación de Recursos indica si la solución propuesta puede asignar recursos de red.

Tabla 2.1: Diferencias entre Software Defined Sensor Networks

Referencia	Propuesta	OpenFlow	Agregación de Datos	Gestión	Asignación de Recursos
[LTQ12]	Sensor OpenFlow	✓	✓	✓	X
[ZMG+13]	Multitasking SDN	X	X	✓	✓
[MR11]	Explotation OpenFlow	✓	✓	✓	X
[CGMP12]	SDWN	✓	X	✓	✓
[GMMP15b]	SDN-WISE	✓	✓	✓	✓
[DGAM14]	Smart WSN	X	X	✓	X

2.7. Redes Celulares Inalámbricas

En esta sección, se presenta una descripción general de las WCN y cómo se pueden añadir las SDN a esta arquitectura para resolver los desafíos existentes. Dichos desafíos son descritos bajo diferentes categorías y también las soluciones propuestas para ellos. Finalmente, esta sección concluye con una tabla resumen donde se comparan las soluciones propuestas descritas en esta sección.

2.7.1. Arquitectura

Las WCN están compuestas por teléfonos y dispositivos inteligentes que están conectados a Internet a través de estaciones base que están instaladas en torres celulares. Al mismo tiempo, las estaciones bases se clasifican como puerta de enlace de servicio y puerta de enlace de red de datos. Estas puertas de enlace funcionan como plano de control y como plano de datos. Por un lado, el plano de control incluye establecimiento de la conexión móvil, Enrutamiento, gestión de movilidad, QoS [LMR12] y QoE. Por otro lado, el plano de datos se compone de control de acceso y monitorización de tráfico. Sin embargo, la estrecha unión de ambos planos abre desafíos en términos de gestión de recursos y de escalabilidad en las WCN [WLY+17].

La arquitectura de WCN se compone de varios componentes; los dos componentes principales son *Radio Access Network* (RAN) y el *Core Network* (CN). La RAN consiste en torres celulares situadas de manera cercana que proporcionan el acceso a los dispositivos del usuario. Además, esta tecnología ofrece funcionalidades como la asignación de recursos [WYL+17] y la gestión esencial. Por su parte, el CN proporciona conectividad entre diferentes RAN y también servicios como el establecimiento de la conexión y

mantenimiento, entre otros.

2.7.2. Desafíos

Debido a la demanda actual de las WCN, la transferencia de datos ha aumentado, resultando en un gran impacto para la experiencia móvil del usuario [Cis11] y para los recursos. Entonces, este hecho abre un gran desafío en términos de gestión eficaz de los recursos. Se proponen varias técnicas para optimizar la asignación de recursos para lograr un mejor desempeño de las WCN. En [AATK13] [DMW+11] se proponen estrategias como el control de potencia, división de celdas y una sofisticada asignación de canales. Dependiendo de cómo se lleven a cabo estas estrategias, surgen otros desafíos: adecuar QoS y QoE para el tráfico de voz y vídeo.

2.7.3. Redes Definidas Software Aplicadas a las Redes Celulares Inalámbricas

La aplicación de SDN en WCN elimina la carga de la puerta de enlace de la red de paquetes ya que desacoplando los planos de datos y control permiten la escalabilidad y la capacidad de gestión de la red. Como consecuencia, se reducen los componentes hardware y el coste del hardware. Por lo tanto, tener un controlador centralizado mejora el rendimiento y la estabilidad de la red, logrando la asignación global de recursos y gestión de interferencias. El diseño de las *Software Defined Wireless Cellular Network* (SDWCN) se compone del plano de control (conjunto de módulos de aplicación que proporcionan balanceo de carga, QoS, gestión de movilidad y establecimiento de conexión) y el plano de datos (estaciones base y torres celulares), como se muestra en la Figura 2.6.

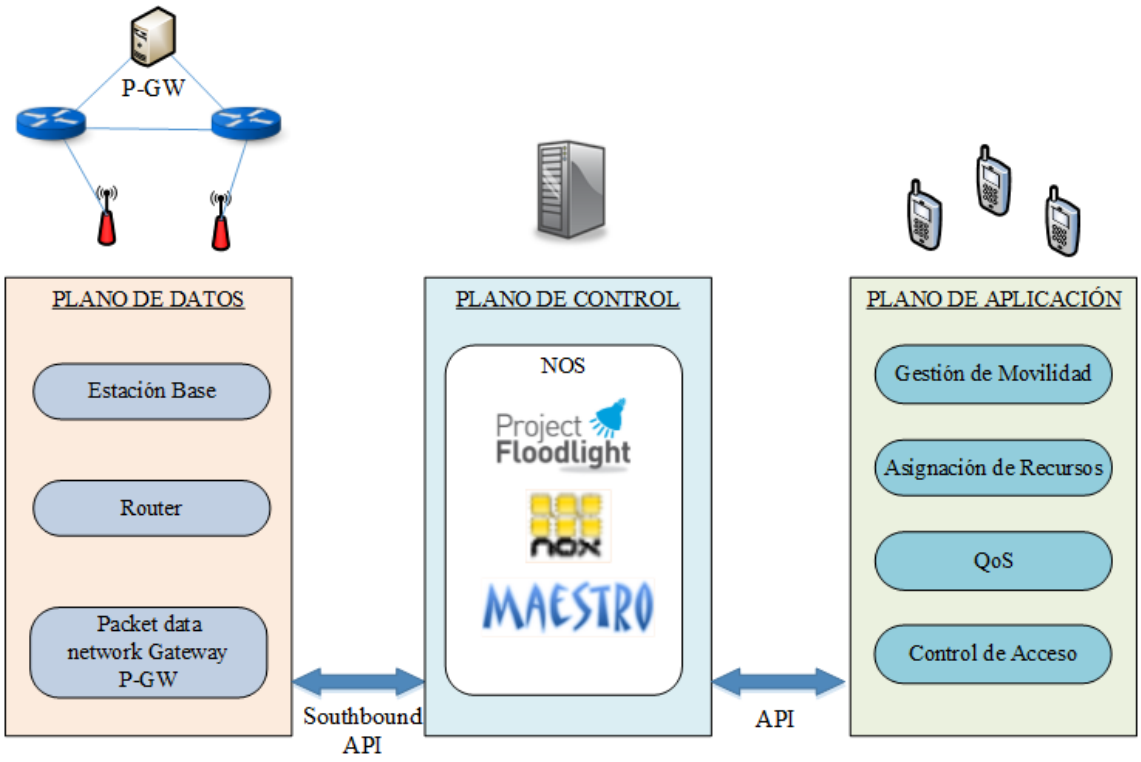


Figura 2.6: Arquitectura SDWCN

La implementación física de WCN se basa en conjuntos de celdas para proporcionar servicios de comunicación a los usuarios de la red. Por lo tanto, los teléfonos deben estar dentro de la cobertura del punto de acceso para estar conectados en tales redes. Por lo tanto, las WCN utilizan conjuntos de celdas para aumentar el rendimiento de la red en términos de las demandas de tráfico de los usuarios. En particular, las celdas se dividen en otras más pequeñas que son administradas por los puntos de acceso. Debido a esta división, las interferencias entre las celdas limitan eventualmente la capacidad de la red produciendo celdas solapadas. Por lo tanto, el paradigma de división de celdas (número de celdas y su tamaño) plantea problemas difíciles, como la gestión de la movilidad y el equilibrio de carga en estas redes [GPLK13].

Trabajos como [LMR12] utilizan el modelo SDN en las WCN para asignar recursos de la mejor forma posible . Se han dividido estas soluciones en tres grupos diferentes atendiendo al componente de la WCN que se está mejorando. El primer grupo pertenece a trabajos centrados en las RAN, el segundo grupo está dirigido al CN y, finalmente, las soluciones que utilizan la asignación de recursos a través de RAN y CN al mismo tiempo:

Respecto a la asignación de recursos en RAN, trabajos como SoftRAN [GPLK13] proponen un plano de control de acceso centralizado de radio para abstraer un grupo de estaciones base en una sola estación base virtual. Construye una red tridimensional

de recursos basada en la abstracción de espacio, tiempo y frecuencia. Sin embargo, este diseño tiene en cuenta el retraso de la red incluso cuando la asignación de recursos se realiza rápidamente. Por lo tanto, el control lógico centralizado es responsable de aspectos de control de la red global, mientras que los dispositivos del plano de datos son manejados por un proceso de distribución de tareas. Además, SoftRAN es capaz de mejorar el proceso de transición de una base de radio a otra y la asignación de recursos basado en interferencias. También, asigna energía a los recursos que se colocan en una estación base en una WCN densa. Chen et al. proponen SoftMobile [CZCT14], un enfoque que proporciona una distribución de tareas alternativas para resolver tres subproblemas: cómo puede distribuirse la información de estado, cómo pueden ser configuradas las celdas de manera coherente y, finalmente, cómo se pueden manejar las operaciones entre celdas en tiempo real. Cada uno de estos problemas es manejado por tres controladores separados.

Existe una necesidad de asignación de recursos en el CN donde la RAN conecta los dispositivos a internet. Las redes celulares tradicionales centralizan esta asignación de recursos en la estación base. Las puertas de enlace de la red de paquetes datos son responsables de verificar la disponibilidad de los recursos para satisfacer una demanda de éstos con prioridad que pertenezca a un usuario o una aplicación o realizar las acciones correspondientes (negando la solicitud o liberando el recurso). Este modelo plantea problemas de escalabilidad que requieren puertas de enlace de red de paquetes de datos costosas para satisfacer este desafío. De esta manera, SoftCell [JLVR13] propone una solución para eliminar la complejidad de las puertas de enlace de la red de datos de paquetes para acceder a switches colocados en las estaciones base. Sin embargo, esta solución presenta importantes requisitos de ancho de banda que se traducen en congestión y también en una limitación de la escalabilidad de la red. Para satisfacer esta materia, SoftCell presenta un algoritmo que agrega paquetes multidimensionales para poder disminuir el tamaño de la tabla de reenvío en los dispositivos del plano de datos.

Con respecto a las soluciones que utilizan RAN y CN programables, el desafío de la asignación de recursos está presente tanto en RAN como en el CN. SoftAir [AWL15] es una arquitectura de red inalámbrica para 5G basada en el modelo SDN. Su plano de control consta de dos componentes: herramientas de gestión de red y aplicaciones personalizadas que brindan servicios a los operadores de redes. Por su parte, el plano de datos se basa en los switches CN y estaciones base que utilizan redes de acceso de radio definidas por software. El CN programable ofrece virtualización de red y clasificación de tráfico eficientes. Concretamente, utiliza una virtualización de tres vías de la red que garantiza un funcionamiento eficaz y una asignación de recursos de red sin conflictos entre operadores de red. La clasificación del tráfico se divide en clasificación local (realizada por estaciones base) y clasificación global (realizada por el controlador). Otro desafío es la gestión de movilidad que SoftAir realiza en dos pasos: gestión de la localización y reenrutamiento automático QoS. El reenrutamiento se realiza explotando lo la vista global de la red que obtiene el controlador centralizado. La unión entre colocación optimizada y el patrón de

movilidad de los usuarios establece una ruta de control de tráfico óptima desde dispositivos de plano de datos a través del controlador.

Una variación de la arquitectura Softair es MyNet [ZVS+15], un despliegue jerárquico de controladores para redes 5G [SAA+16] que son responsables de cada parte de la red para reducir el flujo del control del tráfico. Yazici et al. [YKS14] también proponen una arquitectura jerárquica titulada CMaaS que se manifiesta una jerarquía de controlador de cuatro capas. La capa inferior, denominada controlador de UE, es responsable de administrar la selección de la tecnología de acceso de radio disponible para un usuario basándose en el estado de la red local. La siguiente capa, denominada como un controlador *Base Station* (BS) que administra la administración de recursos de radio sensible al tiempo y los programa con la vista de red local. Continuando con la arquitectura en capas, el controlador RAN controla el conjunto de estaciones base obteniendo una vista regional. La capa más alta, llamada controlador de red, ofrece una vista de red global y administra servicios como QoS, gestión de la movilidad y encaminamiento. Las decisiones de control se distribuyen desde las capas inferiores a las superiores. Paralelamente, las capas superiores obtienen información de red de los controladores para obtener una vista de estado global y tomar decisiones de control.

La Tabla 2.2 recopila las soluciones basadas en SDN para WCN y sus características. Las cuatro columnas situadas a la derecha, describe las propiedades de cada solución. La columna llamada OF representa si el modelo propuesto es compatible con el protocolo OF. A continuación, la siguiente columna llamada Escalabilidad, determina si la propuesta puede ser escalable en términos de tamaño de red. La columna de plano de datos muestra qué componente del plano de datos se está mejorando en la propuesta (RAN, CN o ambos). Finalmente, la última columna representada como asignación de recursos indica si la solución propuesta es capaz de asignar recursos de red.

Tabla 2.2: Diferencias entre Software Defined Networks en Wireless Cellular Networks

Referencia	Propuesta	OpenFlow	Escalabilidad	Plano de Datos	Asignación de Recursos
[GPLK13]	SoftRAN	X	X	RAN	✓
[CZCT14]	SoftMobile	X	X	RAN	✓
[JLVR13]	SoftCell	X	✓	CN	X
[AWL15]	SoftAir	✓	X	RAN y CN	✓
[ZVS+15]	MyNet	X	X	RAN y CN	✓
[YKS14]	CMaaS	✓	X	RAN	✓

2.8. Retos y Desafíos

Las ventajas que ofrece SDN como tecnología aplicable a las redes de producción masiva se encuentran cercanas pero no disponibles. Más aún, existen algunos retos en términos

de seguridad, escalabilidad, confiabilidad, entre otros aspectos, que deben superarse con el fin de ser consideradas aceptables para usuarios comerciales. A continuación, se analizan brevemente estos aspectos.

Como se explicó anteriormente, la separación de los planos de datos y de control permite su independiente desarrollo y evolución. En el plano de datos la velocidad de procesamiento de paquetes depende principalmente de la tecnología utilizada en el hardware, ya sea Application-Specific Integrated Circuits (ASIC), Application-specific Standard Products (ASSP), Field Programmable Gate Array (FPGA) o multicore CPU/GPP. Por su parte, en el plano de control el rendimiento depende principalmente del hardware y del NOS (Beacon, POX, Floodlight [Flo17]). Sin embargo, el bajo desempeño de uno de los dos puede ocasionar problemas significativos, como son la pérdida o retraso de paquetes, comportamientos erróneos de la red o denegación de servicio. Por esta razón, es necesario que los diseños de hardware y software para componentes de redes SDN tengan balance en rendimiento, coste y facilidad de desarrollo.

Por otro lado, OF utiliza los recursos de hardware comunes en los equipos actuales mediante el uso de tablas de flujo. Sin embargo, SDN puede extenderse más allá de las tablas de flujo y utilizar otros recursos adicionales que ofrece actualmente el hardware [VCBLGV13]. La integración y estudio de nuevas funcionalidades entre el plano de control y el plano de datos personalizado es un campo recién abierto. Aplicaciones como cifrado, análisis, clasificación de tráfico y dispositivos como middleboxes, procesadores de paquetes personalizados, entre otros, pueden integrarse y ser usados eficientemente con la tecnología SDN. Por otro lado, el número y la ubicación de los controladores dentro de la red es una pregunta abierta. El análisis presentado en [HSM12] expone que los elementos determinantes para la elección del número y ubicación del controlador son la topología de la red y el rendimiento que se espera de la red.

La seguridad es otro aspecto fundamental que también debe ser tomado en cuenta. Por ejemplo, no todos las aplicaciones de red deben contar con los mismos privilegios de acceso [SSHC+13]. Es necesaria la asignación de perfiles, autenticación y autorizaciones para acceder a los recursos de la red. Por otro lado, OF establece el uso opcional de *Transport Layer Security* (TLS) como herramienta de autenticación entre el controlador y el switch. Sin embargo, no existen especificaciones claras que brinden seguridad para sistemas de múltiples controladores que intercambian información entre sí y con los switches. Asimismo, debido a que OF establece que un paquete desconocido sea enviado completamente o su cabecera al controlador, fácilmente se pueden ejecutar ataques de denegación de servicio mediante el envío de múltiples paquetes desconocidos al switch.

La transición de arquitecturas actuales hacia arquitecturas SDN es de igual forma un campo abierto. A pesar de que actualmente ya existen equipos con soporte para OF (NEC, IBM) en el mercado, es imposible remplazar toda la infraestructura ya instalada. El período de transición requiere de mecanismos, protocolos e interfaces que permitan coexistencia eficiente de ambas arquitecturas. Actualmente existen esfuerzos para lograr este objetivo: la

Open Networking Foundation (ONF) publicó el protocolo IF-Config [Ope13] como primer paso para la configuración de equipos con soporte OF. De igual manera, el *European Telecommunications Standards Institute* (ETSI) y el *Forwarding and Control Element Separation Working Group* (ForCES) trabajan en la estandarización de interfaces para el correcto desarrollo de esta tecnología.

Capítulo 3

Trabajos Relacionados

El objetivo general de este capítulo es proporcionar una visión general del estado del arte en diferentes campos sobre la que se ha basado la investigación y las contribuciones de la presente Tesis doctoral. Principalmente, se describen los trabajos relacionados sobre la monitorización de redes SDN. Además, se proporcionan los retos en el campo de la seguridad para redes SDN y cómo los trabajos existentes proponen soluciones para mitigar dichos desafíos. Finalmente, se describe el estado del arte de las redes 5^{th} *Generation* (5G) y los vehículos aéreos no tripulados.

3.1. Monitorización

El campo de la monitorización ha sido ampliamente estudiado en las redes tradicionales IP. Dependiendo de la estrategia utilizada, las soluciones pueden ser clasificadas como pasivas o activas. En los métodos activos se envían los paquetes de sondeo junto con el tráfico normal a través de la red. Estos paquetes se recopilan y analizan para estimar valores como el retardo, estado de las conexiones *End to End* (E2E), etc.

Por su parte, en los métodos pasivos no se añaden paquetes a la red y solamente se analiza la información que atraviesa la misma. Estas tareas se llevan a cabo por agentes que se encuentran en los dispositivos a ser monitorizados. Dichos agentes envían esta información a través de algún protocolo de monitorización conocido como *Simple Network Management Protocol* (SNMP) [CFSD90] o *Network Configuration Protocol* (NETCONF) [EBSB11]. Se utilizan también herramientas para la recolección de información como NetFlow [Cla04], sFlow [PL04] ó jFlow [Mye99]. OF permite que los switches envíen información de su estado a través del intercambio de mensajes propios del protocolo. Con dicha información el controlador estima el estado actual de toda la red. Por ejemplo, el proyecto PayLess [CBAB14] proporciona una API RESTFul para enviar estadísticas de flujos en diferentes niveles de agregación. Asimismo, propone un algoritmo de recolección de estadísticas adaptativo, que disminuye la sobrecarga de la red. El nivel de utilización de los enlaces se obtiene a través de los mensajes flow removed y statistic request. MonSamp

[RSC14] copia y reenvía el tráfico a un agente externo para el análisis (collector and analyzer). La información recolectada por el agente se envía al controlador SDN, que utiliza un algoritmo que optimiza la monitorización modificando la frecuencia de muestreo.

OpenNetMon [vADK14] realiza la monitorización de las estadísticas de los flujos, basándose en parámetros como el retardo o la pérdida de paquetes. La frecuencia de monitorización depende de la variación de las mediciones con respecto al valor previamente calculado. OpenTM [TGG10] analiza diferentes estrategias que son capaces de reducir la sobrecarga en los switches OF. Los experimentos muestran que la estrategia no uniforme proporciona una precisión razonable. Este algoritmo selecciona aleatoriamente dos switches de la ruta y consulta la información de los switches más cercanos al destino.

La propuesta del presente trabajo incluye encapsulación, creación de perfiles de usuario, así como la medición de diferentes parámetros (tasa de transmisión, tasa de pérdidas y retardo). Además, no duplica la información en dispositivos externos, proporcionando una estructura modular para la generación de nuevos algoritmos de monitorización. Por otra parte, el módulo orquestador tiene en cuenta no solamente la carga del controlador sino también el tamaño de la red para la gestión de los diferentes algoritmos.

La monitorización de redes es el uso de un sistema que monitoriza constantemente una red en busca de componentes lentos o defectuosos y se lo notifica al administrador de la red en caso de que se produzcan cortes entre los dispositivos u otros problemas. La monitorización de una red es parte de su gestión, ya que se utiliza para obtener una vista global del tráfico que circula por ella. Las herramientas de monitorización son muy útiles para que los administradores de red analicen la red para prevenir futuros problemas de congestión, flujos de tráfico ilegítimos, entre otros problemas.

Mientras que un *Intrusion Detection System* (IDS) monitoriza una red desde el exterior en busca de amenazas, un sistema de monitorización de red monitoriza el estado de la red en busca de problemas causados por servidores sobrecargados o bloqueados, conexiones de red u otros dispositivos que utilizan diferentes métricas. Las métricas más comunes de medición de características de una red son el tiempo de respuesta y la disponibilidad, aunque las métricas de consistencia y confiabilidad están comenzando a ganar popularidad. La incorporación generalizada de dispositivos de optimización de *Wide Area Network* (WAN) está teniendo un efecto adverso en la mayoría de las herramientas de monitorización de red, especialmente cuando se trata de medir retardos precisos de E2E porque limitan la visibilidad del tiempo de retardo de ida y vuelta (round-trip time). Generalmente, algunos eventos como peticiones fallidas de solicitud de estado de los dispositivos, timeout en el establecimiento de una conexión, o mensajes que no pueden ser obtenidos, producen acciones en el sistema de monitorización. La monitorización de redes se puede enfocar tanto en métodos activos como pasivos. Teniendo en cuenta las dos formas de monitorización, se extrae que cada una tiene sus ventajas y desventajas, pero ambos enfoques son complementarios.

- **Métodos Activos**: son métodos que envían paquetes a dispositivos de red como switches, servidores, entre otros. Por lo tanto, dado que estos métodos inyectan paquetes en la red, impactan directamente en el tráfico de la red. Estos paquetes agregados al tráfico se denominan paquetes de prueba (probe packets) y su función principal es obtener y medir estadísticas como retraso, tiempo de ida y vuelta, entre otros.

- **Métodos Pasivos**: son métodos soportados por hardware de propósito especial (sniffers) o integrados en otros dispositivos como routers, switches o hosts. En caso opuesto a los métodos activos, no aumentan el tráfico dentro de la red cuando miden estadísticas. Los métodos pasivos son muy valiosos en la resolución de problemas de red, pero, por otro lado, son limitados para aislar la ubicación exacta del fallo.

Respecto al uso de métodos pasivos en redes tradicionales, protocolos como SNMP [CFSD90] y NETCONF [EBSB11] permiten monitorizar dispositivos de red con sus estadísticas. De forma similar, las herramientas de monitorización como NetFlow [Cla04] y sFlow [PL04] se usan para estimar una muestra y estadísticas completas de tráfico en redes basadas en flujo.

jFlow [Mye99] es una extensión del lenguaje de programación Java [GJSB00] que permite verificar las anotaciones de flujo de forma estática. Además, proporciona un compilador que verifica (también estáticamente) y evita las fugas de información a través de canales de almacenamiento. jFlow proporciona características que hacen que la verificación de flujo no sea tan restrictiva como otros lenguajes de programación. Estas características son: modelo de etiquetado descentralizado que permite la creación de políticas privativas, un control de acceso que permite asignar y cambiar privilegios de código, un polimorfismo de etiquetado en las clases de datos, una verificación de etiquetas en tiempo de ejecución para garantizar que la información no se filtre y, finalmente, la inferencia automática de etiquetado para abreviar las anotaciones que en otro caso son necesarias. Entonces, aprovechando todas estas características, jFlow es una herramienta poderosa para obtener estadísticas de flujo a través de la red.

Una de las principales propuestas relacionadas con la monitorización de redes en redes SDN es SuVMF [CKY+14]. Proporciona una arquitectura novedosa para redes SDN a gran escala basada en una función de monitorización virtual unificada definida por software utilizando métodos pasivos. Este trabajo consta de tres entidades importantes que son responsables de la gestión de la monitorización, control inteligente de los datos obtenidos y por último, módulos que filtran y transforman los datos. La recopilación de estadísticas utiliza varios métodos pasivos como sFlow [PL04] y SNMP [CFSD90], entre otros.

Métodos pasivos y activos para medir el rendimiento de la red se utilizan en [STH14]. Esta propuesta usa mensajes baliza para enviar paquetes sonda e instalar flujos adicionales en los switches. Luego, estas balizas se utilizan para estimar la tasa de pérdida de paquetes y el retraso. Una solución híbrida entre métodos pasivos y activos es el framework que

se expone en [VCPFGV17]. Este trabajo propone un framework de monitorización de red basado en un módulo orquestador con un método flexible para obtener estadísticas de red. Además, crea un perfil de usuario basado en sus necesidades, recuperando estadísticas con métodos pasivos (tasa de datos y tasa de error) y activos (paquetes sonda a los switches).

OpenNetMon [vADK14] monitoriza las métricas por flujo en las redes OpenFlow. Está especializado en el rendimiento, el retraso y la pérdida de paquetes para determinar si se cumplen los parámetros de QoS en conexiones E2E para encontrar rutas adecuadas. Busca el número mínimo de consultas a los switches para obtener las métricas utilizando un intervalo de peticiones que depende del aumento-disminución de sus valores previos.

En lugar de utilizar una estrategia única para monitorizar, la aplicación de diferentes métodos para consultar estadísticas puede ayudar a reducir la sobrecarga en los switches y en las redes. En este orden, OpenTm [TGG10] propone seguir una distribución de consulta no uniforme con respecto a un esquema uniforme. OpenTm manifiesta que esta estrategia es mucho más rápida que las formas existentes de estimación de tráfico en redes IP. Chowdhury et al. [CBAB14] proponen Payless, un framework de monitorización eficiente y de bajo coste para redes SDN. Su característica principal es que proporciona una vista abstracta de la red y una manera regular de solicitar estadísticas sobre los recursos de la red. Además, dado que se ha desarrollado como un conjunto de componentes conectables, proporciona interfaces para conectarlos a todos (API RESTful de alto nivel).

Continuando con la monitorización de redes de flujo, Flowsense [YLZ+13] es una aproximación que busca estadísticas de alta precisión con un coste de medición cero utilizando la separación física de los planos de control y datos en SDN. Las características clave son: la duración que tarda el flujo en la entrada de la tabla de flujo, la cantidad de tráfico que coincide con ese flujo y, finalmente, el puerto de entrada del tráfico que coincide con la entrada. Sin embargo, tiene algunas limitaciones, ya que depende en gran medida del tipo de tráfico que va a monitorizar, ya que los grandes flujos como una transmisión de vídeo pueden retrasar su cálculo y utilización.

Yu et al. [YJM13] proponen OpenSketch, un esquema de medición de tráfico para SDN que separa el plano de datos de medición del plano de control. Desde el punto de vista del plano de control, OpenSketch proporciona una librería que se encarga de configurar el pipeline y asignar los recursos de red para múltiples labores de medición. Por otro lado, en el plano de datos se proporciona un pipeline trifásico: hashing (recoge el campo de origen de los paquetes y les aplica una función de hashing), filtrado (reúne el campo de destino de los paquetes y los filtra según una regla de coincidencia) y contabilidad (cada regla de coincidencia es asignada con un campo de índice que se usará en la tercera fase para calcular la ubicación del contador). Estos pasos intentan reducir la memoria de los switches y maximizar la precisión en los valores medidos de la red monitorizada.

Suárez-Varela et al. [SVBR18] proponen un sistema de monitorización habilitado para SDN y reconocimiento de aplicaciones. Este sistema de monitorización está dividido en dos bloques. El primer bloque, llamado SBAR, es un sistema basado en OF que

produce informes de monitorización a nivel de flujo. El segundo bloque es una herramienta de análisis de datos que procesa los informes resultantes de SBAR y muestra valiosas estadísticas de red a través de una interfaz web al usuario. SBAR, a su vez, está compuesto por un módulo de medición que se encarga de mantener los valores de flujo en los switches e informarlas a los controladores y un módulo de clasificación, que produce etiquetas de clasificación para cada flujo en los informes de medición. FlowCover [SWXH14] es un esquema de bajo coste para monitorizar redes SDN reduciendo considerablemente el tráfico de comunicación disminuyendo la agregación de mensajes de solicitud y respuesta abandonando consecuentemente la vista global de la red. Además, aprovecha el tráfico real a través de la red cambiando el esquema de sondeo de solicitudes. Por lo tanto, el esquema de monitorización toma toda esta información y calcula un plan de sondeo efectivo y lo reenvía al recolector de estadísticas de flujo.

Duplicar el tráfico y enviarlo a un agente de monitorización es la idea utilizada en MonSamp [RSC14] donde los autores crean dos agentes denominados colector y analizador los cuales leen continuamente los flujos dentro de los switches. Según la capacidad del monitor y la congestión de los enlaces de la red, el algoritmo aumentará o disminuirá las reglas de flujo en los switches. Las propiedades de estas estrategias de monitorización se pueden usar en paralelo con otros módulos para obtener mejores resultados en términos de QoS, entre otros. En [VCPFGV15] se presenta un framework para el enrutamiento optimizado de contenido multimedia en combinación con un módulo de monitorización para proporcionar QoS en diferentes servicios multimedia. Su principal ventaja es la manera fácil de agregar algoritmos de enrutamiento para obtener la mejor ruta en el envío de datos.

Continuando con propuestas de duplicación del tráfico, Li et al [Li18] propone un sistema de detección de anomalías basado en un canal lateral ligero para la reducción del volumen del tráfico el cual es monitorizado finalmente por un IDS. El tráfico en Science DMZ se refleja (duplica) en un sistema de monitorización pasivo, que consiste en switches OF para la dirección del tráfico y máquinas virtuales que ejecutan un sistema de detección ligero e instancias IDS como Bro o Snort. Todo el tráfico reflejado se entrega al sistema de detección liviano, que realiza la detección de anomalías basadas en canales laterales. Al mismo tiempo, se distribuye una copia del tráfico a las instancias de IDS. Una vez que un flujo se considera válido (es decir, tráfico legítimo) basado en el análisis del sistema de detección liviano, el sistema de detección liviano actualizará las reglas de flujo en los switches OF de modo que el flujo considerado válido no se copiará en las instancias IDS para tratamiento. Sin embargo, este flujo válido aún debe ser monitorizado por el sistema de detección liviano. Seufert et al [STG18] revisan e investigan el impacto de la adaptación en el QoE de HAS en diferentes estudios de QoE. Basándose en los resultados encontrados, concluyen que se pueden diseñar mejores lógicas de adaptación, las cuales alcanzan un QoE más alto maximizando primero el tiempo en capas de alta calidad, en lugar de centrarse en una frecuencia de conmutación baja o un comportamiento conservador de

conmutación. Además, han investigado diferentes estrategias de asignación de recursos para la transmisión de vídeo y los flujos de navegación web en un enlace compartido que se produce cuello de botella. Wang et al. [WSYX18] proponen el esquema LESLA que está diseñado para detectar tanto el nodo y controlar ataques de saturación en cada dispositivo para reducir la sobrecarga del proceso de monitorización del tráfico. El proceso de LESLA se compone de diferentes etapas: en la etapa de inicialización, el controlador está entrenado con un alto volumen de registros de tráfico de red actualizados para generar un modelo de detección de anomalías. La etapa de entrenamiento se lleva a cabo offline y el modelo generado se transmite a cada dispositivo de reenvío. Según el modelo de detección producido por el controlador, cada vez que un dispositivo recibe un paquete, el dispositivo ejecutará el algoritmo de detección de ataques localmente para clasificar el paquete entrante. Si el paquete se clasifica como anormal, se bloqueará o restringirá dentro de un área limitada. De lo contrario, el paquete se tratará más de acuerdo con las políticas de gestión.

3.2. Métricas de Calidad

Actualmente debido al desarrollo de las capacidades computacionales y de visualización de los dispositivos terminales, la cantidad de información multimedia que circula por la red ha crecido exponencialmente. Para el año 2013 el tráfico de vídeo alcanzó el 66 % del total del tráfico IP (público y privado) y se espera que para el año 2021 se alcance un número de dispositivos conectados generando tráfico que será tres veces la población global [Cis17]. El contenido en *High Definition* (HD) se ha convertido en un gran factor de nivel de calidad, en 2011 por primera vez el tráfico de vídeo en HD sobrepasó la definición estándar. A pesar de los incrementos en ancho de banda, la infraestructura de red actual tiene dificultades para evitar congestión en las transmisiones multimedia y en consecuencia la degradación de la calidad de vídeo (congelado de imágenes, playback entre audio-vídeo). SDN gracias a su visión global del estado de la red abre nuevas perspectivas para mejorar y garantizar QoS en transmisiones multimedia.

Németh et al [NSSG12] propone algunas funcionalidades adicionales para el switch OF. Estas capacidades son: filtros Bloom con reenvío, Greedy routing y codificación de la información en la red. Propone asimismo la modificación del protocolo OF añadiendo tres acciones simples (localizadas en tres registros) soportadas por una lógica XOR entre dos flujos de datos. La codificación se realiza haciendo una copia de cada paquete que se vaya a cifrar enviándola a una cola auxiliar que contiene solamente paquetes cifrados.

Egilmez et al presenta OpenQoS [EDBT12] para la implementación de QoS en transmisiones multimedia sobre redes OF. OpenQoS agrupa el tráfico entrante como flujos de datos y flujos multimedia, donde los primeros mantienen la ruta tradicional del camino mínimo mientras que los segundos son dinámicamente situados en rutas donde la calidad de servicio está garantizada. Además de los módulos proporcionados por el controlador

Floodlight, OpenQoS añade dos módulos para habilitar la administración de las rutas.

En [GEB+13] se propone una QFF basado en OF, el cual pretende maximizar la calidad de experiencia QoE del usuario. QFF monitoriza el estado de todos los switches y aplicaciones de vídeo y distribuye dinámicamente los recursos de la red a cada dispositivo.

Otro de los conceptos que ha tomado fuerza es el de QoE [LCMP12b]; el cual intenta redefinir la QoS tomando en consideración el nivel de aceptación del usuario de un determinado servicio o aplicación multimedia. En este sentido, SDN permite optimizar las tareas de administración multimedia. Por ejemplo, en [KSKD+12] se mejora la experiencia QoE a través de la optimización de rutas. Esta arquitectura consiste de dos elementos: el QMOF, el cual obtiene los parámetros multimedia y determina la configuración apropiada para el enlace. El segundo elemento, PAF mantiene actualizada la topología de la red. En el caso de una degradación de la calidad en los enlaces, el sistema automáticamente modifica los parámetros de los mismos tomando en cuenta las prioridades de los usuarios.

3.3. Seguridad en Redes Definidas por Software

La arquitectura de SDN introduce un gran potencial en el campo de la innovación en el uso de las redes. La unión entre la programabilidad y las visión centralizada de la red, suponen un proceso de colección de inteligencia para IDS y *Intrusion Prevention System* (IPS). Esta base de conocimiento se traduce en una mayor robustez frente ataques maliciosos respecto a las redes tradicionales.

Para elaborar la funcionalidad de la arquitectura SDN, se presentan tres planos principales: Plano de Datos, Plano de Control y por último el Plano de Aplicación. El Plano de Datos es la combinación de los elementos que reenvían los flujos de trafico basado en las instrucciones provistas del plano de control. El Plano de Control es un plano lógico de control centralizado que ejecuta un sistema operativo, mantiene la visión global de la red y provee abstracciones hardware a las aplicaciones SDN. El Plano de Aplicación contiene las aplicaciones SDN con varias funcionalidades, como gestión de la red, implementación de políticas y servicios de seguridad. Por tanto se definen el plano de datos, el plano de control y el plano de aplicación.

La presente sección proporciona un análisis de las diferentes soluciones a desafíos de seguridad en SDN. Este estudio abarca desde esquemas de replicación de controladores de la red a través de resolución de conflictos hasta mecanismos de autenticación. De forma similar, varias propuestas se han planteado para explotar el framework de SDN y así mejorar la seguridad. Además, los aspectos de seguridad son categorizados de acuerdo al plano que se va a atacar o que se ve afectado.

3.3.1. Seguridad en los Planos

Las redes de comunicaciones seguras contienen unas propiedades básicas que son: integridad, confidencialidad, autenticación, disponibilidad de la información y su conocimiento [DS07]. Los profesionales de la seguridad deben salvaguardar los datos, activos de la red (dispositivos) y las transacciones de la comunicación a través de la red para proveer a ésta de una protección contra ataques maliciosos.

Varios aspectos sobre la seguridad en SDN se categorizan en la Tabla 3.1. En ella, se presentan los ataques y/o aspectos sobre la seguridad (filas) respecto a los diferentes planos afectados de la infraestructura SDN (columnas). El primer grupo de problemas de seguridad en SDN son los que comprenden el acceso no autorizado a las aplicaciones SDN. En el segundo, se recogen las técnicas de modificación de datos en los paquetes que circulan en la red. Las aplicaciones maliciosas que insertan reglas fraudulentas o "secuestran" en el controlador se categorizan en el tercer grupo. El cuarto comprende los ataques más extendidos dentro de las redes que son los de *Denial of Service* (DoS). Por último, se agrupan los asuntos de seguridad que se aplican en la configuración de la red.

Tabla 3.1: Categorización de la Seguridad en las Distintas Planos

Ataque o reto en seguridad	**Plano de Datos**	**Plano de Control**	**Plano de Aplicación**
Acceso no autorizado			
Acceso al controlador no autorizado		X	X
Aplicaciones no autenticada		X	X
Modificación de datos			
Modificación de reglas de flujo para modificar paquetes	X	X	
Aplicaciones maliciosas			
Inserción de reglas maliciosas		X	X
Secuestro del controlador	X	X	
Denegación de servicio			
Inundación de la comunicación Controller-Switch	X	X	
Inundación tabla de flujo del switch	X		
Asuntos de configuración			
Ausencia de TLS (u otra técnica de autenticación)	X	X	
Ejecución de las políticas		X	X

La seguridad ha sido un requisito imprescindible en las redes comunicación debido a su complejidad, a la dificultad para ofrecer soluciones seguras basada en propietarios y a la débil noción para identificar redes IP. El principal reto necesario para permitir la comunicación activa es garantizar la formación de nodos activos que frenen el avance de programas de inyección de datos maliciosos. Como resultado, la seguridad activa [LCM03] y otros ensayos de seguridad [MLP+01] fueron propuestos para asegurar la seguridad de los nodos a través de mecanismos de autenticación y autorización. Sin embargo, la complejidad del manejo y seguridad de estos nodos activos sigue siendo una tarea desafiante en las redes activas.

Greenberg et al. [GHM+05b] han permitido asociar la naturaleza frágil de las redes

de comunicación a la complejidad del control y manejo de los planos presentes en las redes tradicionales. A consecuencia de las mencionadas dificultades, se ha propuesto el "4D approach", categorizado como ensayo y cuyo nombre viene definido por los cuatro planos: decisión, diseminación, descubrimiento y datos. En el ámbito empresarial, SANE [CGA+06] del inglés "Secure Architecture for the Networked Enterprise" trata sobre una arquitectura diseñada para facilitar las conexiones entre empresas ejerciendo tres funciones principales: permitir la autenticación del usuario, hosts y switches a la vez que mantiene las claves simétricas de cada una de las comunicaciones seguras. Además, notifica y controla el acceso a los servicios disponibles. Por último, controlar toda la conectividad en la red SANE.

3.3.1.1. Plano de Datos

Ya que la capacidad de toma de decisión no es tomada por los switches, los principales desafíos para mantener la seguridad se centran en el reconocimiento de reglas de flujos genuinas respecto de aquellas reglas falsas o maliciosas. Un segundo reto está basado en el número de flujos de entrada que un switch puede mantener hasta que el controlador resuelva las reglas de dichos flujos. Debido a esto, el plano de datos es propenso a ataques de saturación ya que los buffers son limitados y no podrían atender a todos los flujos solicitados (TCP/UDP). La separación de los planos de control del de datos permite la manipulación de flujos sigilosamente por parte de un atacante a través de la manipulación de las reglas OF, resultando en ataques de varias redes activas tales como el *Man In The Middle* (MITM) o los ataques de "agujero negro" (black-hole attacks). Además, SDN permite que el tráfico de red sea enrutado a través de un cortafuego centralizado para asegurar el plano de datos. Sin embargo, los mensajes de monitorización entre un switch y el controlador pueden tardar tiempo necesario mientras que los recursos del switch sean agotados por flujos falsos hasta verse comprometidos en un ataque de DoS.

En el mismo contexto, los Sistemas de Monitorización son esenciales en la protección de la red respecto a ataques. En [BdSMP10], los autores presentan un método de detección de DDoS basado en varias características del flujo de tráfico. Este sistema usa un controlador basado en NOX [GKP+08] el cual monitoriza los switches en intervalos de tiempo regulares y usa Mapas Auto-Organizados (*Self Organizing Map* (SOM)) para identificar flujos de tráfico anormales.

Una aplicación específica de sistemas de monitorización como son los IDS, ha sido el foco de muchas soluciones SDN. Skowyra et al. [SBB13] proponen un IDS basado en aprendizaje el cual utiliza la arquitectura SDN tanto para detectar como responder ataques en dispositivos móviles. Además, en [GNBC10] se describen *Network Intrusion Detection System* (NIDS) y *Network Intrusion Prevention System* (NIPS) los cuales permiten al administrador de la red a configurar patrones de caracteres para ser usados en un módulo de DPI.

Por último, la especificación de switches OF, describe el uso de la Seguridad de la capa TLS con una autenticación por parte de los switches y los controladores. La escasez de inclusión de TLS por parte de los fabricantes, debido a que su estándar no está especificado y por otra parte, la posibilidad de los ataques de DoS, son el centro de la evaluación de las vulnerabilidades de OF. Benton et al. [BCS13] afirman que la ausencia del uso de TLS podrían liderar tanto la inserción de reglas fraudulentas como la modificación de reglas en los switches de la red.

3.3.1.2. Plano de Control

Los principales retos en seguridad y las amenazas existentes en el plano de control se describen a continuación:

1. **Amenazas desde las aplicaciones:** las aplicaciones implementadas en la parte superior del plano de control plantean serios peligros en la seguridad de éste. Por lo general, mantener la seguridad del controlador supone un reto desde el punto de vista de la capacidad de éste para autenticar las aplicaciones y autorizar las fuentes usadas por las aplicaciones con una correcta auditoría, aislamiento, y seguimiento [HWZ13]. Las aplicaciones poseen tanto requisitos funcionales (por parte de los controladores subyacentes) como requisitos de seguridad (la ruta de datos se debe garantizar). Por ejemplo, aplicaciones de balanceo de carga necesitan de estadísticas de la red como tasa de tráfico o contadores de paquetes de cada switch para realizar el equilibrado de la carga de tráfico. Otras aplicaciones, como, por ejemplo, las aplicaciones de IDS necesitan de la inspección de la cabecera de los paquetes.

2. **Amenazas debido a la escalabilidad:** la escalabilidad del controlador alude su susceptibilidad a ataques DoS y DDoS. Otro desafío de las implementaciones actuales de controladores está en la especificación del número de dispositivos de reenvío necesarios para ser dirigidos por un único controlador que haga frente a las restricciones de retardo. Si el número de flujos en el controlador aumenta, existe una alta probabilidad de que el tiempo de almacenamiento se incremente de modo que sea profundamente dependiente de la potencia del procesamiento del controlador. La limitación de las capacidades del controlador puede dar lugar a un punto de fallo, el cual podría solucionarse a través del uso de múltiples controladores. Sin embargo, en [YBG13] se ha demostrado que simplemente utilizando múltiples controladores en SDN, no se puede proteger la red de un punto de fallo. La razón está es que la carga que acumula los controladores operativos por parte de los controladores fallidos, puede exceder sus capacidades y por tanto resultar en el empeoramiento de la situación. Un ejemplo de este error es el denominado fallo en cascada de controladores el cual se describe en [YBG13]. Además, la sincronización de sistemas distribuidos también supone un gran problema en seguridad en cuanto que un componente del sistema esté siendo atacado mientras no está sincronizado con los demás.

3. **Ataques DoS:** los ataques DoS y DDoS son unos de los principales desafíos en la amenaza de la seguridad de los controladores de SDN. Un ataque DoS es un intento de hacer indisponible un recurso para que los usuarios legítimos no puedan usarlo. En [SG13] se ha demostrado que un ataque DoS en SDN explota la lógica de separación de los planos del control de datos de SDN. Además, en [FBMP12] ha sido comprobado dicho ataque en los controladores SDN donde un atacante continuamente envía paquetes IP con encabezados aleatorios para poner al controlador en un estado de no respuesta. Los autores [FBMP12], usan un controlador secundario para mejorar la resistencia a este tipo de ataques, sin embargo, el mecanismo de detección de DoS o DDoS sigue siendo necesario ya que este controlador secundario podría también ser susceptible a estos ataques. Por tanto, el uso de múltiples controladores no es la respuesta para los ataques DDoS, ya que pueden desencadenar fallos en cascada de múltiples controladores tal y como se ha demostrado en [YBG13].

3.3.1.3. Plano de Aplicación

SDN presenta DoS propiedades principales las cuales permiten establecer las bases tanto para la innovación de redes como para la superación de los retos relacionados con la seguridad. Esto es posible gracias a la capacidad para controlar una red a través de un software y también a la centralización de la inteligencia de la red en controladores de [KRV13]. Muchos de los retos planteados por las aplicaciones SDN para contrarrestar la amenaza a la seguridad son:

1. **Autenticación y autorización:** El principal tema en los dominios SDN se centra en la creación de aplicaciones que permitan la autenticación ante la actual y rápida tendencia de la ingeniería de software y hazañas de hackers. En OF, las aplicaciones ejecutadas en el controlador implementan la mayoría de las funcionalidades del plano de control y son comúnmente desarrolladas por otras partes más que por los proveedores de controladores. Así pues, la autenticación de un creciente número de aplicaciones en redes programables con una arquitectura de control centralizada es el principal reto en seguridad. Estudios dirigidos por el grupo de Kreutz et al. [KRV13] han propuesto vectores de riesgo/amenazas para describir la vulnerabilidad de la seguridad en SDN. Como resultado, se ha observado la ausencia de mecanismos convincentes para establecer una relación de confianza entre el controlador y las aplicaciones en SDN. Por tanto, una aplicación maliciosa podría potencialmente crear un caos en la red ya que los controladores de SDN proporcionan abstracciones que son traducidas por las aplicaciones a los comandos de configuración para las infraestructuras subyacentes. De igual modo, si un servidor de aplicaciones que almacena detalles de los usuarios se encuentra comprometido, las credenciales de los usuarios legítimos podrían ser usados para permitir flujos autorizados, aunque realmente falsificados en la red. Además, existen varias técnicas que certifican el

uso de dispositivos de red en el sistema, aunque sin mecanismos para acreditar las aplicaciones de red.

2. **Control de acceso y permisos:** Debido a la implementación de las aplicaciones en la mayoría de los servicios de red en SDN, un correcto control de acceso y mecanismos de responsabilidad son requeridos para asegurar la seguridad en la red. Hartman et al [HWZ13] identifica tres clases de aplicaciones que pueden afectar a la seguridad de la red en SDN: aplicaciones sensibles a la red, aplicaciones que proveen servicios a la red y por último aplicaciones que combinan las DoS clases anteriores. A parte de esto, el control de acceso y la responsabilidad de aplicaciones anidadas (aplicaciones que usan instancias de otras aplicaciones) es un desafío real en SDN. Un ejemplo de este tipo de aplicaciones está mencionado en [XTL+12] donde se describen aplicaciones en SDN que contiene tanto aplicaciones anidadas en SDN como no. La diferencia entre ellas es que las que son anidadas son capaces de localizar y establecer directamente comunicaciones con el controlador mientas las que no tienen anidadas se comunican indirectamente con datagramas es formatos específicos. Del mismo modo, una aplicación SDN legítima puede volverse comprometida y convertirse en una puerta de entrada para el acceso no autorizado al plano de control de red. Por último, la responsabilidad del mantenimiento de los recursos de la red mediante aplicaciones anidadas es otro gran desafío.

3.3.1.4. Aspectos Adicionales de Seguridad

Los atacantes usan varias técnicas de escaneo para objetivos vulnerables en la red. Una técnica para frustrar este tipo de ataques es la creación de direcciones IP virtuales aleatorias usando SDN [JASD12]. Este método usa el controlador OF para gestionar una fuente de direcciones IP virtuales las cuales serán asignadas a los hosts dentro de la red ocultando las direcciones IP reales. Actualmente, se han producido varios debates sobre la integración o no de middleboxes que provean seguridad en SDN, para explotar los beneficios de su programabilidad y así redireccionar el tráfico seleccionado a través del él. Por ejemplo, la arquitectura Slick presentada en [ABF+13] propone un controlador centralizado responsabilizado de instalar y migrar funciones dentro de middleboxes personalizados. Por otro lado, en [FSYM13] se describe una arquitectura llamada FlowTags la cual plantea el uso de middleboxes mínimamente modificados que interactúan con el controlador SDN a través de una API propia. Principalmente, se apoya de la información sobre el flujo del tráfico de la red la cual será posteriormente guardada en la cabecera (paquete etiquetado) de los paquetes. Esta función provee rastreo del flujo y habilita el enrutamiento de los paquetes etiquetados. Colocar los middleboxes en un lugar apropiado debe ser determinado junto con la penalización de rendimiento que puede ser tolerado cuando el tráfico es desviado a través de los enlaces adicionales. Estas cuestiones todavía no se han resuelto.

3.3.2. Soluciones de Seguridad

3.3.2.1. Soluciones en el Plano de Datos

Como punto de referencia, se debe proteger el plano de datos de aplicaciones maliciosas, las cuales pueden instalar, cambiar o modificar reglas de flujos en la ruta de datos. Por tanto, es necesario reforzar los mecanismos de seguridad de grano fino tales como la autenticación y autorización usados por las aplicaciones ya que permiten el cambio de las reglas de flujos. FortNox [PSY+12] es una de las plataformas ejemplo que no solo facilita a los controladores NOX OF comprobar en tiempo real las posibles contradicciones de las reglas de flujos sino que además autoriza a las aplicaciones OF antes de que éstas puedan cambiar las reglas de flujos. Debido a la importancia de la conectividad del controlador para permitir el correcto funcionamiento de los switches, aquellas conexiones redundantes o mecanismos de rápida recuperación de links deben ser ejecutados en el momento para establecer la conexión entre el controlador y el switch y no alargar un retardo que pueda dar pie a posibles ataques. En [FBMP12] se propone la replicación del controlador para mantener operativo el switch incluso cuando el controlador principal fallara o presentara comportamientos anómalos. Por otro lado, la planificación adecuada de la red y la segmentación pueden también ayudar a reforzar la adaptación de los switches OF y maximizar su conectividad con los controladores.

3.3.2.2. Soluciones en el Plano de Control

Las soluciones para la seguridad del plano de control se encuentran categorizadas en:

1. **Aplicaciones:** el controlador intensificador de la seguridad *Security Enhanced* (SE) Floodlight [PCF+15], es una versión extendida del controlador Floodlight [Flo17] original. SE Floodlight proporciona mecanismos para la separación de privilegios a partir de la adición de un Northbound API programable seguro al controlador, para operar como mediador entre las aplicaciones y el plano de datos.

2. **Escalabilidad del controlador:** en los estándares OF de SDN, el controlador instala reglas de forma separada para cada conexión cliente, también denominado micro flujo, promoviendo la instalación de una inmenso números de flujos en los switches y la carga pesada en el controlador. Por consiguiente, se han sugerido varios ensayos tanto para minimizar la carga en el controlador como para distribuir las funciones del plano de control o maximizar la potencia de procesamiento y memoria de los controladores. En [Fer13] se expone el análisis comparativo de los modelos de varios controladores OF reactivos y proactivos para comprobar la escalabilidad. Los controladores reactivos aplican las reglas después de que los flujos lleguen al switch, mientras que los controladores proactivos establecen las reglas de flujos antes. Así pues, en [Fer13] se demuestra que los controladores proactivos son más escalables que sus competidores/oponentes.

3. **Mitigación DoS:** los ataques DoS o DDoS podrían verse mitigados tras analizar el comportamiento de los flujos y las estadísticas de éstos almacenados en los switches OF. Debido a la fácil búsqueda de las estadísticas del switch en el controlador OF, el proceso de recopilación de estadísticas en OF, es comparablemente rentable debido al bajo gasto. Aplicando inteligencia artificial encontramos los SOM que son redes neuronales usadas para transformar un patrón de n dimensiones en un mapa de 1 o 2 dimensiones. En [BdSMP10] se presenta un SOM [Koh90] para la detección de ataques DDoS.

4. **Ubicación del controlador:** la problemática de la ubicación del controlador se presenta en [HSM12] donde se muestra que el número de controladores y las localizaciones topológicas de los éstos son DoS puntos clave para la escalabilidad de la red y la adaptación en SDN.

3.3.2.3. Soluciones en el Plano de Aplicación

En SDN, el controlador actúa como un plano intermedio entre el hardware de la red y aplicaciones ocultando la complejidad de la red a las aplicaciones. Por lo tanto, la arquitectura de control centralizado que ofrece SDN hace fácil el despliegue de nuevas aplicaciones que reúnan estadísticas de red y características de los paquetes a través del controlador para implementar nuevos servicios de seguridad. Un ejemplo es el lenguaje FRESCO [SPY+13], basado en scripts, el cual facilita a los desarrolladores implementar nuevas aplicaciones de seguridad que pueden ser empleadas en cualquier controlador OF.

1. **Control de acceso y permisos:** las aplicaciones necesitan trabajar en su entorno funcional y tener controlado el acceso a los las fuentes de red. PermOF [WCH+13] es un sistema de grano fino basado en permisos usado para proporcionar a las aplicaciones OF acceso a el controlador OF y la ruta de datos. Su diseño está basado en un conjunto de permisos y mecanismos aislado. Este conjunto de permisos está a su vez categorizado en permisos de lectura, notificación, escritura y sistemas.

2. **Cumplimiento con la seguridad de red:** en SDN, las aplicaciones deben tener una visión de la red consistente y ser consciente de los cambios y las condiciones de la red. Así pues, en [BZZ+14] se ha propuesto un método que no sólo facilita la verificación y la depuración de las aplicaciones SDN sino que a la vez favorece el mantenimiento de la consistencia y la percepción de los cambios de las características de la red.

3.4. Redes 5G

Durante las últimas décadas, las redes de comunicación celular han evolucionado de 2^{th} *Generation* (2G) a 3^{th} *Generation* (3G) [3gs00] y luego a 4^{th} *Generation* (4G) [MA15]. Gracias a la mejora continua en los dispositivos móviles, una nueva generación de redes celulares llamada 5G [MA15] permite transferir una gran cantidad de información en una alta velocidad de datos. Las redes 5G proporcionan accesibilidad rápida y puntual a la información en cualquier momento y lugar.

Debido a esta mejora y a algunas investigaciones realizadas, se espera que la cantidad de dispositivos conectados a redes celulares hasta el final del año 2021 supere los 50 mil millones de dispositivos [Eri11]. Los resultados obtenidos conducen a la producción de una gran cantidad de tráfico de datos. Por lo tanto, el desarrollo y la mejora de los dispositivos requieren un mayor ancho de banda y menos retraso [Eri15]. En consecuencia, se ha propuesto una nueva generación de comunicaciones celulares denominada 5G. Obviamente, las redes 5G se considerarán uno de los requisitos esenciales porque las redes actuales y predecesoras como 3G y 4G no pudieron responder al volumen de transmisión de datos requerido por los usuarios. Los principales avances tecnológicos que traerán renacimiento a las redes de comunicación inalámbricas se describen en [ANLC16].

Se prevé que la 5G abrirán oportunidades de desarrollo para nuevas industrias (mercados verticales). Sin embargo, estos mercados verticales comienzan a usar casos con requisitos previos de viraje que las redes 5G futuras necesitan ayudar de manera eficiente. La segmentación de red podría ser una solución característica para acomodar simultáneamente, en un framework de sistema típico. [OLAL+17] define el concepto de segmentación de red, con un foco específico en su aplicación a redes 5G. Además, sostiene una revisión sobre la arquitectura SDN propuesta por la ONF [PLHea11] y muestra que proporciona instrumentos para ayudar a la segmentación. Si bien dicha arquitectura está preparada para la implementación de la segmentación de red, se queda corto en algunas capacidades básicas que puede proporcionar NFV [YBSS17]. Como correspondencia a las redes 5G, el *Wireless Network Function Virtualization* (WNFV) ayudará a la virtualización de las capacidades de todo el sistema para mejorar la disposición de las 5G-IoT [LDXZ18], en el que NFV desacoplará los equipos adaptables, versátiles de las capacidades básicas del sistema para potenciar el 5G-IoT en torno a los servidores convencionales en la nube. La NFV tiene la intención de proporcionar un sistema adaptable a las aplicaciones 5G-IoT, que permitirá la segmentación de red personalizada sobre la nube distribuida para crear redes programables para aplicaciones 5G-IoT. En este sentido, la contribución de la presente Tesis Doctoral es compatible con la monitorización de los elementos de enrutamiento de la red de 5G.

Recientemente, se han realizado varios estudios de investigación en redes 5G. Los autores en [MA15] han analizado diferentes tipos de proliferación de ondas en los sistemas celulares. Trabajos como [ANLC16, CSSK+15] estudian varios modelos que ofrecen

soluciones adecuadas, pero independientemente de esto, los desafíos no se han revisado por completo. Además de los estudios mencionados anteriormente, otros trabajos diferentes, como por ejemplo, [ANLC16] investigan alrededor de cinco tecnologías disruptivas de redes 5G (mmWave, mMIMO, comunicaciones de máquina a máquina, etc.). En cualquier caso, eso podría aclarar más estrategias como SDN, NFV e intercambios *Device to Device* (D2D). En otros trabajos diferentes [WHG+14, MZD+15, ARS16], se han propuesto estructuras e innovaciones clave.

Las redes celulares 5G adoptarán muchas mejoras distintivas, en contraste con las redes 4G, para proporcionar velocidades de transmisión ampliadas con latencia disminuida, rendimiento y confiabilidad mejorados, tamaño de dispositivos terminales disminuidos y hardware eficiente en términos de energía. Sin embargo, el aumento de las innovaciones avanzadas impulsará este avance más allá de 5G o comúnmente llamado como redes celulares 6^{th} *Generation* (6G). Los controladores clave para las redes celulares 6G se pueden describir en [AEH19] como: redes totalmente conectadas (con la expansión del IoT, cada dispositivo remoto estará asociado con al menos una red de acceso inalámbrico servida por diferentes *Access Point* (AP) o BS. De esta manera, trabajos como [SBC19, DB18] presentan una visión para la sexta generación 6G y sus requisitos que proporcionan velocidades de bits ultra altas, como 100 o incluso 1,000 Gb/s, disponible solo en sitios especiales, y ofreciendo servicios de largo alcance, baja tasa de bits, energía ultrabaja y larga latencia.

3.5. Vehículos Aéreos no Tripulados

Dar disponibilidad universal a varios tipos de dispositivos es el desafío clave para 5G y tecnologías que van más allá del 5G (*Beyond 5G* (B5G)). Se considera que los vehículos aéreos no tripulados, abreviadamente UAV (de sus siglas en inglés) son una parte importante de las redes inalámbricas futuras que posiblemente pueden fomentar la transmisión inalámbrica y reforzar las transmisiones de alta velocidad. En contraste con las correspondencias con una base fija, los *Unmanned Aerial Vehicle* (UAV) tienen propiedades notables: organización adaptable, enlaces de asociación sólidos *Line-of-Sight* (LoS) y grados de estructura adicionales de independencia de la movilidad controlada. En [LFZ18] se presenta un estudio exhaustivo sobre la correspondencia de los UAV con los sistemas remotos 5G/B5G. Para empezar, presentan rápidamente los antecedentes básicos y las redes integradas espacio-tierra-aire, al igual que examinan las dificultades de investigación relacionadas con la arquitectura de las redes integradas emergentes. En ese punto, ofrecen una estudio exhaustivo de diferentes procedimientos 5G que dependen de las plataformas UAV, los cuales se clasifican por diversos dominios, incluida la capa física, la capa de red, la comunicación, el registro y el almacenamiento. Del mismo modo, se realiza un croquis con un número extraordinario de temas de investigación abiertos y se distingue como posibles lineas de investigación futuras.

Actualmente, las plataformas UAV son una fuente importante de información utilizada para la evaluación, el mapeo y los problemas de visualización en *3 Dimensions* (3D). Dado que estas plataformas se pueden considerar como una alternativa de mínimo esfuerzo en contraste con la fotogrametría aérea tripulada tradicional, se presentan nuevas aplicaciones en el área de corto alcance y proximidad. Varios tipos de vehículos aéreos no tripulados, como los de hélice giratoría, de ala fija y los cuadrirrotor [LLM10] son aptos para reproducir la adquisición de información fotogrametríca con cámaras conservadoras novatas o computarizadas que vuelan en modos manual, semiautomatizado y autosuficiente. Siguiendo un proceso de trabajo fotogramétrico regular, se pueden entregar resultados 3D como modelos avanzados de superficie o paisaje, modelos 3D terminados, datos vectoriales, incluso en grandes zonas. Por ejemplo, [NJR+12] presenta una investigación sobre avalanchas que utilizan UAV para obtener información importante requerida para la generación efectiva de orto-mosaicos dependientes de *Digital Terrain Model* (DTM) fotogramétricos, para minimizar los errores de georreferenciación. Además, Nex et al. [NR14] describen el estado del arte de UAV para aplicaciones geomáticas, dando una revisión de varias etapas, aplicaciones y análisis contextual de UAV, demostrando las mejoras más recientes del manejo de imágenes de UAV.

Abujubbeh et al [AATF19] presenta una descripción general de la aplicación de las redes SDSN en redes inteligentes que mejoran la solidez de la red. Utilizando diferentes trabajos de investigación que emplean el paradigma SDN en redes inteligentes, los autores manifiestan las deficiencias en los sistemas de energía convencionales que utilizan el modelo de red inteligente para alcanzar oportunidades como el control automatizado sobre los recursos de red que mejoran la resistencia de la red. Además, los autores analizan los dispositivos necesarios y las tecnologías de comunicación necesarias para una transición exitosa a redes inteligentes basadas en SDN. Al-Turjman et al. [ATAMM20] describen en detalle la aplicación de SDN en redes UAV. De esta manera, el paradigma SDN ha sido probado como una solución para la comunicación multi-UAV ya que ofrece servicios flexibles para el desacoplamiento del plano de gestión de los UAV y la capacidad de programación de la red. Los autores también investigan el despliegue de los UAV, definiendo dos modos de modelado (en línea y fuera de línea) para la planificación y la generación de rutas. Además, este trabajo discute la aplicación de SDN como nueva solución en el campo de drones para mantener los requisitos de seguridad de la red, especialmente en redes de emergencia donde la robustez de la red es el objetivo principal. Siguiendo esta línea, este trabajo abre investigaciones y trabajos futuros en seguridad, privacidad, recolección de energía y SDN. Otro desafío en el campo de los UAV es el posicionamiento de drones en la aplicación de misiones críticas. De esta manera, [AT19] propone un enfoque que utiliza un filtro de Kalman de dos niveles que intenta resolver los problemas de posicionamiento. Utiliza los datos recuperados del sistema de navegación inercial y el sistema de posicionamiento global GPS para predecir los errores de posicionamiento del INS.

Capítulo 4

Contribuciones

Del estudio de la literatura se han obtenido distintos enfoques y aproximaciones de las técnicas actuales que se utilizan para monitorizar redes SDN. La principal búsqueda de las herramientas de monitorización es la obtención de estadísticas lo más precisas posibles intentando no sobrecargar la red con mensajes al controlador para obtener dichas estadísticas. Con respecto a dichos resultados se han proporcionado las distintas contribuciones de esta tesis doctoral buscando la reducción del número de peticiones estadísticas en los switches de la red (disminuyendo así el flujo de mensajes al controlador). Por lo tanto, se ha extraído que el problema de optimización en el flujo de mensajes de control en una red SDN es una cuestión muy importante ya que distintas aplicaciones de esta red requieren de una alta demanda de computación y recursos.

El objetivo general de este capítulo es presentar tres arquitecturas para reducir el número de mensajes en una red SDN, especialmente, los mensajes de consultas estadísticas a los switches que componen la red. La Sección 4.1 de este capítulo explica los términos QoS, QoE y el método de identificación de los paquetes de vídeo sobre los cuales se van a realizar las simulaciones de las distintas contribuciones. Esta sección es necesaria para entender cómo se han construido las demás contribuciones ya que es parte común y necesaria de las simulaciones sobre las que se prueban las arquitecturas de monitorización. La primera arquitectura descrita en la Sección 4.2 propone un framework básico de monitorización en el que se monitorizan todos los switches que componen la topología. A continuación, en la Sección 4.3 se presenta un framework de monitorización con optimización usando la ley de conservación de flujo el cual trata de reducir el número de consultas estadísticas a los switches de la topología. La tercera propuesta se encuentra en la Sección 4.4 describiendo un framework cuya monitorización aplica una optimización agrupando los switches de la topología usando agrupamiento para reducir el número de consultas estadísticas. Finalmente, la última contribución presentada en la Sección 4.5 proporciona un análisis de diferentes algoritmos de agrupamiento usando la arquitectura de la tercera contribución presentando los resultados y concluyendo cúal de los algoritmos estudiados es el que reduce el problema de optimización.

4.1. Métricas de Calidad

QoS y QoE son términos que se utilizan a menudo libremente e, incluso a veces, de manera intercambiable [QoS]. QoS realiza el seguimiento de los componentes discretos de la infraestructura de red tales como servidores, routers o el tráfico de red (paquetes IP, flujo de transporte, etc). Además, la suma de la calidad de los componentes no es igual a la calidad de la suma de componentes. En otras palabras, por tener mejores componentes en una red no significa que se comporte de manera mejor que otra con otros componentes de calidad inferior. La única manera de evaluar la calidad que experimenta un usuario es conectarse al servicio como un usuario. Por lo tanto, los indicadores de rendimiento de QoE están centrados en el usuario a la hora de descargar una página web, acceder a una aplicación, realizar una llamada de teléfono, etc.

El procedimiento para la mejora en la calidad de experiencia del usuario consiste en las siguientes fases:

1. Inicio de funciones básicas del controlador.
2. Identificación de los paquetes de vídeo que circulan por la red.
3. Exploración de la topología de la red.
4. Monitorización de las estadísticas o estado actual de la red.
5. Cálculo del camino óptimo.
6. Reconfiguración de los Switches para establecimiento de nueva ruta.

Todas estas fases de las que consta el algoritmo se pueden apreciar en el diagrama de flujo de la Figura 4.1.

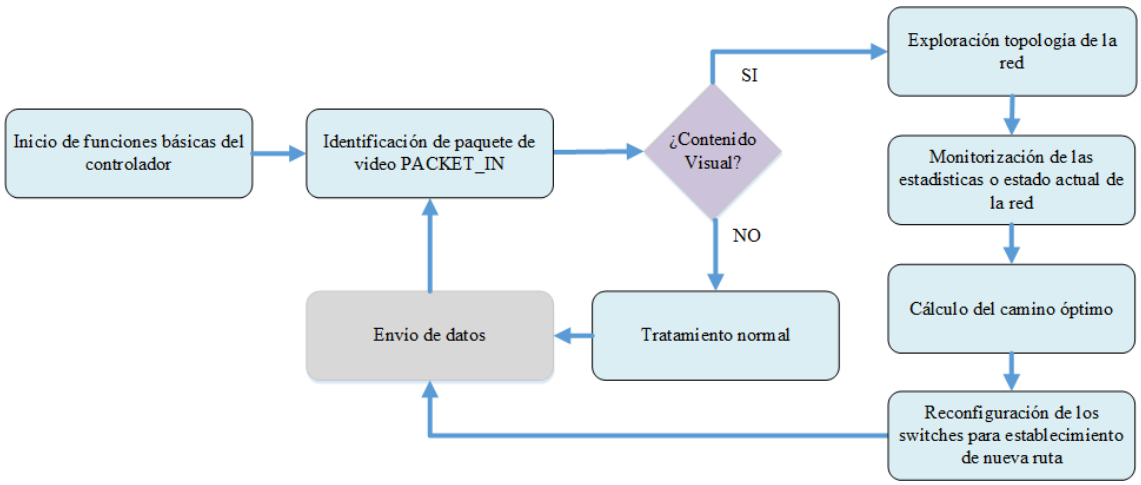


Figura 4.1: Procedimiento de Optimización de la QoE

4.1.1. Inicio de Funciones Básicas del Controlador

Una vez ejecutado Floodlight, se configuran las características deseadas en el controlador.

- **Recepción de PACKET_IN**: se monitorizan todos los paquetes que entran al controlador de este tipo, es decir, paquetes que llegan a un switch y no tienen una regla de flujo asociadas a ellos. Este tipo de paquetes puede ser cualquiera de los que componen los mensajes asíncronos descritos en el Apéndice A.5.1.2.
- **Servicios de Floodlight**: La Tabla 4.1 presenta los servicios que deben activarse.

Tabla 4.1: Características del Vídeo usado en las Simulaciones

Servicio	Descripción
ITopologyService	mantiene la información de la topología que comunica al controlador.
IDeviceService	recorre los dispositivos para saber cómo se encuentran y determina la posición de un dispositivo.
IRoutingService	módulo que calcula rutas entre dos dispositivos dentro de la red OpenFlow.
ILinkDiscoveryService	servicio responsable del descubrimiento y mantenimiento del estado de los enlaces en la red OpenFlow.
IListeners	herramienta de monitorización que cuando recibe un evento se activa y ejecuta la acción que tenga asociada al mismo.
IThreadPoolService	módulo envuelto por un servicio Java de ejecuciones programadas. Puede ser usado para tener hilos de ejecución en tiempos específicos o periódicamente.
Timers	herramienta para lanzar ejecuciones cada cierto período de tiempo.

4.1.2. Identificación de los Paquetes de Vídeo

En este punto el controlador analiza la cabecera del paquete tipo PACKET_IN que entra al controlador y lee el puerto de transporte destino (L4-Transport Source Port). Los campos que pueden ser leídos en un paquete se presentan en la Figura 4.2.

Si el puerto de transporte del paquete es un puerto utilizado para la transmisión de vídeo (en la implementación se utiliza el puerto 5532), el paquete será procesado utilizando el cálculo del camino óptimo para su posterior envío. En caso contrario, el paquete tendrá un procesamiento normal.

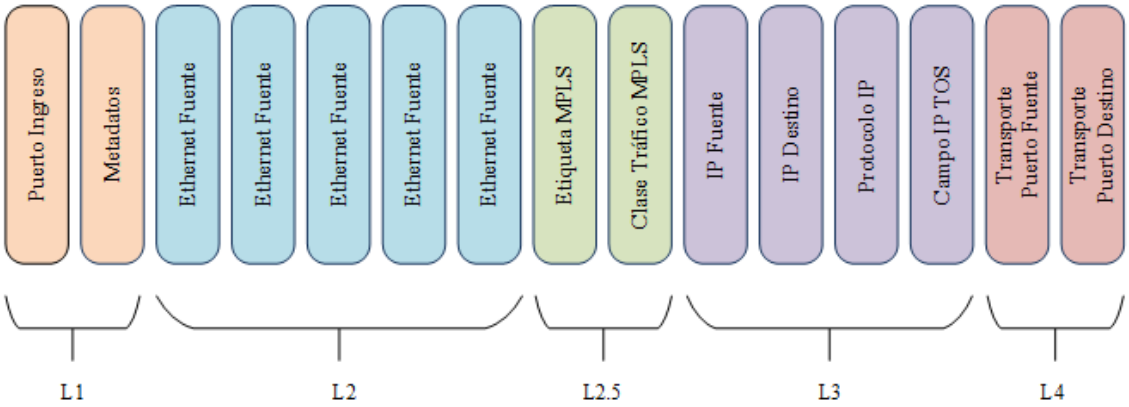


Figura 4.2: Paquete PACKET_IN Recibido por el Controlador OpenFlow

4.1.3. Exploración de la Topología

En esta fase se realiza una comprobación de la topología de red para obtener una visión de todos los dispositivos y las conexiones que la componen. Para ello se utilizan los servicios de Floodlight llamados ILinkDiscoveryService e IDevice (descritos en la Sección 4.1.1). La topología de los enlaces y Switches presentados en el plano de infraestructura se representa como el Grafo 4.1.

$$G(N, A) \tag{4.1}$$

donde N es el conjunto de Switches y A es el conjunto de los arcos (enlaces). Más concretamente, $arc(i, j) \in A$ representa el enlace desde el nodo i hacia el nodo j.

Cada enlace está asociado con un valor llamado peso del enlace o coste del enlace. En este caso, c_{ij} representa el coste del enlace en el $arc(i, j)$.

El coste del enlace representa un valor fijo como, por ejemplo, saltos ($c_{ij} = 1$), una variación de tiempo de un campo (jitter), pérdida de paquetes o el ancho de banda disponible. R_{st} representa el conjunto de todos los caminos o rutas disponibles desde el switch fuente s hasta el switch destino t. En este enfoque se asumen varios supuestos:

- la red es dirigida;
- la red contiene un camino directo desde un switch hacia cualquier otro switch;
- la red no contiene ciclos negativos.

4.1.4. Monitorización de las Estadísticas y Cálculo del Estado de la Red

Las medidas de rendimiento de una red es un gran desafío. El manager del módulo de estadísticas provisto por NOS puede leer la tasa teórica de datos máxima en un puerto particular a través del parámetro OFP_PORT_FEATURES. Sin embargo, la capacidad real o tasa de datos máxima depende de las condiciones y calidad de los enlaces entre

Switches. Por otra parte, OpenFlow 1.0 especifica contadores en el switch en diferentes niveles (tablas, puertos, flujos, colas) pero no provee retardo de paquetes o medidas de jitter. Además, el controlador lee esta información con el inevitable retardo de tiempo del enlace controlador - switch.

Los modelos propuestos en [EDBT12] [DT13] presentan algoritmos para mejorar el diagnóstico de rendimiento para redes OpenFlow. Otra solución es una integración de medidas del plano de datos o herramientas especializadas como sFlow [FHTG10]. Tomando este modelo como referencia, se ha definido el Algoritmo 1 utilizando el coste del módulo de rendimiento de la red.

Algoritmo 1: Función de Rendimiento de la Red

Input: Grafo de la Red $G(N, A)$
Periodo de Monitorización T
Ancho de Banda del enlace $bw(i, j)$ para cada $arc(i, j)$
Ajuste factor α

Result: Coste $c_{ij} \forall arc(i, j) \in A$

1 **procedure** *FUNCION_COSTE*
2 // Inicio temporizador k con periodo t
3 **foreach** *periodo* $k = 0, 1, 2, 3 \in t$ **do**
4 **foreach** $arc(i, j)$ **do**
5 //Lectura de bytes enviados s_i del puerto del enlace inicial
6 $s_i^k = tx\_bytes\ in\ ofp\_port\_stats(node, port(i))$
7 //Lectura de bytes recibidos r_j del puerto del enlace final
8 $r_j^k = rx\_bytes\ in\ ofp\_port_stats(node, port(j))$
9 **if** $k = 0$ *(tiempo inicial)* **then**
10 $c_{ij} = 1$;
11 **if** $k > 0$ *(cada periodo)* **then**
12 //Cálculo de la tasa de datos del enlace
13 $dr_{ij} = \frac{s_i^k - s_i^{k-1}}{t}$
14 //Cálculo de los paquetes perdidos del enlace
15 $pl_{ij} = \frac{s_i^k - r_j^{k-1}}{t}$
16 //Cálculo del coste del enlace
17 $c_{ij} = 1 + \frac{\alpha dr_{ij} + (1-\alpha) pl_{ij}}{bw_{ij}}$
18 **end procedure**

Primero, el módulo lee la tasa máxima de datos de los enlaces (OFP_PORT_FEATURES) y añade esta información como valor inicial en el coste.

Después, el módulo inicia un temporizador para monitorizar continuamente los contadores de estadísticas de los puertos (OFP_PORT_STATS) y coleccionar periódicamente los bytes enviados.

La división de los bytes enviados respecto al período devuelve la tasa de bytes/segundos. Este valor es actualizado como el vector coste de cada enlace.

Con estos datos se calculará el camino óptimo a seguir (punto 0) por los paquetes que

componen el archivo de vídeo.

4.1.5. Cálculo del Camino Mínimo

Hay múltiples técnicas para proveer QoE en arquitecturas de red típicas hop-by-hop [NCC10]. Sin embargo, la visión global de la red provista por SDN y la estrategia basada en flujos OpenFlow puede facilitar nuevos modelos de algoritmos eficientes de routing. En esta Tesis se presenta una solución básica para QoE Routing basado en un grafo de nodos y enlaces. Este procedimiento esta descrito en el Algoritmo 2.

Algoritmo 2: Función de Enrutamiento QoE

Input: Grafo de la Red $G(N, A)$
Coste enlace c_{ij} $\forall$ $arc(i, j) \in A$
Ancho de Banda del enlace $bw(i, j)$ para cada $arc(i, j)$
Ajuste factor α

Result: Coste c_{ij} $\forall arc(i, j) \in A$

1 **procedure** *RUTA_QoS*
2 //packet in recibido del OF-Switch
3 $pi = PACKET\_IN$
4 **if** *pi fulfil QoS parameters* **then**
5 //búsqueda de ruta r entre fuente y destino usando el coste de enlaces
6 $r_{st} = Dijkstra(G, s, t, c_{ij})$
7 //creación de mensajes flowmod F para la ruta r con mayor prioridad
8 $F_{st} = flowmod\_message\_function(r_{st})$
9 //envío de mensajes flowmod F a los OF-switches
10 $write\_flow\_mod(F_{st})$
11 **else**
12 //procesamiento del paquete pi con la mejor estrategia de esfuerzo y menor prioridad
13 **end procedure**

En esta implementación el módulo de QoE Routing lee los parámetros QoE provistos por el plano de aplicación a través de la RestAPI. Por ejemplo, el usuario puede establecer una prioridad alta a la transmisión de vídeo (obteniendo mejor calidad) respecto a otro tipo de datos a enviar. Además, este módulo también recibe el valor de coste c_{ij} del enlace devuelto por el módulo de rendimiento de la red al igual que el grafo $G(N, A)$ del módulo de encargado de la topología. Cuando el controlador recibe un mensaje PACKET_IN, éste analiza la cabecera y establece si el paquete cumple los requisitos QoE. Si es así, el módulo lee las direcciones IP fuente e IP destino y envía esta información al módulo de cálculo del camino mínimo. Esta implementación usa el algoritmo de Dijsktra [NCC10] para establecer la ruta en la red en función del coste provisto por el módulo de rendimiento de la red. La ruta elegida será el camino con el coste mínimo. Esta ruta será instalada en los dispositivos de red a través del manager de flujo de Switches. Dicho manager usa los mensajes flowmod para configurar las tablas de flujo de los Switches con mayor prioridad.

Floodlight provee el servicio IRouting para establecer un camino regular entre Switches fuente y destino.

4.1.6. Reconfiguración de los Switches para Establecimiento de una Nueva Ruta

Una vez obtenida la ruta óptima entre el host fuente y el destino se realiza la programación remota de los Switches. Este proceso se realiza configurando las tablas de flujo de cada switch del camino resultante con un mensaje tipo flowmod. En dicho mensaje se establece que todo paquete, cuyo puerto destino alojado en la cabecera sea 5532, se redirija por el puerto calculado previamente en el algoritmo. Debido a la naturaleza aleatoria de la red, la configuración de los Switches no puede ser estática, por lo que los flujos instalados en el switch necesitan actualizarse periódicamente. Con este fin, las reglas instaladas en el switch tienen un hard_timeout constante. Esto quiere decir que, después de t segundos, el flujo es removido del switch y se envía un mensaje tipo flowremoved al controlador. De esta manera, el controlador recalcula el camino y configura nuevamente los Switches.

4.2. Framework Básico de Monitorización

El framework de monitorización propuesto consiste en diferentes módulos funcionales dentro del plano de control y aplicación de la arquitectura SDN. El plano de datos incluye los switches, routers y otros elementos responsables de las tareas de reenvío de datos. El plano de control toma decisiones sobre el comportamiento de la red y envía estas instrucciones al plano de datos a través de una Southbound API (OF). El plano de control ofrece funcionalidades al plano de aplicación gracias a una Northbound API (*Representational State Transfer* (REST) API), donde se implementan políticas y aplicaciones de alto nivel. La Figura 4.3 muestra los componentes del framework propuesto.

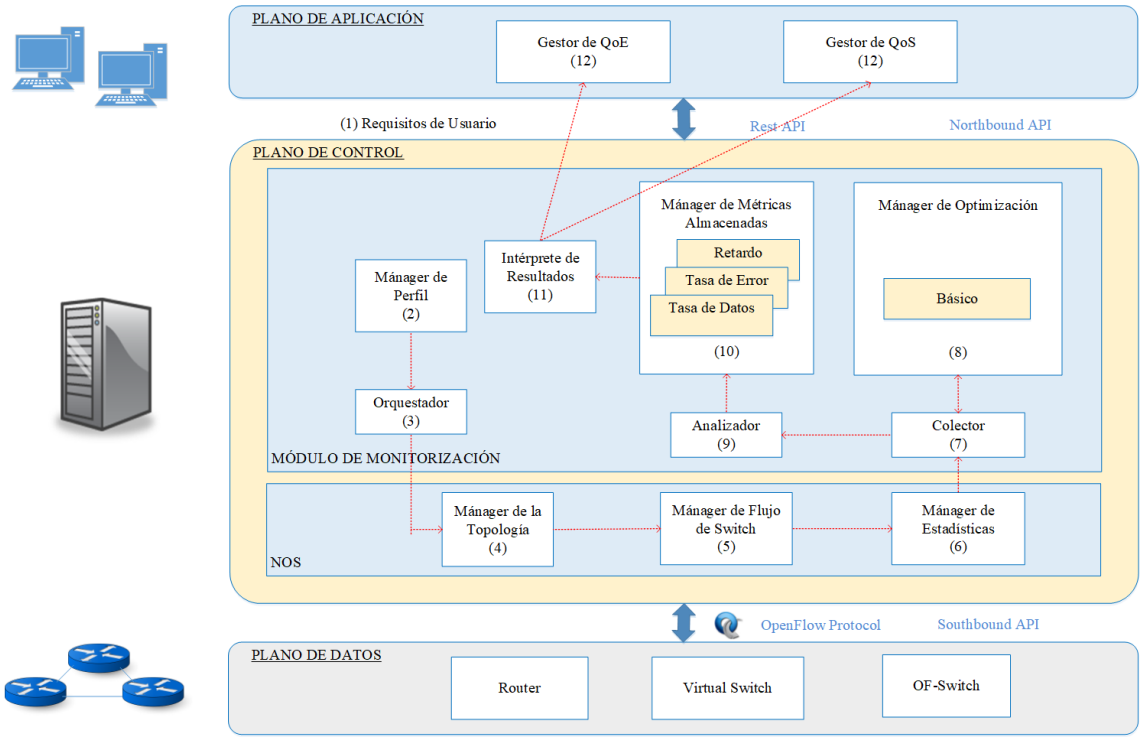


Figura 4.3: Framework de Monitorización Básico

4.2.1. Componentes Funcionales

Los principales componentes funcionales del framework son los siguientes:

1. **Requisitos de Usuario:** La aplicación que necesita información sobre la red crea un nuevo perfil de monitorización. En este perfil, el usuario (persona encargada de configurar los parámetros de la monitorización) especifica el ID (identificador) de usuario, el tipo de métrica y el nivel de precisión. Esta información se envía al plano de control a través de una Northbound API (REST API). Por tanto, se garantiza que la aplicación recibirá información actualizada del rendimiento de la red en función de las métricas requeridas. El usuario puede cambiar la configuración del perfil sin afectar a otros módulos o cambiar el comportamiento de la red.

2. **Mánager de Perfil:** Este modulo recibe los diferentes requisitos del plano de aplicación, crea un perfil de monitorización para cada usuario, registra las solicitudes de módulos ubicados dentro del plano de control (módulos internos) a través de una API interna y actualiza el módulo Intérprete de Resultados para sincronizar y proporcionar precisión en los resultados enviados a los usuarios.

3. **Orquestador:** Este módulo organiza y programa los diferentes módulos de monitorización en función de los requisitos del Mánager de Perfiles. Para ello, tiene

en cuenta la información proporcionada por el Mánager de la Topología y equilibra la carga de solicitudes en el plano de infraestructura. Este módulo también trata de reducir la duplicación de solicitudes y evita la redundancia de datos.

4. **Mánager de la Topología:** Este módulo utiliza el *Link Layer Discovery Protocol* (LLDP) y analiza los mensajes de OF para identificar los dispositivos de red, sus capacidades y los enlaces presentados en el plano de infraestructura. La topología se organiza en un grafo $G(N, A)$, donde N representa los switches y A los enlaces presentes en la infraestructura. La información de la topología se envía a otros módulos del plano de control.

5. **Mánager de Flujo de Switch:** Recibe las instrucciones de los algoritmos de monitorización organizados por el Orquestador y envía los mensajes de OF para solicitar información a los switches. Además, este módulo utiliza la información del Mánager de la Topología y modifica las tablas de flujo en caso de establecer una ruta para paquetes de sonda.

6. **Mánager de Estadísticas:** Recibe la información proporcionada por los mensajes de OF y recupera la información estadística de estos mensajes. Esta información se envía al módulo Colector.

7. **Colector:** Este módulo filtra y organiza la información proporcionada por el Mánager de Estadísticas y almacena la información relevante. Este módulo puede usar algoritmos de minería de datos y algoritmos de búsqueda para reducir el espacio requerido para almacenar los datos.

8. **Mánager de Optimización:** Este módulo contiene las técnicas de optimización para ser aplicados en la petición de Estadísticas en los dispositivos del plano de datos. Están implementadas mediante algoritmos dependiendo de la estrategia que usan para disminuir las peticiones de estadísticas.

9. **Analizador:** Lee los resultados del Colector y realiza el procesamiento de eventos, la correlación de datos y los algoritmos de aprendizaje para proporcionar métricas de alto nivel. Las métricas de alto nivel se centran no solo en dispositivos individuales (switch), sino también en una vista global o en una agrupación compleja distribuida de dispositivos (dominio de red). Del mismo modo, el Analizador puede inferir tendencias y valores pronosticados sobre el comportamiento futuro de la red.

10. **Mánager de Métricas Almacenadas:** Consiste en un contenedor que registra y almacena los algoritmos de monitorización disponibles. Estos algoritmos son modulables y se pueden agregar, modificar y actualizar fácilmente. Este Manager puede incluir módulos propios o de terceros. Los módulos usan OF como Southbound API y pueden también incluir API externas para recibir información de otras herramientas de monitorización externas (NetFlow [Cla04], SFlow [PL04]).

11. **Intérprete de Resultados:** Este módulo recibe los resultados del módulo Analizador e infiere eventos de *Health of Network* (HoN) tales como violaciones de *Access Control List* (ACL) actuales y futuras, alertas de IDS/firewall, ataques DDoS, detección de intrusiones o comportamiento anómalos. Además, decide si ha habido un aumento/disminución del tráfico, y puede solicitar un mayor ancho de banda si es necesario.

12. **Aplicaciones:** Las diferentes aplicaciones de red reciben solo la información relevante de monitorización de alto nivel. Los usuarios pueden cambiar el perfil del monitor de acuerdo con sus necesidades y modificar dinámicamente las solicitudes en los switches. Además, el usuario puede aplicar acciones reactivas y proactivas proporcionando respuestas dinámicas de diferentes eventos de red.

4.2.2. Simulación y Resultados

La implementación de la arquitectura de monitorización básica se ha evaluado mediante la simulación de una topología compuesta por 3 switches OpenFlow (s_1, s_2 y s_3) y 2 host (h_1 y h_2) conectados a s_1 y s_3 respectivamente. Los enlaces [switch: s_1 - puerto: p_2, switch: s_2 - puerto: p_1] y [switch: s_2 - puerto: p_2, switch: s_3 - puerto: p_2] están configurados con valores de velocidad de datos máxima (1 Mbps de ancho de banda) y un porcentaje de pérdida (5 % en el enlace s_1-s_2 y 10 % en el enlace s_2-s_3). Esta configuración de los enlaces se traduce en una pérdida del tanto por ciento configurado de los paquetes que circulan a través de ellos (se utiliza una pérdida de datos para asemejarlo a la vida real ya que dicha pérdida se puede producir por diversos motivos como por ejemplo errores físicos de la red, apagado de la red, etc..). La Figura 4.4 describe esta topología.

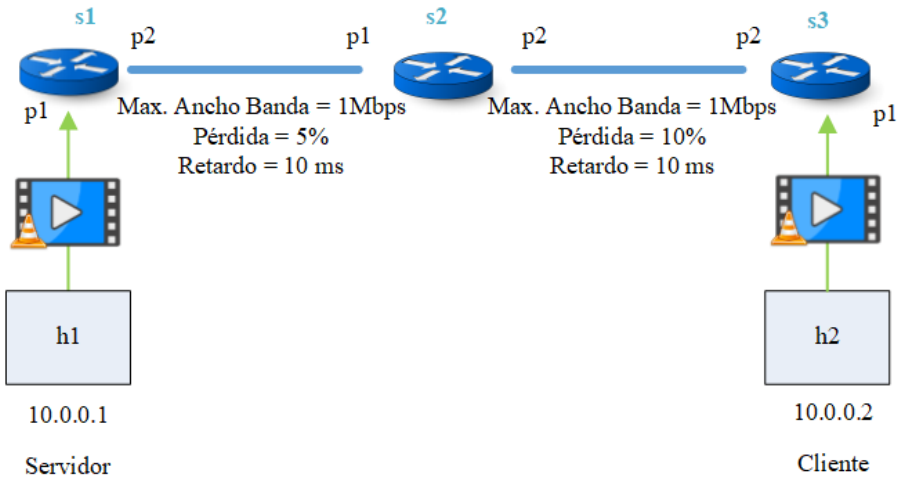


Figura 4.4: Topología de Prueba Inicial

Todas las simulaciones se ejecutaron en un simulador de red llamado Mininet v2.1.0

[min17] utilizando scripts de Python, que contiene la topología para examinar descrita anteriormente. Mininet puede crear con pocos pasos topologías personalizadas para simular en un ordenador. El conjunto de prueba se ejecutó en una máquina virtual dentro de un servidor modelo TD350 de ThinkServer que está compuesto por un Intel Xeon Series E5-2600 v3 con 12 núcleos, 64 GB de RAM DDR4 y Ubuntu 14.04 como sistema operativo. Dentro del servidor, se encuentra la máquina virtual que se compone de un sistema operativo invitado Linux (Ubuntu v. 13.04) (64 bits, 8 GB de RAM, 2 CPU, 8 GB de HD). Los módulos de monitorización de agrupamiento y conservación de flujo se implementan utilizando el controlador Floodlight [Flo17] basado en Java dentro de dicha máquina virtual.

Una simulación consiste en una transmisión de vídeo enviada desde el host 1 al host 2 usando el servidor de vídeo VLC y RTP/UDP como protocolo de transmisión al mismo tiempo que se ejecuta el módulo de monitorización. Por lo tanto, la tarea principal del módulo de monitorización es medir las estadísticas de los switches de red (tanto los enlaces donde se envía el vídeo como el resto de ellos). Utilizamos el vídeo "Highway_cif" [vid17d] para la transmisión de vídeo donde sus características se recopilan en la Tabla 4.2 mientras que el tiempo de monitorización (t_{mon}) es de 200 ms.

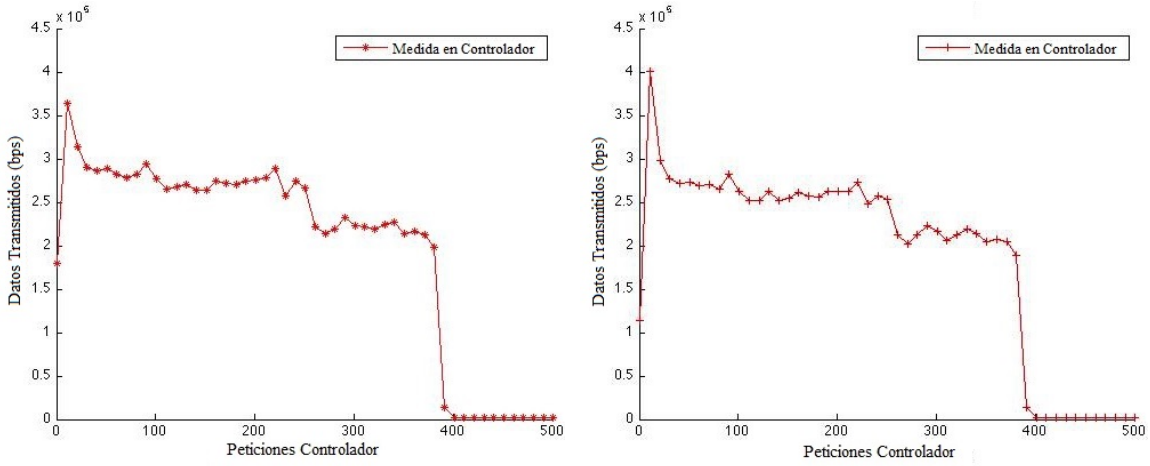
Tabla 4.2: Características del Vídeo usado en las Simulaciones

Vídeo	Formato	Frames	Duración (segundos)	Tamaño	Tasa de Datos (kbits/s)
Highway_cif	MPEG 12	2000	80	2.5 MB	257.4

Las pruebas se han repetido dos veces en las mismas condiciones, con un periodo de monitorización t_{mon} de 400 ms monitorizando todos los switches. La duración de las simulaciones es de 80 segundos.

Las Figuras 4.5a y 4.5b muestran el flujo de tráfico entre dos simulaciones en las que se monitorizan todos los switches de la topología. La Figura 4.5a describe el flujo de datos (en bps) a través de los enlaces s_1-s_2 (switch: s_1 - puerto: p_2, switch: s_2 - puerto: p_1) y la Figura 4.5b los enlaces entre s_2-s_3 (switch: s_2 - puerto: p_2, switch: s_3 - puerto: p_2). Como se esperaba, ambos enlaces experimentan un aumento en la tasa de datos debido a la transmisión del vídeo entre h_1 y h_2. Tan pronto como finaliza la transmisión (alrededor de 200 peticiones), la tasa de datos disminuye, lo que demuestra la eficiencia del algoritmo para detectar cambios en la transmisión de la red. De manera similar, la diferencia promedio entre los enlaces s_1-s_2 con/sin la mejora es de 45,49 Kbps y 54,33 Kbps en el enlace s_2-s_3. Estos resultados confirman que la mejora mantiene buenos niveles de precisión y reduce el número de peticiones en el plano de datos.

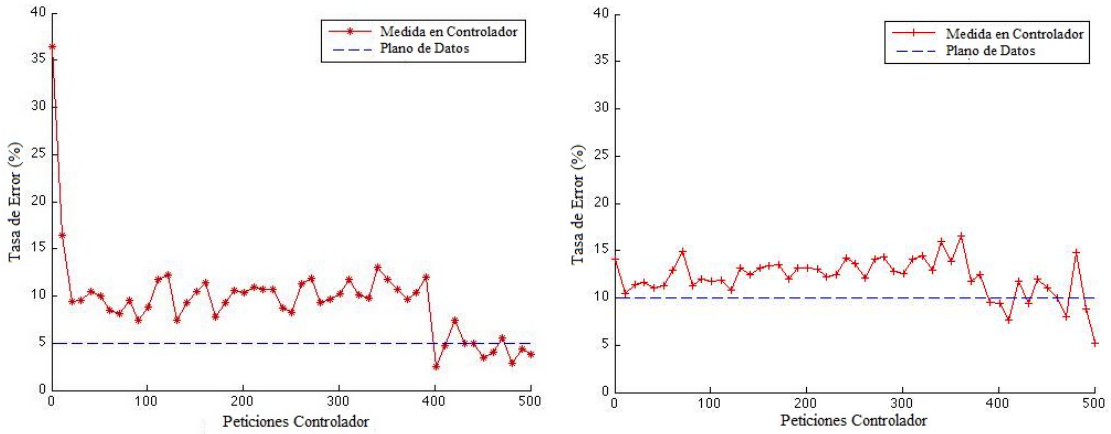
Además, la tasa de pérdida de los enlaces s_1-s_2 y s_2-s_3 se representa en las Figuras 4.6a y 4.6b. La línea roja (continua) representa el porcentaje de tasa de pérdida medido por el controlador, mientras que la línea azul (línea de puntos) muestra el porcentaje netem configurado en mininet o plano de datos (s_1-s_2 = 5 %, s_2-s_3 = 10 %). En este caso,



(a) Tasa de Datos del enlace $s_1 - s_2$ (b) Tasa de Datos del enlace $s_2 - s_3$

Figura 4.5: Tasa de Datos de Monitorización de todos los Switches

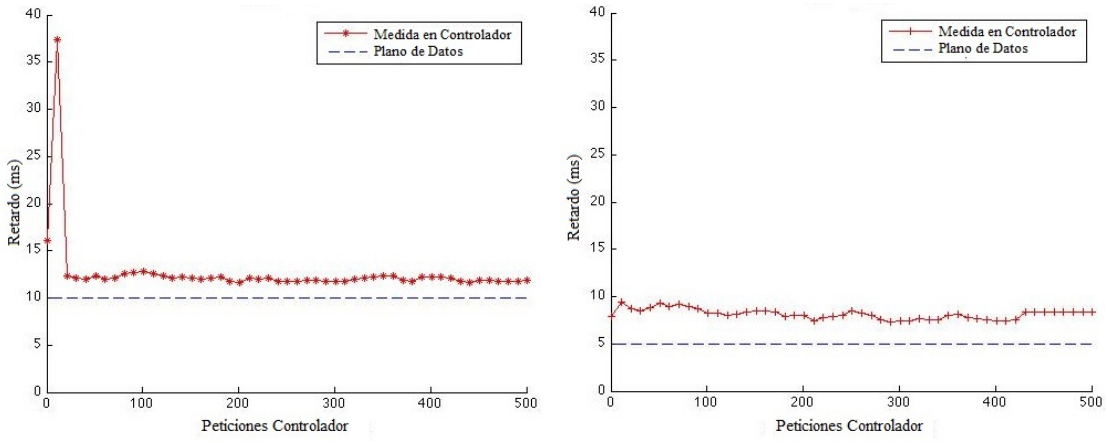
el controlador también es capaz de detectar la pérdida de información entre enlaces. Sin embargo, la información aumenta la variación entre el plano de datos y el valor estimado por el controlador en la transmisión de vídeo.



(a) Tasa de Error del enlace $s_1 - s_2$ (b) Tasa de Error del enlace $s_2 - s_3$

Figura 4.6: Tasa de Error de Monitorización de todos los Switches

La estimación del retraso del controlador para s_1-s_2 y s_2-s_3 se muestra en las Figuras 4.7a y 4.7b. La línea azul (línea de puntos) muestra el retraso del plano de datos de s_1-s_2 (10 ms) y s_2-s_3 (5 ms) y la línea roja (continua) muestra el valor medido por controlador. En ambos casos, se puede ver que el retraso medido es ligeramente mayor en comparación con el plano de datos Este efecto es causado por la latencia adicional entre el switch y el controlador. En otras palabras, por ejemplo, los paquetes de sonda en s_1-s_2 se mueven del controlador para cambiar s1, luego de s_1 a s_2 y el El switch s_2 envía el paquete de vuelta al controlador. Esta variación se puede minimizar calculando el retraso presente entre el controlador y el switch utilizando el mismo procedimiento activo (enviar y recibir un paquete de sonda del mismo nodo) [vADK14]. Sin embargo, esta propuesta aumentará no solo el tráfico en el switch-controlador ruta de datos, pero también las tareas de procesamiento de switch y controlador.



(a) Retardo del enlace $s_1 - s_2$

(b) Retardo del enlace $s_2 - s_3$

Figura 4.7: Retardo en la Monitorización de todos los Switches

4.3. Framework de Monitorización basado en la Ley de Conservación Flujo

El objetivo de esta contribución es buscar mejores resultados con valores más precisos al mismo tiempo que optimizamos la computabilidad de la red respecto a la contribución anterior. Proponemos un framework de monitorización optimizado basado en el algoritmo de Conservación de Flujo como se ilustra en la Figura 4.8.

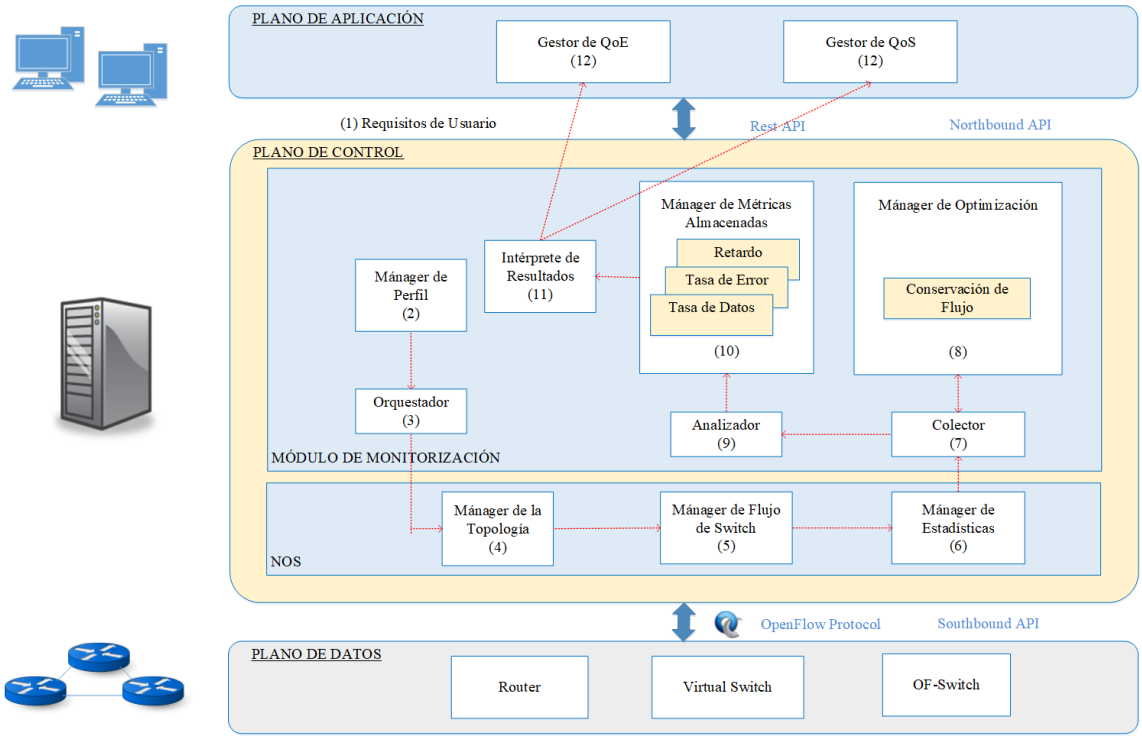


Figura 4.8: Framework de Monitorización baso en la Ley de Conservación de Flujo

La mayoría de las estrategias de monitorización solicitan el estado de todos los puertos en los dispositivos de red. Este método para obtener información de red es muy preciso pero al mismo tiempo demasiado ineficiente. Por lo tanto, para reducir las consultas de monitorización en los recursos, proponemos no monitorizar los switches que contienen solo dos puertos dentro de ellos como se explica en el Algoritmo 3. En cada período de monitorización t_{mon}, $FUNCION\_OPTIMIZACION$ recorre todos los switches que componen la topología y verifica a cuántos switches vecinos están conectados con él: si solo tiene dos switches adyacentes ($deg(s_i) == 2$ puerto entrante, puerto saliente), no se solicita estadísticas, de lo contrario, exige el estado del switch con $CALCULAR\_ESTADISTICAS$.

Algoritmo 3: Función de Optimización de la Topología

Input: Grafo Red $G(S, L)$
Result: Grafo Red Optimizado $G(S, L)$

```
1 procedure FUNCION_OPTIMIZACION
2     foreach switch s_i ∈ S do
        //Comprueba si el número de puertos entrantes y salientes del enlace es 2
3         if (deg(s_i) == 2 && not(contains_host(s_i)) then
4             CALCULA_ESTADISTICAS_OPTIMIZADAS(s_i);
5         else
6             CALCULA_ESTADISTICAS(s_i);
7 end procedure
```

El cálculo de los switches optimizados depende de los enlaces de sus switches adyacentes. El Algoritmo 4 describe el procedimiento para su calculo. Establece el valor de la tasa de datos en el puerto j del switch adyacente s_k que está conectado con él (switch s_i). Hay que considerar que el grafo de la red G es dirigido, por lo que todos los flujos de paquetes entrantes F_i del switch s_i (llegan al puerto entrante I) se enviarán al otro puerto (puerto saliente O). Esta afirmación satisface la ley de conservación de flujo que describe que la suma del flujo de tráfico que entra en un switch s_i es aproximadamente la misma que la suma del tráfico que sale de sí en un nodo de grado 2. Formalmente, la ley de conservación de flujo se establece como Ecuación 4.2.

$$\sum_{j=1}^{|F|} F_j(I_{S_i}) - \sum_{j=1}^{|F|} F_j(O_{S_i}) \approx 0 \tag{4.2}$$

Por esta razón, omitimos la solicitud de monitorización de este switch y luego reducimos el número total de consultas en cada período de monitorización t_{mon}. Estas solicitudes se envían desde el controlador OpenFlow a través del canal seguro en un mensaje llamado $OFPT\_STATS\_REQUEST$. Una vez que el switch recibe este mensaje, responde su estado al controlador con otro mensaje llamado $OFPT\_STATS\_REPLY$.

Algoritmo 4: Función de Cálculo de Estadísticas Optimizadas

Input: Grafo Red $G(S, L)$
Result: Grafo Red Optimizado $G(S, L)$

1 **procedure** $CALCULA\_ESTADISTICAS\_OPTIMIZADAS$
2 **foreach** *port* $s_j \in s_i$ **do**
//Lee los bytes enviados por el puerto adyacente j del switch adyacente k en un periodo de tiempo t con
3 s_{ij}^t = bytes in OFPT_STATS_REQUEST$(s_k, port(j))$
4 **if** $(t > t_{init})$ **then**
5 $d_{sij} = \frac{s_{ij}^t - s_{ij}^{t-1}}{t_{mon}}$;
6 **else**
7 $CALCULA\_ESTADISTICAS(s_i)$;
8 **end procedure**

4.3.1. Simulaciones y Resultados

Para evaluar la viabilidad de la técnica mejorada aplicando el algoritmo de conservación de flujo respecto a la monitorización de todos los switches se realizarion simulaciones utilizando la topología descrita en la Figura 4.9.

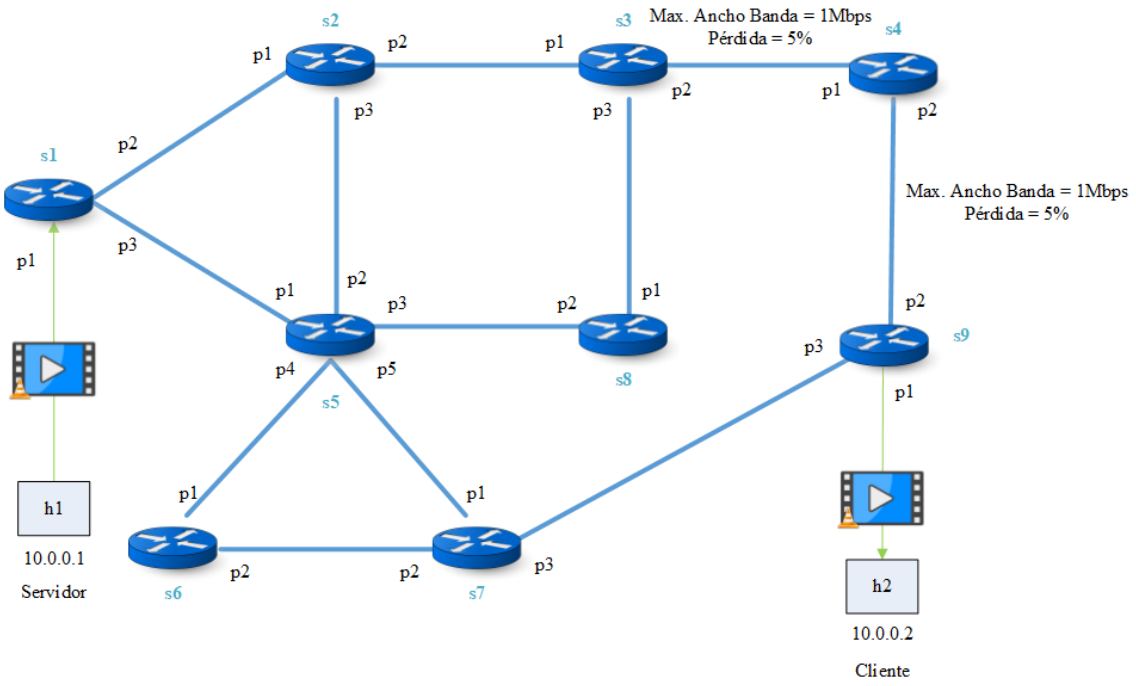


Figura 4.9: Topología de Prueba de la Optimización de Flujo de Conservación

La topología se compone de 9 switches OpenFlow ($s_1, s_2, s_3, s_4, s_5, s_6$, s_7, s_8 y s_9) y 2 host (h_1 y h_2) conectados a s_1 y s_9 respectivamente. Los enlaces [switch: s_3 - puerto: p_2, switch: s_4 - puerto: p_1] y [switch: s_4 - puerto: p_2, switch: s_9 - puerto: p_2] están configurados con valores de velocidad de datos máxima (1 Mbps de ancho de banda) y porcentaje de pérdida (5 % de los paquetes se perderán).

Todas las simulaciones se ejecutaron en un simulador de red llamado Mininet v2.1.0 [min17] utilizando scripts de Python, que contiene la topología para examinar descrita anteriormente. Mininet puede crear con pocos pasos topologías personalizadas para simular en un ordenador. El conjunto de prueba se ejecutó en una máquina virtual dentro de un servidor modelo TD350 de ThinkServer que está compuesto por un Intel Xeon Series E5-2600 v3 con 12 núcleos, 64 GB de RAM DDR4 y Ubuntu 14.04 como sistema operativo. Dentro del servidor, se encuentra la máquina virtual que se compone de un sistema operativo invitado Linux (Ubuntu v. 13.04) (64 bits, 8 GB de RAM, 2 CPU, 8 GB de HD). El módulo de monitorización de conservación de flujo se implementa utilizando el controlador Floodlight [Flo17] basado en Java dentro de dicha máquina virtual.

Una simulación consiste en una transmisión de vídeo enviada desde el host 1 al host 2 usando el servidor de vídeo VLC y RTP/UDP como protocolo de transmisión al mismo tiempo que se ejecuta el módulo de monitorización. Por lo tanto, la tarea principal del módulo de monitorización es medir las estadísticas de los switches de red (tanto los enlaces donde se envía el vídeo como el resto de ellos). Utilizamos el vídeo "Highway_cif" [vid17d] para la transmisión de vídeo donde sus características se recopilan en la Tabla 4.2 mientras que el tiempo de monitorización (t_{mon}) es de 200 ms. La conexión entre el host cliente y el host servidor es una ruta fija compuesta de [switch: s_1 - puerto: p_1, switch: s_1 - puerto: p_2, switch: s_2 - puerto: p_1, switch: s_2 - puerto: p_2, switch: s_3 - puerto: p_1, switch: s_3 - puerto: p_2, switch: s_4 - puerto: p_1, switch: s_4 - puerto: p_2, switch: s_9 - puerto: p_2, switch: s_9 - puerto: p_1].

Las pruebas se han repetido dos veces en las mismas condiciones, con un periodo de monitorización t_{mon} de 200 ms cambiando dicho periodo en cada uno. En la primera simulación se ha aplicado la optimización propuesta en esta sección mientras que en la otra aplica el método propuesto en [VCPFGV17] el cual monitoriza todos los switches. La duración de las simulaciones es de 80 segundos.

Los resultados de las simulaciones en la que se extrae la tasa de datos como métrica, el número de peticiones a los switches usando como método la monitorización de todos éstos es de 5115 peticiones, mientras que usando el algoritmo de flujo de conservación es de 3454. La diferencia de 1661 peticiones muestra una reducción del 33 % de peticiones con respecto a una simulación monitorizando todos los switches como se muestra en la Tabla 4.3.

Tabla 4.3: Resultados usando Flujo de Conservación respecto a Todos los Switches para Tasa de Datos

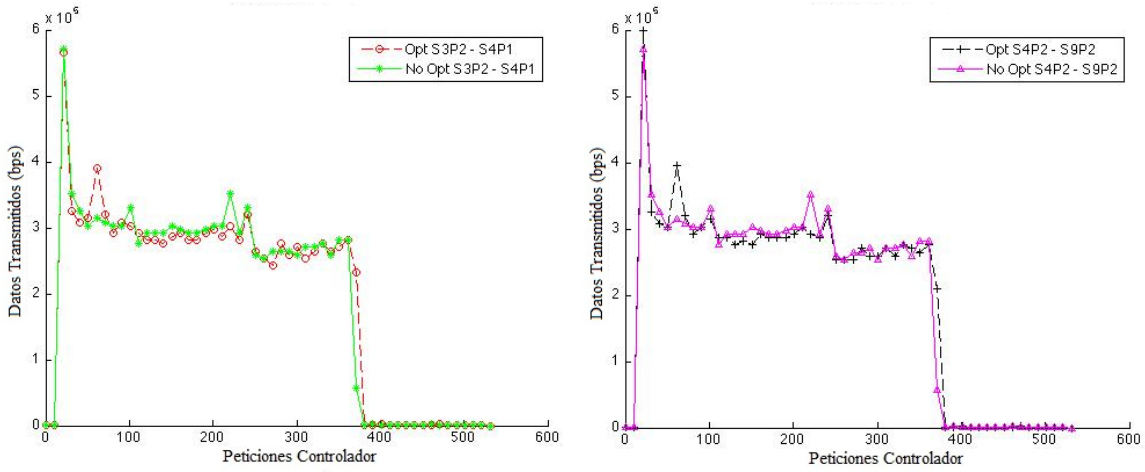
Método	No. de Peticiones	Reducción	Reducción (%)	Enlace	Diferencia
Flujo de Conservación	3454	1661	33 %	Enlace s_3-s_4	30.48 Kbps
Todos los switches	5115			Enlace s_4-s_9	30.66 Kbps

Desde el punto de vista de los valores de la tasa de datos, la diferencia de los valores obtenidos usando la optimización de la Ley de la Conservación de Flujo y monitorizando todos los switches es la siguiente: para el enlace s_3-s_4 la diferencia es de 30,48 KB y para el enlace s_4-s_9 es de 30,66 KB por lo que la diferencia es insignificante. Los resultados de las simulaciones se presentan a continuación.

La Figuras 4.10a y 4.10b muestran el flujo de tráfico entre dos simulaciones en las que una aplica la optimización del Flujo de Conservación respecto a otra que no la utiliza. La Figura 4.10a describe el flujo de datos (en bps) a través de los enlaces s_3-s_6 (switch: s_3 - puerto: p_4, switch: s_6 - puerto: p_1) y la Figura 4.10b los enlaces entre s_6-s_5 (switch: s_6 - puerto: p_2, switch: s_5 - puerto: p_3). Las líneas puntiagudas (líneas azules) muestran la tasa de datos que el servidor está enviando el vídeo, mientras que las líneas continuas (líneas verdes) muestran la tasa de datos obtenida utilizando el método no mejorado y las líneas de puntos (líneas rojas) muestran la tasa de datos con Algoritmo mejorado. Como se esperaba, ambos enlaces experimentan un aumento en la tasa de datos debido a la transmisión del vídeo entre h_1 y h_2. Tan pronto como finaliza la transmisión (alrededor de 200 peticiones), la tasa de datos disminuye, lo que demuestra la eficiencia del algoritmo para detectar cambios en la transmisión de la red. De manera similar, la diferencia promedio entre los enlaces s_3-s_6 con/sin la mejora es de 45,49 Kbps y 54,33 Kbps en el enlace s_6-s_5 como describe en la Tabla 4.3. Estos resultados confirman que la mejora mantiene buenos niveles de precisión y reduce el número de peticiones en el plano de datos.

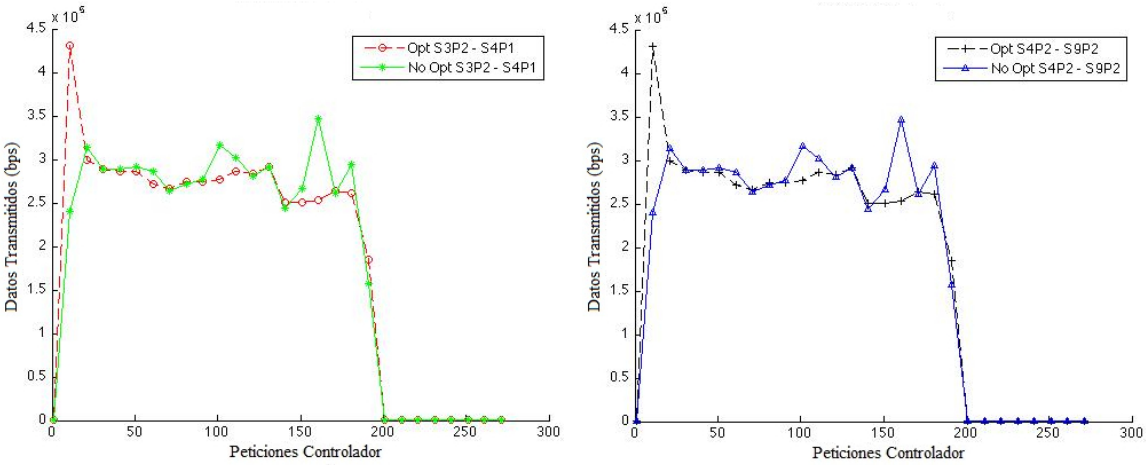
Por lo tanto, según el diseño de nuestro algoritmo, podemos estimar aproximadamente que se logra una reducción de mensajes de peticiones del 33 % en una topología similar al experimento. Esta reducción pertenece a la proporción entre los mensajes de estadísticas contabilizadas en los enlaces $s_3 - s_4$ y $s_4 - s_9$ de las simulaciones mejoradas y no mejoradas.

Por su parte, la Figura 4.11a y la Figura 4.11b muestran el flujo de tráfico entre dos simulaciones en las que una aplica la optimización del Flujo de Conservación respecto a otra que no la utiliza con un periodo de monitorización de 400ms. La Figura 4.11a describe el flujo de datos (en bps) a través de los enlaces s_3-s_6 (switch: s_3 - puerto: p_4, switch: s_6 - puerto: p_1) y la Figura 4.11b los enlaces entre s_6-s_5 (switch: s_6 - puerto: p_2, switch: s_5 - puerto: p_3)



(a) Tasa de Datos del enlace $s_3 - s_4$ (b) Tasa de Datos del enlace $s_4 - s_9$

Figura 4.10: Tasa de Datos de Monitorización basada en Flujo de Conservación respecto a Monitorización de todos los Switches



(a) Tasa de Datos del enlace $s_3 - s_4$ (b) Tasa de Datos del enlace $s_4 - s_9$

Figura 4.11: Tasa de Datos de Monitorización basada en Flujo de Conservación respecto a Monitorización de todos los Switches

4.4. Framework de Monitorización basado en Agrupamiento

Siguiendo el objetivo de esta tesis doctoral, se propone la siguiente contribución buscando mejorar los resultados de la arquitectura basada en la Ley de Conservación de Flujo. El framework de monitorización optimizado basado en el algoritmo de conservación de flujo propuesto se ilustra en la Figura 4.12.

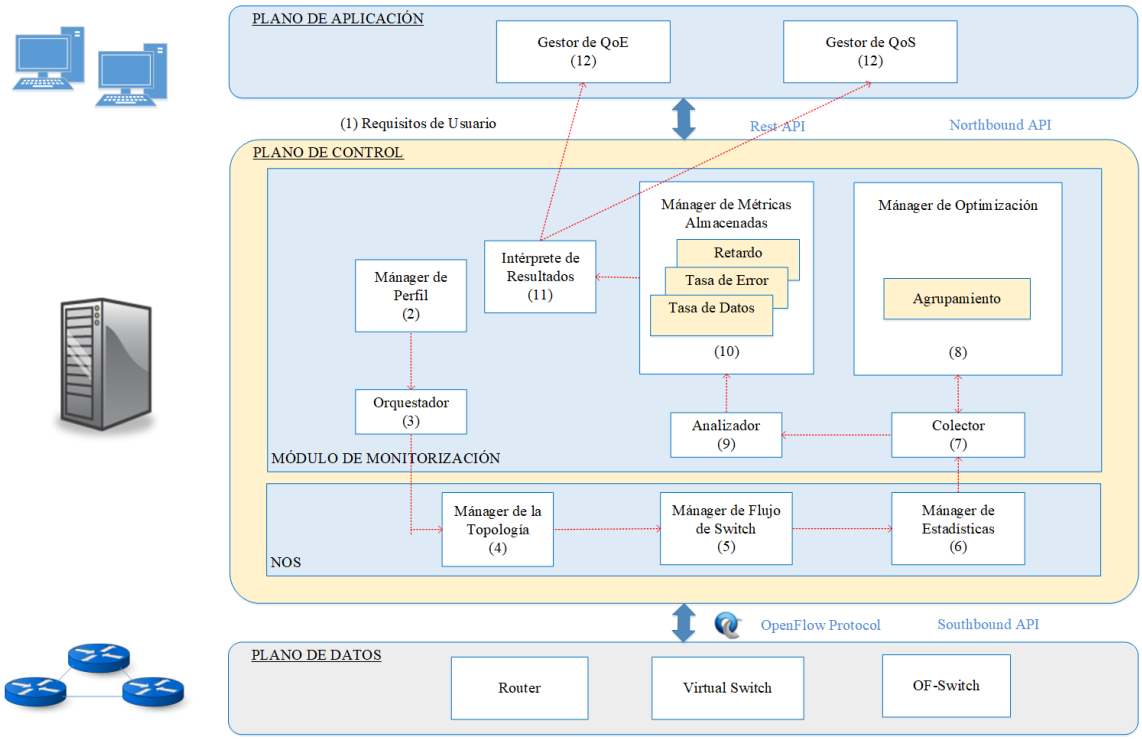


Figura 4.12: Framework de Monitorización con Agrupamiento

El alto rendimiento del contenido de un vídeo y otras aplicaciones es una consecuencia del estado y los componentes de la red. En otras palabras, si la carga de la red no es tan alta, el controlador equilibrará sus recursos con la aplicación o el servicio que demanda dichos recursos. Debido a esto, las herramientas de monitorización son importantes en redes con grandes demandas de computación. Para obtener el mejor rendimiento y respuesta de la red, es importante no cargar la CPU y la memoria del controlador con mensajes de consultas estadísticas. De esta forma, analizamos la topología de la red y buscamos estrategias para mejorar estas consultas estadísticas. Las principales mejoras del framework son el algoritmo mejorado basado en el flujo de conservación y la tecnica mejorada de agrupamiento descritos a continuación.

4.4.1. Notación de la Red

Nuestro modelo abstracto de red se recopila en la Tabla 4.4. La topología de la red se representa como un gráfico dirigido $G = (S, L)$ donde $S = \{s_1, s_2, ..., s_{ns}\}$ denota el conjunto de switches y $L = \{l_1, l_2, .., l_{nl}\}$ representa el conjunto de enlaces entre switches. Además, el conjunto de grupos se denota como $C = \{c_0, c_1, ..., c_{nc-1}\}$ donde cada grupo es al mismo tiempo un conjunto de switches. Por ejemplo, el $arc(i, j)$ representa el enlace del switch origen i al switch destino j. Además, $ns = |S|$, $nl = |L|$ y $nc = |C|$ denotan el número de switches, enlaces y grupos en la topología respectivamente. Por último,

definimos el grado de un switch (número de enlaces incidentes) como $deg(s_i)$ donde $s_i \in S$. Suponemos que ns, nl y nc son finitos y $deg(s_i) > 0$.

Tabla 4.4: Modelo de Notación de Red

Símbolo	Descripción
$G = (S, L)$	Grafo de la Red
S	Conjunto de Switches en el Grafo de la Red G
L	Conjunto de Enlaces en el Grafo de la Red G
C	Conjunto de Grupos que contienen switches
$s_1, s_2, ..., s_n$	Switches Genéricos en el Grafo de la Red G
$l_1, l_2, ..., l_n$	Enlaces Genéricos en el Grafo de la Red G
$c_0, c_1, ..., c_{nc-1}$	Grupos Genéricos en el Grafo de la Red G
$ns = \|\|S\|\|$	Número de Switches en el Grafo de la Red G
$nl = \|\|L\|\|$	Número de Enlaces en el Grafo de la Red G
$nc = \|\|C\|\|$	Número de Grupos en el Grafo de la Red G
$arc(i, j) \in L$	Enlace desde switch fuente s_i al switch destino s_j
$deg(s_i)$	Número de enlaces en un switch s_i

Además, suponemos que los datos monitorizados se agrupan en flujos de paquetes F. Por lo tanto, todos los paquetes de datos que se envían en G pertenecen al menos a un flujo de paquetes $F_i \in F$. Cada F_i tiene una ruta desde su switch origen s_i a su switch destino s_j.

4.4.2. Algoritmo Mejorado basado en Flujo de Conservación

Una de las mejoras utilizadas en esta propuesta es el framework para monitorizar redes SDN descrito en la Sección 4.3. Este framework divide el plano de control en varios módulos para realizar diferentes operaciones y proporciona estadísticas de red al plano de aplicación. De esta manera, la técnica realizada por estos módulos reduce el número de consultas estadísticas que realiza el controlador a los switches de la topología. Dicha reducción se debe a la utilización del algoritmo de conservación de flujo en switches de 2 grados, ya que todo el tráfico de datos que recibe el puerto entrante del switch se reenviará al puerto saliente del mismo switch y, por lo tanto, el tráfico a través de dichos enlaces será casi el mismo.

4.4.3. Técnica Mejorada de Agrupamiento

La otra técnica de mejora de monitorización propuesta en este capítulo se basa en métodos de agrupamiento utilizando agrupamiento. El término agrupamiento corresponde a un procedimiento que agrupa un conjunto de objetos en varios subconjuntos de manera que los objetos comparten una o más características en común. En nuestro caso, la característica común entre todos los switches de S será su número de puertos y su valor crítico.

La aplicación de métodos de agrupamiento en un conjunto de objetos a estudiar ofrece varios beneficios en términos de rendimiento, escalabilidad y gestión, entre otros. Estos beneficios se traducen en este trabajo de investigación de la siguiente manera: en primer lugar, el uso de métodos de agrupamiento para monitorizar estadísticas aumenta el rendimiento en la infraestructura SDN debido a la reducción del flujo de tráfico al controlador SDN. En segundo lugar, la aplicación de nuestro algoritmo mejorado de agrupamiento ofrece escalabilidad ya que no importa la cantidad de dispositivos de red (por ejemplo, una gran red SDN con más de un único controlador SDN) en la infraestructura. Finalmente, el uso de métodos de agrupamiento ofrece una simplificación en la administración de los switches, ya que switches similares pueden gestionarse de la misma manera.

La decisión de utilizar el número de puertos como característica común de los switches de red se ha alcanzado debido a la probabilidad de la densidad de tráfico que fluye dentro de dichos switches. Además, el número de puertos de un switch implica directamente en la probabilidad del posible número de rutas para reenviar los datos entrantes dentro de ellos. Por lo tanto, a mayor número de puertos, la probabilidad de que el switch se utilice en una ruta de datos es mayor debido a la cantidad de posibles switches destino que están conectados a sí mismo. Es importante aclarar que, como paso inicial de esta propuesta, los autores han intentado encontrar una solución para las infraestructuras de red SDN puras.

Antes de comenzar a solicitar estadísticas a los switches, $Clustering\_Function(G)$ lee la topología de red estructurada en un grafo $G = (S, L)$ que se utilizará para crear los grupos. Como G contiene el grafo de la red, la función $Create\_Cluster(G, Algorithm, N)$ lo divide en N grupos, definidos como $c_0, c_1, ..., c_{nc-1}$. Esta agrupación de switches se basa en el número de puertos que se componen de aplicando uno de los algoritmos disponibles: K-medias, Hierarchical, Esperanza-Maximización o Basado en Densidad. A continuación se proporciona una breve explicación sobre los diferentes algoritmos de agrupamiento:

- **K-medias**: es un método de agrupamiento que tiene como objetivo dividir n observaciones en c grupos basados en la media más cercana de cada observación (vector de media única).

- **Hierarchical**: es un método de análisis de agrupamiento que crea una jerarquía de grupos basada en la distancia de las conexiones de los objetos.

- **Esperanza-Maximización (EM)**: es un método iterativo para buscar la máxima probabilidad o el máximo de un conjunto. En otras palabras, estima las medias y las desviaciones estándar para cada grupo para maximizar la probabilidad.
- **Basado en Densidad**: es un algoritmo de agrupamiento que define áreas con mayor densidad de objetos que el resto del conjunto de datos.

Una vez que se ha seleccionado el algoritmo, los criterios para construir los grupos son los siguientes:

- Los switches que contienen dos puertos (entrante y saliente) se agruparán en un grupo en el que se aplicará el algoritmo de conservación de flujo (propuesto en la Sección 4.3).
- En otro caso, los grupos que contienen switches con tres o más puertos se agruparán por su criticidad. Este valor crítico se asigna a cada switch dependiendo de cuántas veces aparezca dicho switch en todas las rutas disponibles para enviar los datos del host del cliente al host del servidor.

El proceso de agrupamiento descrito anteriormente está implementado en la función $CREAR\_GRUPO(G, Algorithm, N)$ en el Algoritmo 5. Concretamente, la función $Load\_Instances()$ recorre todos los switches y les asigna valores de cuántos puertos tienen y, de la misma manera, la función $Calculate\_Criticality()$ les asigna valores de su criticidad. A continuación, el paso de agrupación en grupos se realiza a través de la función $Run\_Clusters(S, Algorihtm, N)$ el cual introduce uno de los algoritmos de agrupamiento implementados (K-medias, Hierarchical, Esperanza-Maximización, Basado en Densidad) y el número de grupos nc como parámetros.

Algoritmo 5: Función de Crear Grupo

Input: Grafo Red $G(S, L)$
Algoritmo Alg
Número de Grupos N
Result: Conjunto de Grupos C

```
1 procedure Crear_Grupo
2     foreach switch s_i ∈ S do
3         s_i = Load_Instances() ;
4         s_i = Calculate_Criticality();
5         C = Run_Clusters(S, Alg, N);
6     return C;
7 end procedure
```

Una vez que se ha realizado el proceso de agrupamiento, nuestra herramienta de monitorización comienza a recopilar información de la red en cada período de monitorización t_{mon}. Para reducir las consultas de estadísticas, aplicamos diferentes estrategias en cada grupo de la red:

- El grupo que contiene switches con dos puertos ($deg(s_i) == 2 \ \forall s_i \in c_j$) aplica la mejora basada en el algoritmo de conservación de flujo propuesto en la Sección 4.3). De esta forma, los Algoritmos 6, 7 y 8 se aplicarán al grupo$c_i = (S_i, L_i)$ que se establecerá como el grafo de entrada del Algoritmo 6. Por lo tanto, estos switches se liberan de las peticiones ya que sus datos se obtendrán a través de sus switches vecinos.

- Para los grupos que contienen switches con tres o más puertos, los switches N (donde N es un porcentaje del número total de switches de cada grupo fijado por el usuario) se eligen aleatoriamente en cada t_{mon} período. De esta manera, todos los switches de cada grupo serán monitorizados pero no en el mismo período sino en diferentes.

- Otra estrategia que contiene switches con tres o más puertos se basa en elegir los switches más concurrentes (en términos de tráfico) en el mismo período de monitorización. Por lo tanto, los switches que no reenvían el tráfico no serán monitorizados y la cantidad de consultas estadísticas disminuirá.

El controlador utiliza los grupos para realizar las consultas como indica el Algoritmo 6. Para propuestas experimentales, se ha elegido la tasa de datos y la tasa de error como métricas para demostrar la viabilidad de esta propuesta.

La tasa de datos es una métrica de monitorización que puede prevenir la congestión de enlaces, fallos de éstos y ataques DDoS entre otros. Por lo tanto, el controlador inicia la supervisión de la topología de la red en todos los grupos obtenidos en el Algoritmo 6. El grupo que contiene switches de 2 grados no se supervisará, ya que sus valores se calcularán con los valores de tasa de datos de los switches vecinos mediante la función $Neighbour\_Switch()$. En otro caso, para los grupos c_j que contienen switches cuyo grado es mayor o igual que tres, el controlador elige al azar N (valor fijo) switches $s_i \in c_j$ usando un método pasivo $ofpt\_port\_stats(nodo, puerto(j))$ proporcionado por Floodlight [Flo17] para enviar la solicitud ($OFPT\_STATS\_REQUEST$) a cada switch s_i a través de un canal seguro. Una vez que el switch ha recibido dicho mensaje, responde al controlador con otro mensaje llamado ($OFPT\_STATS\_REPLY$) que indica su estado. El controlador recibe dicha respuesta e identifica el contador de dicho switch con su puerto en la topología. La tasa de datos enviados $d_{s_{ij}}$ se calcula como la diferencia entre los bytes enviados ($d_{s_{ij}^k}$ - $d_{s_{ij}^{k-1}}$) en un período de monitorización t_{mon}. Para obtener métricas originales de tasa de datos $d_{s_{ij}}$, el controlador contrasta si los enlaces de los switches se están monitorizando en dicho período.

Algoritmo 6: Función de Cálculo de la Tasa de Datos

Input: Conjunto de Grupos C
Result: Tasa de Datos $d_{s_{ij}}$ enviada $\forall arc(i,j) \in L$

```
1  procedure Calcular_Tasa_Datos
2      //Comienza tiempo monitorización t con periodo t_mon;
3      foreach periodo k = 0, 1, 2 ∈ t do
4          foreach grupo c_i ∈ C do
5              if ports_number(c_i) == 2 then
6                  foreach arc(i,j) ∈ c_i do
7                      s_m = Neighbour_Switch(G, arc(i,j));
8                      d_{s_ij} = d_{s_im};
9              else
10                 s_z = choose_random(c_i);
11                 is_monitored(s_z) = true;
12                 s_z^k = tx_bytes in ofp_port_stats(node, port(i))
13                 if k > 0 then
14                     d_{s_zj} = (s_z^k − s_z^{k−1}) / t_mon;
15                 if is_monitored(s_m) == true for each s_m ∈ c_i then
16                     foreach switchs_m ∈ c_i do
17                         is_monitored(s_m) = false;
18 end procedure
```

Por lo tanto, aplicando estas técnicas mejoradas, estamos disminuyendo el número de solicitudes a los switches con respecto a una técnica no mejorada, como las herramientas de monitorización que supervisan todos los switches de la topología.

Continuando con las métricas de la red, la tasa de error es la medida que calcula el porcentaje de pérdida de paquetes en un enlace. El Algoritmo 7 es el procedimiento responsable de calcular la tasa de error en un enlace $arc(i,j)$ dentro de la topología.

Algoritmo 7: Función de Cálculo de la Tasa de Paquetes Perdidos

Input: Enlace Red $arc(i,j)$
Periodo Monitorización t_{mon}
Periodo de Tiempo k
Result: Tasa Paquetes Perdidos lr_{ij} for $arc(i,j)$

```
1  procedure Tasa_Paquetes_Perdidos_Switch
2      s_i^k = tx_bytes in ofpt_port_stats(src_node, src_port(i))
3      r_j^k = rx_bytes in ofpt_port_stats(dst_node, dst_port(i))
4      if k > 0 then
5          lr_{s_ij} = ((s_i^k − s_j^{k−1}) − (r_i^k − r_j^{k−1})) / t_mon;
6  end procedure
```

El controlador utiliza un método pasivo $ofpt\_port\_stats(node, port(i))$ que envía peticiones a un puerto ($OFPT\_STATS\_REQUEST$) de los switches $s_i \in c_j$ a través de un canal seguro. Una vez que el switch recibe este mensaje, responde con su estado al controlador con el mensaje ($OFPT\_STATS\_REPLY$). El controlador recibe la respuesta e identifica el contador con el switch y el puerto correspondiente en la topología. Entonces, el valor de la tasa de error en un enlace $arc(i,j)$ es definido como $lr_{s_{ij}}$ o $lr'_{s_{ij}}$ y se calcula como la diferencia de los bytes enviados $d_{s_{ij}}$ o $d'_{s_{ij}}$ en el switch origen y el puerto origen s_{ij} o s'_{ij} menos los bytes entrantes en el puerto destino de el switch destino $r_{s_{ij}}$ o $r_{s'_{ij}}$ en un período de monitorización t_{mon}.

Por último, el Algoritmo 8 calcula la tasa de error para cada enlace de la topología. En este sentido, se recorren los grupos del conjunto C y se hace distinción del cálculo de la tasa de error según el número de puertos de los switches de dicho grupo: si son iguales a dos (puerto entrante y saliente) se toma el valor directo obtenido de la tasa de paquetes perdidos descrita en el Algoritmo 7. Sino, se utiliza el Algoritmo 7 con los valores de los switches adyacentes.

Algoritmo 8: Función de Cálculo de la Tasa de Error

Input: Conjunto de Grupo C
Result: Tasa Error $lr_{s_{ij}}$ $\forall arc(i,j) \in L$

```
1  procedure CALCULA_TASA_ERROR
2      //Comienza tiempo monitorización t con periodo t_mon;
3      foreach periodo k = 0,1,2 ∈ t do
4          foreach grupo c_i ∈ C do
5              if ports_number(c_i) == 2 then
6                  foreach arc(i,j) ∈ c_i do
7                      lr_m = packetLossRateSwitch(Neighbour_Switch(G, arc(i,j)));
8                      lr_{s_ij} = lr_{s_im};
9              else
10                 s_z = choose_random(c_i);
11                 is_monitored(s_z) = true;
12                 lr_{s_ij} = packetLossRateSwitch(G, arc(i,j));
13                 if is_monitored(s_m) == true for each s_m ∈ c_i then
14                     foreach switchs_m ∈ c_i do
15                         is_monitored(s_m) = false;
16 end procedure
```

4.4.4. Simulaciones y Resultados

La viabilidad de la técnica mejorada de agrupamiento respecto al algoritmo de conservación de flujo ha sido evaluada realizando simulaciones con una topología compuesta por 7 switches OpenFlow ($s_1, s_2, s_3, s_4, s_5, s_6$ y s_7) y 2 host (h_1 y h_2) conectados

a s_1 y s_7 respectivamente. Los enlaces [switch: s_3 - puerto: p_4, switch: s_6 - puerto: p_1] y [switch: s_6 - puerto: p_2, switch: s_5 - puerto: p_3] están configurados con valores de velocidad de datos máxima (1 Mbps de ancho de banda) y porcentaje de pérdida (5 % de los paquetes se perderán). Esta topología se describe en la Figura 4.13.

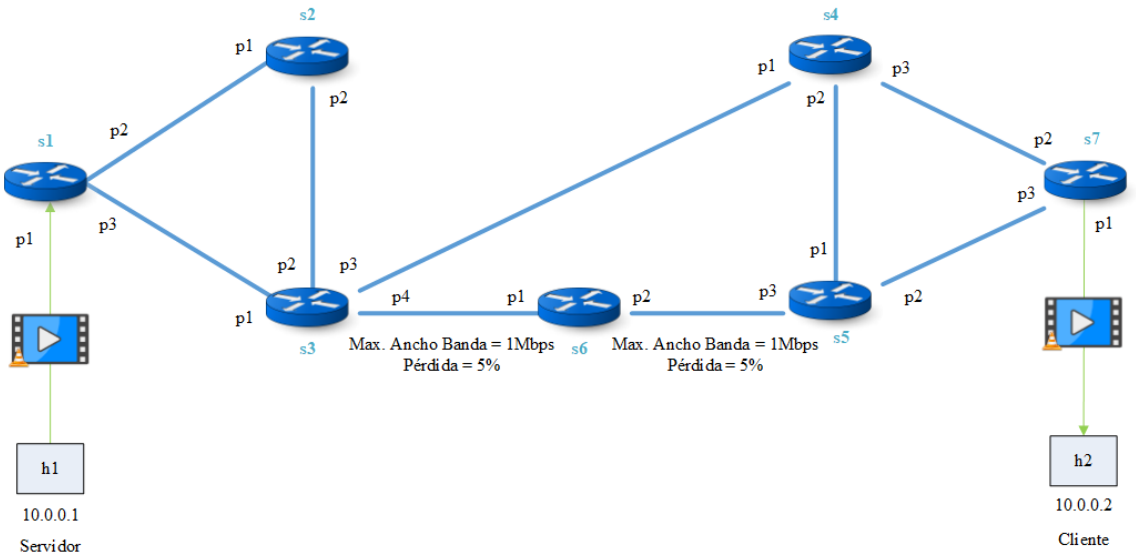


Figura 4.13: Topología de Prueba con Optimización basada en Agrupamiento

Todas las simulaciones se ejecutaron en un simulador de red llamado Mininet v2.1.0 [min17] utilizando scripts de Python, que contiene la topología para examinar descrita anteriormente. Mininet puede crear con pocos pasos topologías personalizadas para simular en un ordenador. El conjunto de prueba se ejecutó en una máquina virtual dentro de un servidor modelo TD350 de ThinkServer que está compuesto por un Intel Xeon Series E5-2600 v3 con 12 núcleos, 64 GB de RAM DDR4 y Ubuntu 14.04 como sistema operativo. Dentro del servidor, se encuentra la máquina virtual que se compone de un sistema operativo invitado Linux (Ubuntu v. 13.04) (64 bits, 8 GB de RAM, 2 CPU, 8 GB de HD). Los módulos de monitorización de agrupamiento y conservación de flujo se implementan utilizando el controlador Floodlight [Flo17] basado en Java dentro de dicha máquina virtual.

Una simulación consiste en una transmisión de vídeo enviada desde el host 1 al host 2 usando el servidor de vídeo VLC y RTP/UDP como protocolo de transmisión al mismo tiempo que se ejecuta el módulo de monitorización. Por lo tanto, la tarea principal del módulo de monitorización es medir las estadísticas de los switches de red (tanto los enlaces donde se envía el vídeo como el resto de ellos). Utilizamos un conjunto de vídeos para la transmisión de vídeo donde sus características se recopilan en la Tabla 4.5 mientras que el tiempo de monitorización (t_{mon}) es de 200 ms. Los vídeos utilizados en la transmisión de vídeo son “Highway_cif” [vid17d], “Akiyo_cif” [vid17a], “Bridge-far_cif” [vid17b] y

"Clarie_cif" [vid17c]. La conexión entre el host cliente y el host servidor es una ruta fija compuesta de [switch: s_1 - puerto: p_1, switch: s_1 - puerto: p_3, switch: s_3 - puerto: p_1, switch: s_3 - puerto: p_4, switch: s_6 - puerto: p_1, switch: s_6 - puerto: p_2, switch: s_5 - puerto: p_3, switch: s_5 - puerto: p_2,, switch: s_7 - puerto: p_3, , switch: s_7 - puerto: p_1].

Tabla 4.5: Características de los Vídeos usados en las Simulaciones

vídeo	Formato	Frames	Duración (segundos)	Tamaño	Tasa de Datos (kbits/s)
Highway_cif	MPEG 12	2000	80	2.5 MB	257.4
Akiyo_cif	MPEG 12	300	11.96	481.6 KB	329.8
Bridge-far_cif	MPEG 12	2101	84	2.95 MB	252.1
Clarie_cif	MPEG 12	494	19.72	712 KB	295.8

El conjunto de pruebas se divide en dos casos de estudio:

1. **Caso de Estudio A**: se compone de una prueba de concepto donde las diferencias entre los algoritmos mejorados de agrupamiento y conservación de flujo se manifiestan en la transmisión de vídeo de "Highway_cif" [vid17d]. Luego, este caso de estudio se repite dos veces en las mismas condiciones cambiando el método de monitorización. En el primero se usa la mejora de agrupamiento propuesta en este documento y en el otro, se aplica el método propuesto en la Sección 4.3 que reduce el número de consultas de switches de grado 2 de la topología. Además, para cada método de monitorización, la prueba se repite dos veces nuevamente para separar los valores de cada métrica (tasa de datos y tasa de error). Finalmente, los resultados se comparan con estos dos módulos mejorados para determinar qué método es eficiente en términos de carga y precisión de los datos monitorizados. Los resultados de estas pruebas se dividen en tres métricas: el número de solicitudes para obtener el estado de cada switch, la tasa de datos y la tasa de error de los switches que componen la ruta de transmisión.

2. **Caso de Estudio B**: se centra en la aplicabilidad del algoritmo de agrupamiento en otros vídeos ("Akiyo_cif" [vid17a], "Bridge-far_cif " [vid17b] y "Clarie_cif" [vid17c]). El objetivo de este caso de estudio es verificar la viabilidad de nuestra mejora con diferentes tipos de vídeo en términos de frames, duración, tasa de bits y también diferentes patrones de tasa de datos. Siguiendo el mismo procedimiento que el Caso de estudio A, la prueba se repite dos veces para cada vídeo obteniendo su tasa de datos.

Los parámetros para ambos casos de estudio son: enlaces [switch: s_3 - puerto: p_4, switch: s_6 - puerto: p_1] (en adelante denominado Enlace A) y [switch: s_6 - puerto: p_2, switch: s_5 - puerto: p_3] (en lo sucesivo denominado Enlace B) se establecen con un ancho de banda de 1 Mbps y un porcentaje de pérdida del 5 % respecto a los paquetes que se envían a través de estos enlaces.

4.4.4.1. Resultados obtenidos en el Caso de Estudio A

Con respecto a las simulaciones de tasa de datos que se han realizado, el número de peticiones de monitorización en el método no mejorado durante la simulación (80 s) es de 2079 peticiones, mientras que en el algoritmo mejorado es de 903. La diferencia de 1176 peticiones muestra una reducción del 57 % de peticiones con respecto a una simulación de monitorización no mejorada como se muestra en la Tabla 4.6.

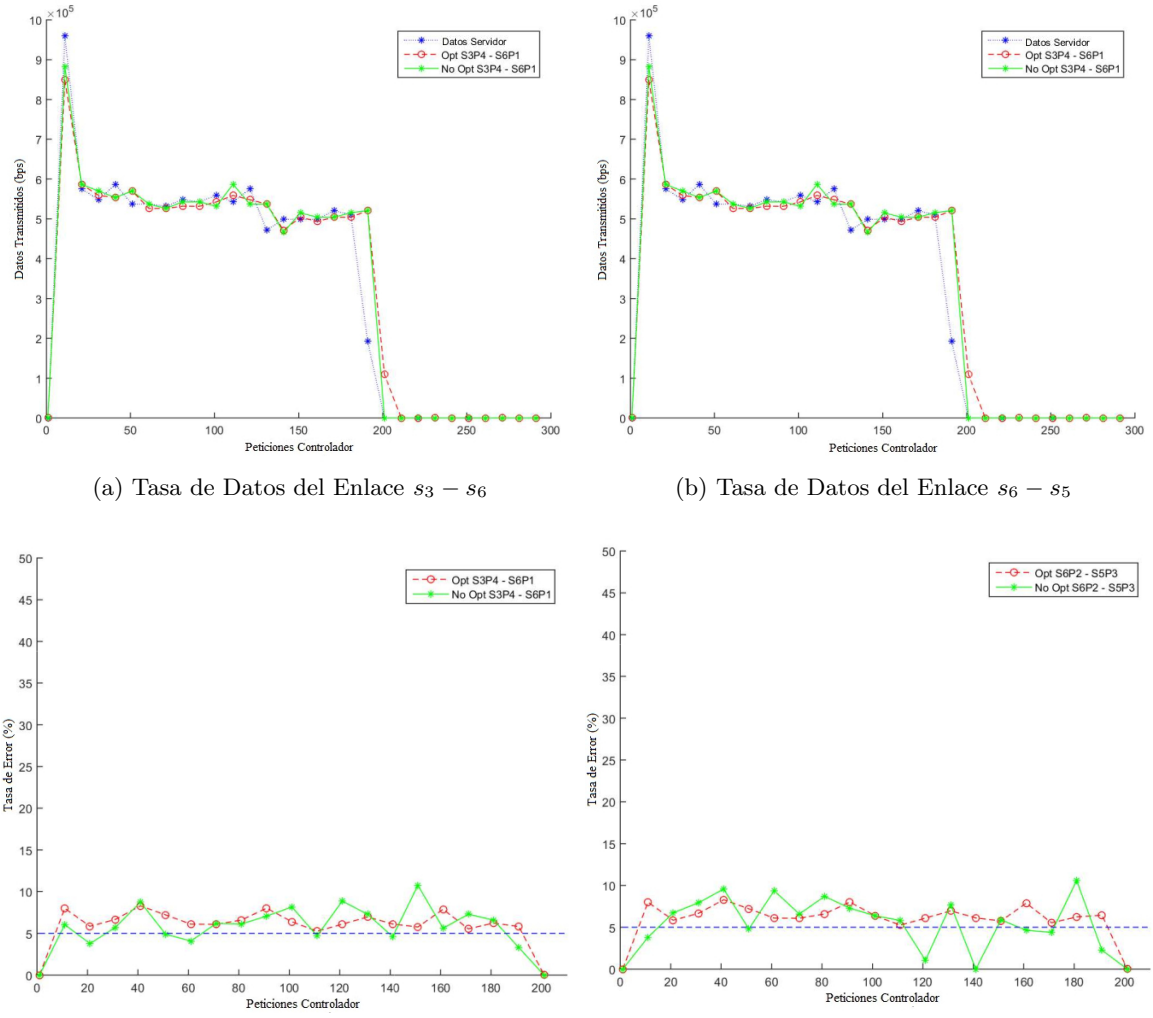
Tabla 4.6: Resultados usando Método Mejorado respecto a No Mejorado para Tasa de Datos dentro del Caso de Estudio A

Método	Número de Peticiones	Reducción	Reducción (%)	Enlace	Diferencia
Mejorado	903	1176	57 %	Link s_3-s_6	45.49 Kbps
No Mejorado	2079			Link s_6-s_5	54.33 Kbps

Por su parte, la Figura 4.14a y la Figura 4.14b muestran el flujo de tráfico entre dos simulaciones en las que una aplica la mejora respecto a otra que no la utiliza. La Figura 4.14a describe el flujo de datos (en bps) a través de los enlaces s_3-s_6 (switch: s_3 - puerto: p_4, switch: s_6 - puerto: p_1) y la Figura 4.14b los enlaces entre s_6-s_5 (switch: s_6 - puerto: p_2, switch: s_5 - puerto: p_3). Las líneas puntiagudas (líneas azules) muestran la tasa de datos que el servidor está enviando el vídeo, mientras que las líneas continuas (líneas verdes) muestran la tasa de datos obtenida utilizando el método no mejorado y las líneas de puntos (líneas rojas) muestran la tasa de datos con Algoritmo mejorado. Como se esperaba, ambos enlaces experimentan un aumento en la tasa de datos debido a la transmisión del vídeo entre h_1 y h_2. Tan pronto como finaliza la transmisión (alrededor de 200 peticiones), la tasa de datos disminuye, lo que demuestra la eficiencia del algoritmo para detectar cambios en la transmisión de la red. De manera similar, la diferencia promedio entre los enlaces s_3-s_6 con/sin la mejora es de 45,49 Kbps y 54,33 Kbps en el enlace s_6-s_5 como describe la Tabla 4.6. Estos resultados confirman que la mejora mantiene buenos niveles de precisión y reduce el número de peticiones en el plano de datos.

De la misma manera que la métrica de tasa de datos, las Figuras 4.14c y 4.14d muestran la tasa de error en los datos de tráfico sobre la ruta de transmisión. La Figura 4.14c describe la tasa de error en el flujo de datos (en bps) a través de los enlaces s_3-s_6 (switch: s_3 - puerto: p_4, switch: s_6 - puerto: p_1) y la Figura 4.14d entre el enlace s_6-s_5 (switch: s_6 - puerto: p_2, switch: s_5 - puerto: p_3). Las líneas continuas (líneas verdes) muestran la tasa de error obtenida utilizando el método no mejorado, las líneas punteadas (líneas rojas) muestran la tasa de error con el algoritmo mejorado y la línea continua punteada (líneas azules) muestra el porcentaje netem configurado en Mininet (5 % de pérdida de paquetes). Como era de esperar, ambos enlaces detectan la pérdida de información entre los enlaces debido a las características de estos enlaces.

Por lo tanto, según el diseño de nuestro algoritmo, podemos estimar aproximadamente que se logra una reducción de mensajes de peticiones del 57 % en una topología similar al experimento. Esta reducción pertenece a la proporción entre los mensajes de estadísticas contabilizadas en los enlaces $s_3 - s_6$ y $s_6 - s_5$ de las simulaciones mejoradas y no mejoradas.



(a) Tasa de Datos del Enlace $s_3 - s_6$

(b) Tasa de Datos del Enlace $s_6 - s_5$

(c) Tasa de Error del Enlace $s_3 - s_6$

(d) Tasa de Error del Enlace $s_6 - s_5$

Figura 4.14: Análisis de la tasa de datos y de error para el envío del vídeo Highway_cif

4.4.4.2. Resultados obtenidos en el Caso de Estudio B

El número medio de consultas de monitorización en el método no mejorado durante la simulación es de 2089 peticiones, mientras que en el algoritmo mejorado el valor medio es 901. Por lo tanto, la diferencia media es de alrededor de 1186 consultas muestra una relación de reducción del 57 % de las peticiones con respecto a un simulación de monitorización no mejorada como se indica en las Tablas 4.7, 4.8 y 4.9.

Tabla 4.7: Resultados usando Método Mejorado respecto a No Mejorado para Tasa de Datos en el vídeo Akiyo dentro del Caso de Estudio B

Método	Número de Peticiones	Reducción	Reducción (%)	Enlace	Diferencia
Mejorado	900	1186	57 %	Link s_3-s_6	3.42 Kbps
No Mejorado	2089			Link s_6-s_5	4.54 Kbps

Tabla 4.8: Resultados usando Método Mejorado respecto a No Mejorado para Tasa de Datos en el vídeo Bridge-far dentro del Caso de Estudio B

Método	Número de Peticiones	Reducción	Reducción (%)	Enlace	Diferencia
Mejorado	903	1183	57 %	Link s_3-s_6	29.86 Kbps
No Mejorado	2089			Link s_6-s_5	39.8 Kbps

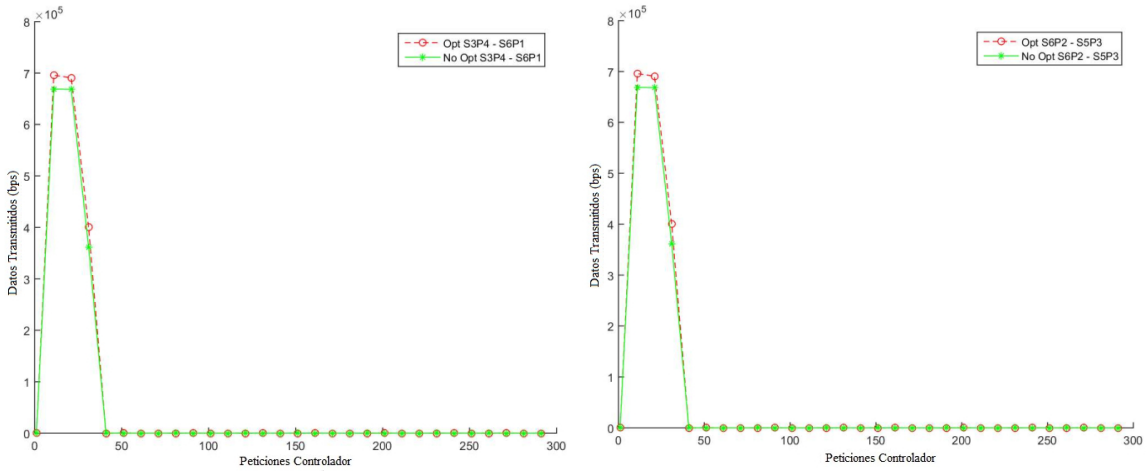
Tabla 4.9: Resultados usando Método Mejorado respecto a No Mejorado para Tasa de Datos en el vídeo Claire dentro del Caso de Estudio B

Método	Número de Peticiones	Reducción	Reducción (%)	Enlace	Diferencia
Mejorado	903	1190	57 %	Link s_3-s_6	5.21 Kbps
No Mejorado	2093			Link s_6-s_5	8.72 Kbps

De la misma manera que el Caso de Estudio A, la Figura 4.15a y la Figura 4.15b muestran el flujo de tráfico en el vídeo llamado "Akiyo_cif" [vid17a], Figura 4.16a y Figura 4.16b muestra el flujo de tráfico en el vídeo llamado "Bridge-far" [vid17b] y finalmente, la Figura 4.17a y la Figura 4.17b muestran el flujo de tráfico en el vídeo llamado "Claire" [vid17c] entre dos simulaciones en las que una aplica la mejora respecto a otra que no la usa. Las Figuras 4.15a, 4.16a y 4.17a describen el flujo de datos (en bps) a través de los enlaces s_3-s_4 (switch: s_3 - puerto: p_2, switch: s_4 - puerto: p_1) y Figuras 4.15b, 4.16b y 4.17b los enlaces entre s_4-s_9 (switch: s_4 - puerto: p_2, switch: s_9 - puerto: p_2). Las líneas continuas muestran la velocidad de datos obtenida utilizando el método no mejorado y las líneas de puntos muestran la velocidad de datos con el algoritmo mejorado. Como era de esperar, nuestro método mejorado es capaz de monitorizar todo tipo de vídeo (sin importar el patrón de tráfico) obteniendo casi la misma tasa con diferencias muy pequeñas y reduciendo a 57 % el número de consultas de monitorización como se indica en las Tablas 4.7, 4.8 y 4.9.

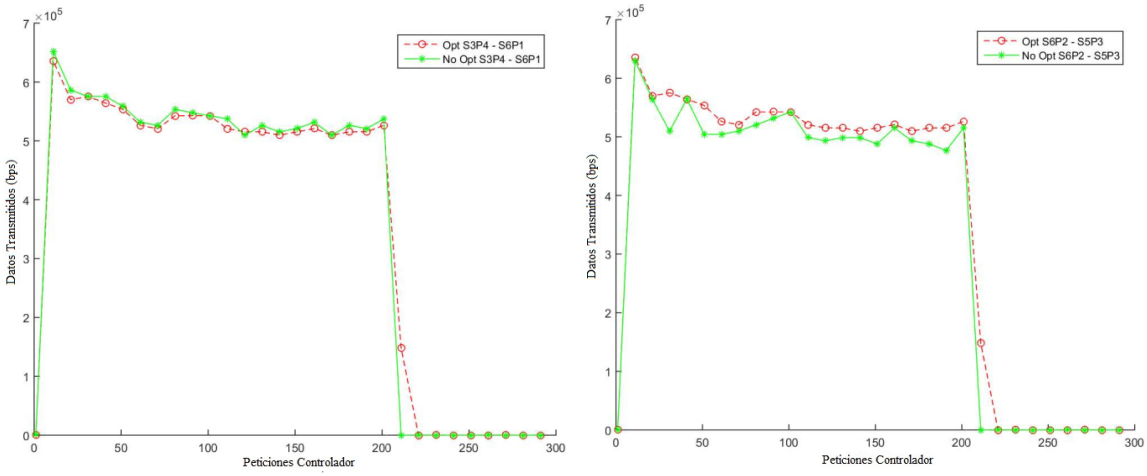
Basado en el diseño de nuestro algoritmo con los vídeos usados en las simulaciones del Caso de Estudio B, podemos estimar aproximadamente que se logra una reducción del mensaje del 57 % en una topología similar al experimento. Esta reducción pertenece a la proporción entre los mensajes de estadísticas contabilizadas en los enlaces $s_3 - s_6$ y $s_6 - s_5$ de las simulaciones mejoradas y no mejoradas.

Para finalizar con la mejora de monitorización de nuestro trabajo de investigación, ambos casos de estudio muestran una reducción en el flujo de tráfico de la red para mantener libre el controlador SDN para otras acciones de administración.



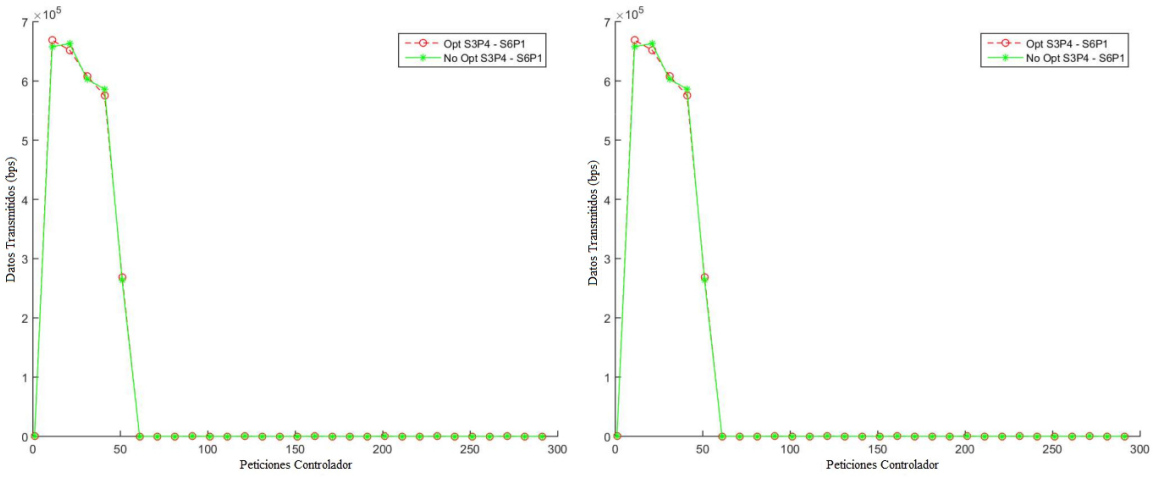
(a) Tasa de Datos del Enlace $s_3 - s_6$ (b) Tasa de Datos del Enlace $s_6 - s_5$

Figura 4.15: Análisis de la tasa de datos y de error para el envío del vídeo Akiyo_cif



(a) Tasa de Datos del Enlace $s_3 - s_6$ (b) Tasa de Datos del Enlace $s_6 - s_5$

Figura 4.16: Análisis de la tasa de datos y de error para el envío del vídeo Bridge-far_cif



(a) Tasa de Datos del Enlace $s_3 - s_6$ (b) Tasa de Datos del Enlace $s_6 - s_5$

Figura 4.17: Análisis de la tasa de datos y de error para el envío del vídeo Clarie_cif

4.5. Análisis de Algoritmos de Agrupamiento en Monitorización

Una vez que se ha demostrado que la arquitectura de monitorización basada en agrupamiento ha reducido el número de consultas estadísticas manteniendo valores muy aproximados a los del host cliente desde donde se envía el vídeo en las simulaciones usando el algoritmo de agrupamiento K-medias, la siguiente contribución amplía dicha arquitectura con dos nuevos algoritmos: Esperanza-Maximización (EM) y Basado en Densidad.

La QoE percibida por los usuarios finales en una transmisión de vídeo depende significativamente del tipo de aplicación y el nivel de QoS proporcionado por la red (por ejemplo, ancho de banda y retraso entre otros). Esto refleja la necesidad de tener un estado de la red completo con sus componentes. Debido a esta razón, las herramientas de monitorización deben diseñarse para recopilar dicha información con precisión. De esta manera, discutimos los beneficios de usar diferentes algoritmos de agrupamiento en una topología de la red para optimizar las consultas de estadísticas al monitorizar la red.

La presente contribución se basa en la aplicabilidad y el estudio de los beneficios al aplicar diferentes algoritmos de agrupamiento en la herramienta de monitorización presentada en la Sección 4.4 de la presente tesis. Este estudio se verifica utilizando las métricas de tasa de datos y tasa de error, que también se describen en la Sección 4.4, para afirmar que es posible disminuir el número de consultas de monitorización mientras se mantiene la precisión de los datos monitorizados.

De la misma manera que el resto de las contribuciones, ésta trata de optimizar el número de solicitudes de switches utilizando diferentes algoritmos de agrupamiento para

reducir el número de solicitudes de switches respetando una técnica no optimizada, como herramientas de monitorización que supervisan todos los switches de una topología. Es importante aclarar que los algoritmos de agrupación no se utilizan para optimizar el número de consultas, sino que se utilizan para crear grupos sobre los cuales se ejecutan las técnicas de optimización descritas en la Sección 4.4. Luego, aplicando diferentes algoritmos, obtendremos diferentes asignaciones de grupo y, por lo tanto, diferentes resultados en las simulaciones.

A continuación, se proporciona una breve explicación sobre los diferentes algoritmos de agrupamiento utilizados en esta contribución:

- **K-medias**: Es probablemente el algoritmo de agrupamiento más conocido y el más simple entre los algoritmos de aprendizaje no supervisados. En realidad, significa que los grupos para un conjunto de datos dado, están representados por una variable "k". Para cada grupo, se define un centroide. El centroide es un punto de datos presente en el centro de cada grupo (considerando la distancia euclidiana). La idea principal consiste en definir los centroides alejados entre sí para que la variación sea menor. Después de esto, cada punto de datos en el grupo se asigna al centroide más cercano. Todos los puntos de datos ahora están asignados. Los k centroides (centroides del grupo) se calculan nuevamente como baricentros de los grupos. Estos nuevos centroides deben asignarse a cada punto de datos en cada punto de datos en grupos como se mencionó anteriormente. Este proceso se repite hasta que los centroides ya no se mueven de sus posiciones. Esto proporciona la configuración para minimizar la medida utilizando una función objetivo (debe definirse anteriormente). El algoritmo de agrupamiento K-medias ha resultado ser muy útil para agrupar datos nuevos. Algunas aplicaciones prácticas que utilizan la agrupación de K-medias son las mediciones de sensores, monitorización de actividad, detección de audio y segmentación de imágenes [KMN+02, HW79].

- **Esperanza-Maximización (EM)**: Es un método iterativo para buscar la máxima probabilidad o el máximo de un conjunto de datos. Este algoritmo se basa en la distribución gaussiana en estadística. Considera una colección de distribuciones gaussianas para el conjunto de datos en un problema de aprendizaje automático. Entonces, esto significa que los datos se representan como un modelo para resolver el problema. En general, este algoritmo elige un componente de distribución gaussiano para un grupo de datos y asigna un valor de probabilidad. Después de esto, se calcula una muestra puntual en función de ese componente gaussiano [Moo96].

- **Basado en Densidad**: Es un enfoque no paramétrico donde los grupos se consideran áreas de alta densidad. Los métodos de agrupamiento Basado en Densidad no requieren número de grupos como parámetros de entrada, ni tampoco hacer suposiciones sobre la densidad subyacente p(x) o la varianza dentro de los grupo que

pueden existir en el conjunto de datos. Como consecuencia, los grupos construidos usando Basado en Densidad no son necesariamente grupos de puntos con una baja disparidad entre pares dentro del grupo, medida por un la función de disimilitud "dis". Por lo tanto, no necesariamente tienen una forma convexa, sino que pueden tener una forma arbitraria en el espacio de datos [KKSZ11].

4.5.1. Notación de la Red

Nuestro modelo abstracto de red se recopila en la Tabla 4.10. La topología de la red se representa como un gráfico dirigido $G = (S, L)$ donde $S = \{s_1, s_2, ..., s_{ns}\}$ denota el conjunto de switches y $L = \{l_1, l_2, .., l_{nl}\}$ representa el conjunto de enlaces entre switches. Además, el conjunto de grupos se denota como $C = \{c_0, c_1, ..., c_{nc-1}\}$ donde cada grupo es al mismo tiempo un conjunto de switches agrupados por un algoritmo de agrupamiento (Alg). Por ejemplo, el $arc(i, j)$ representa el enlace del switch origen i al switch destino j. Además, $ns = |S|$, $nl = |L|$ y $nc = |C|$ denotan el número de switches, enlaces y grupos en la topología respectivamente. Por último, definimos el grado de un switch (número de enlaces incidentes) como $deg(s_i)$ donde $s_i \in S$. Suponemos que ns, nl y nc son finitos y $deg(s_i) > 0$. Además, suponemos que los datos monitorizados se agrupan en flujos de paquetes F. Por lo tanto, todos los paquetes de datos que se envían en G pertenecen al menos a un flujo de paquetes $F_i \in F$. Cada F_i tiene una ruta desde su switch origen s_i a su switch destino s_j.

Tabla 4.10: Modelo de Notación de la Red

Símbolo	**Descripción**
$G = (S, L)$	Grafo de la Red
S	Conjunto de Switches en el Grafo de la Red G
L	Conjunto de Enlaces en el Grafo de la Red G
C	Conjunto de Grupos que contienen switches
$s_1, s_2, ..., s_n$	Switches Genéricos en el Grafo de la Red G
$l_1, l_2, ..., l_n$	Enlaces Genéricos en el Grafo de la Red G
$c_0, c_1, ..., c_{nc-1}$	Grupos Genéricos en el Grafo de la Red G
Alg	Tipo de Algoritmo de Agrupamiento aplicado
$ns = \|\|S\|\|$	Número de Switches en el Grafo de la Red G
$nl = \|\|L\|\|$	Número de Enlaces en el Grafo de la Red G
$nc = \|\|C\|\|$	Número de Grupos en el Grafo de la Red G
$arc(i, j) \in L$	Enlace desde switch fuente s_i al switch destino s_j
$deg(s_i)$	Número de enlaces en un switch s_i

4.5.2. Aplicación de los Algoritmos de Agrupamiento

Antes de comenzar a solicitar estadísticas para cambiar, $Clustering\_Function(G)$ lee la topología de red estructurada en el grafo $G = (S, L)$ que será usado para crear los grupos.

Dado que G contiene el gráfico de red, la función $Create\_Cluster(G, Algorithm, nc)$ lo divide en N grupos, definidos como $c_0, c_1, .., c_{nc-1}$.

La agrupación de switches se basa en el número de puertos que están compuestos y la importancia de aplicar uno de los algoritmos implementados: K-medias, Esperanza-Maximización o Basado en Densidad descritos en la sección anterior.

Una vez que se ha seleccionado el algoritmo, los criterios para monitorizar los grupos dependen del número de puertos. Por un lado, la herramienta de monitorización aplica el principio del algoritmo de conservación de flujo en switches de 2 grados, ya que todo el tráfico de datos que recibe el puerto entrante del switch se reenviará al puerto saliente del mismo switch y, por lo tanto, el tráfico a través de dichos enlaces será casi lo mismo. Por otro lado, los grupos que contienen switches con tres o más puertos se eligen aleatoriamente en cada período t_{mon} que se supervisará. Para obtener más información, estos algoritmos se describen en las Secciones 4.4.2 y 4.4.3 de la presente tesis.

Para propósitos experimentales, elegimos la tasa de datos y la tasa de error como métricas para demostrar la viabilidad de esta contribución. El procedimiento para calcular ambas métricas se describe en la Sección 4.4.3 de la presente tesis. Concretamente, los procedimientos para calcular la tasa de datos y las métricas de error se explican en detalle en el Algoritmo 6 y el Algoritmo 8, respectivamente.

En esta sección, probamos los beneficios de los algoritmos de agrupamiento utilizando la herramienta de monitorización de optimización de agrupamiento descrita en la Sección 4.4 de la presente tesis.

Dividimos las pruebas en dos escenarios (Caso de Estudio A y Caso de Estudio B) utilizando diferentes topologías en términos de tamaño, número de puertos y múltiples rutas para realizar la transmisión de vídeo desde el servidor al servidor del cliente. Este enfoque se ha seguido para verificar la escalabilidad de la herramienta de monitorización propuesta en la Sección 4.4 y las ventajas/desventajas de aplicar un cierto algoritmo de agrupamiento o no.

Realizamos cuatro conjuntos de simulaciones atendiendo al uso de la optimización de agrupamiento o no para cada escenario de prueba:

- La primera simulación se realiza en el Caso de Estudio A para verificar el número de consultas de monitorización a los switches en el peor de los casos. En otras palabras, monitorizar todos los switches de la topología en cada período de monitorización t_{mon}.
- Después, se continua en el Caso de Estudio A realizando simulaciones que aplican los algoritmos de agrupamiento restantes descritos en la Sección 4.5 de la presente tesis ya que las simulaciones de K-medias ya se habían realizado en la Sección 4.4. La aplicación de los algoritmos de agrupación (K-medias, Esperanza-Maximización y Basado en Densidad) impactará en la construcción de grupos atendiendo a las características de cada uno. Una vez que se han creado los grupos, la herramienta

de monitorización aplica las estrategias de monitorización descritas en la Sección 4.4 de la presente tesis.

- La tercera simulación se realiza en el Caso de Estudio B para verificar la cantidad de consultas de monitorización a los switches en el peor de los casos. Este valor se utilizará para comparar la mejora del algoritmo de agrupamiento a posteriori.

- Finalmente, el último conjunto de simulaciones se realiza en el Caso de Estudio B aplicando los algoritmos de agrupamiento descritos en la Sección 4.5 de la presente tesis doctoral.

Una simulación consiste en una transmisión de vídeo enviada desde el host 1 al host 2 usando el servidor de vídeo VLC y RTP/UDP como protocolo de transmisión al mismo tiempo que se ejecuta el módulo de monitorización. Por lo tanto, la tarea principal del módulo de monitorización es medir las estadísticas de los switches de la red (tanto los enlaces donde se envía el vídeo como el resto de ellos). El vídeo utilizado en la transmisión de vídeo es "Highway_cif" [vid17d] y el tiempo de monitorización (t_{mon}) es de 200ms.

Todas las simulaciones se han ejecutado en una plataforma de simulador llamada Mininet v2.1.0 [min17] utilizando scripts de Python, que contiene las topologías para examinar que se describen a continuación. Mininet proporciona un entorno reproducible y controlado para la evaluación con facilidad de configuración y alteración. De esta manera, puede crear con pocos pasos topologías personalizadas para simular en una computadora. El conjunto de prueba se ejecutó en una máquina virtual dentro de un modelo TD350 de ThinkServer que está compuesto por un Intel Xeon Series E5-2600 v3 con 12 núcleos, 64 GB de RAM DDR4 y Ubuntu 14.04 como sistema operativo. Dentro del servidor, se encuentra la VM que se compone de un sistema operativo invitado Linux (Ubuntu v. 13.04) (64 bits, 8 GB de RAM, 2 CPU, 8 GB de HD).

A continuación, se proporciona la información de cada caso de estudio y los resultados de cada simulación utilizando las métricas de la tasa de datos y la tasa de error en dos enlaces de la topología que se han establecido con diferentes parámetros en términos de ancho de banda y pérdida de paquetes.

4.5.3. Evaluación del Caso de Estudio A

El primer estudio es una extensión de la prueba de concepto utilizada en la Sección 4.4.4 de la presente tesis doctoral donde las diferencias entre los algoritmos de optimización basado en agrupamiento (usando el algoritmo K-medias) y de conservación del flujo se manifestaron utilizando la tasa de datos como métrica única en las simulaciones. Por tanto, este caso de estudio se ha repetido y extendido utilizando los algoritmos de agrupamiento restantes que no se usaron (Esperanza-Maximización y Basado en Densidad) añadiendo la tasa de error como nueva métrica.

4.5.3.1. Definición del Escenario

Las simulaciones del Caso de Estudio A han sido realizadas utilizando la topología descrita en la Figura 4.18 compuesta por 7 OF-Switches ($s_1, s_2, s_3, s_4, s_5, s_6, s_7$) y 2 host ($h_1$ y h_2) conectados a s_1 y s_7 respectivamente. Los enlaces [switch: $s_3 – puerto : p_4, switch : s_6 – puerto : p_1$] y [switch: $s_6 – puerto : p_2, switch : s_5 – puerto : p_3$] están configurados con unos valores de tasa de datos máxima (1Mbps de ancho de banda) y porcentaje de perdida (5 % de los paquetes serán perdidos).

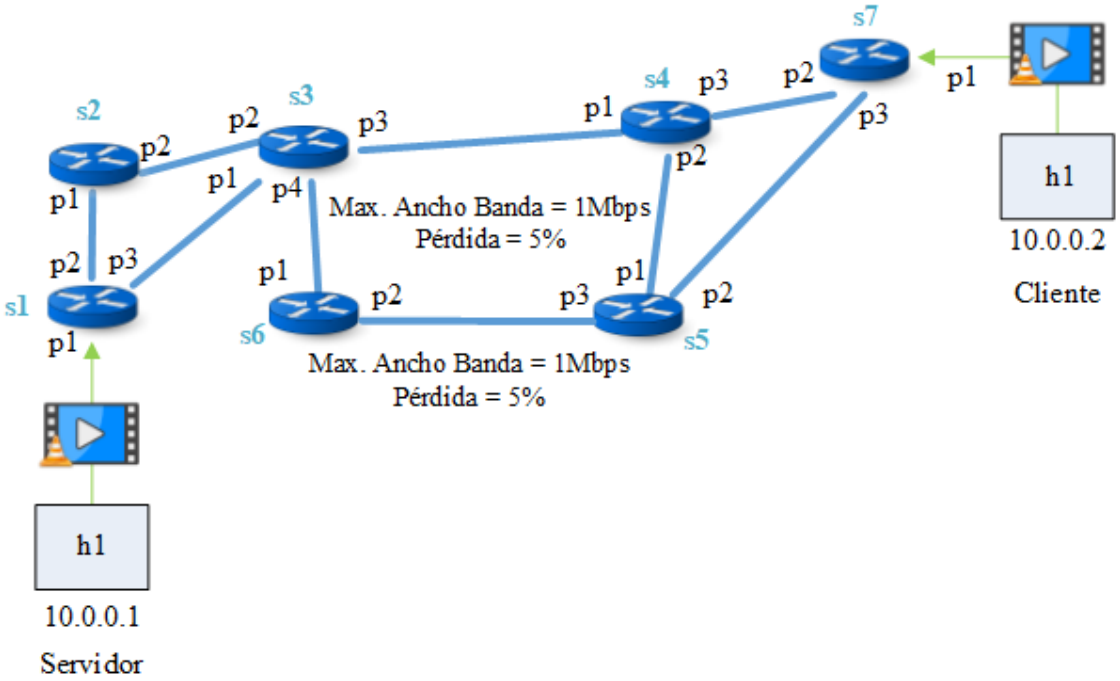


Figura 4.18: Topología de Prueba del Caso de Estudio A

La conexión entre el host cliente y host servidor es una ruta fija compuesta por [$switch : s_1 – puerto : p_1, switch : s_1 – puerto : p_3, switch : s_3 – puerto : p_1, switch : s_3 – puerto : p_4, switch : s_6 – puerto : p_1, switch : s_6 – puerto : p_2, switch : s_5 – puerto : p_3, switch : s_5 – puerto : p_2, switch : s_7 – puerto : p_3, switch : s_7 – puerto : p_1$].

Finalmente, los resultados se comparan usando los módulos optimizados y no optimizados de la Sección 4.4 de la presente tesis aplicando los tres algoritmos de agrupamiento descritos anteriormente para concluir los beneficios de cada algoritmo. Los resultados de estas pruebas se dividen en tres métricas: el número de solicitudes para obtener el estado de cada switch, la tasa de datos y la tasa de error de los switches que componen la ruta de transmisión de vídeo.

4.5.3.2. Simulaciones y Resultados

Antes de comenzar a monitorizar los switches de red, cada algoritmo crea sus grupos en función de sus propias características. La Tabla 4.11 muestra la asignación de cada switch en un determinado grupo utilizando los algoritmos de agrupamiento. Como se muestra en la tabla, la asignación de los switches no es la misma ya que cada algoritmo de agrupamiento tiene sus propias características. Por lo tanto, el número de consultas de monitorización será diferente debido a la asignación. Luego, una vez que se construyen los grupos, la herramienta de monitorización comienza a monitorizar la topología durante la transmisión de vídeo reuniendo las estadísticas de los switches de red que se utilizarán para calcular las métricas de tasa de datos y de tasa de error.

Tabla 4.11: Asignación de Grupos en el Caso de Estudio A

Grupo	K-medias	Esperanza-Maximización	Basado en Densidad
Grupo 0 (c_0)	s_3	$s_4 s_5 s_6$	s_3
Grupo 1 (c_1)	$s_1 s_6$	s_1	$s_1 s_6$
Grupo 2 (c_2)	$s_4 s_5$	$s_2 s_7$	$s_4 s_5$
Grupo 3 (c_3)	$s_2 s_7$	s_3	$s_2 s_7$

Como se indicó anteriormente, los resultados de las simulaciones que aplican el algoritmo K-medias se exponen en la Sección 4.4.4 de la presente tesis doctoral. El número de consultas de monitorización en el método no optimizado durante la simulación (80 segundos) fueron 2079 solicitudes, mientras que en el algoritmo optimizado, la aplicación del algoritmo de agrupamiento K-medias fue 903. Esto supuso una relación de reducción del 57 % de solicitudes con respecto a una Simulación de monitorización no-optimizada.

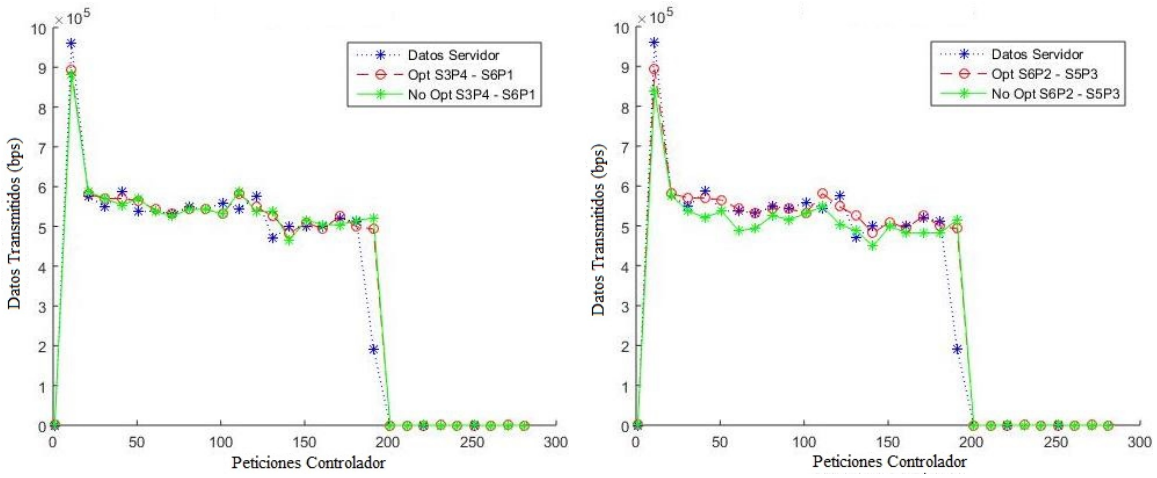
Comenzando con los resultados de la simulación realizada en este trabajo y tomando la métrica de tasa de datos como la medición que aplica el algoritmo de Esperanza-Maximización, el número de consultas de monitorización es 888. La diferencia de 1191 consultas (con el algoritmo no optimizado que es la solicitud 2079) muestra una relación de reducción del 58 % de solicitudes con respecto a una simulación de monitorización no optimizada como se recoge en la Tabla 4.12.

Tabla 4.12: Resultados usando Métodos Optimizado (Esperanza-Maximización) respecto a No Optimizado para Tasa de Datos dentro del Caso de Estudio A

Método	Número de Peticiones	Reducción	Reducción (%)	Enlace	Diferencia
EM	888	1191	58 %	Enlace s_3-s_6	41.22 Kbps
No Optimizado	2079			Enlace s_6-s_5	52.07 Kbps

Por su parte, las Figuras 4.19a y 4.19b muestran el flujo de tráfico entre dos simulaciones en las que una aplica la optimización utilizando el algoritmo Esperanza-Maximización respecto de otro que no lo utiliza. La Figura 4.19a describe el flujo de datos (en bps) a través de los enlaces $s_3 - s_6$ ($switch : s_3 - puerto : p_4, switch :$

$s_6 - puerto : p_1$) y la Figura 4.19b los enlaces entre $s_6 - s_5$ ($switch : s_6 - puerto : p_2, switch : s_5 - puerto : p_3$). Las líneas puntiagudas (líneas azules) muestran la tasa de datos que el servidor está enviando el vídeo, mientras que las líneas continuas (líneas verdes) muestran la tasa de datos obtenida utilizando el método no optimizado y las líneas de puntos (líneas rojas) muestran la tasa de datos con algoritmo optimizado Como se esperaba, ambos enlaces experimentan un aumento en la tasa de datos debido a la transmisión del vídeo entre h_1 y h_2. Tan pronto como finaliza la transmisión (alrededor de 200 solicitudes), la tasa de datos disminuye, lo que demuestra la eficiencia del algoritmo para detectar cambios en la transmisión de la red. Del mismo modo, la diferencia media entre los enlaces $s_3 - s_6$ con/sin optimización es 41,22 Kbps y 52,07 Kbps en el enlace $s_6 - s_5$. Estos resultados confirman que la optimización mantiene buenos niveles de precisión y reduce el número de solicitudes en el plano de datos.



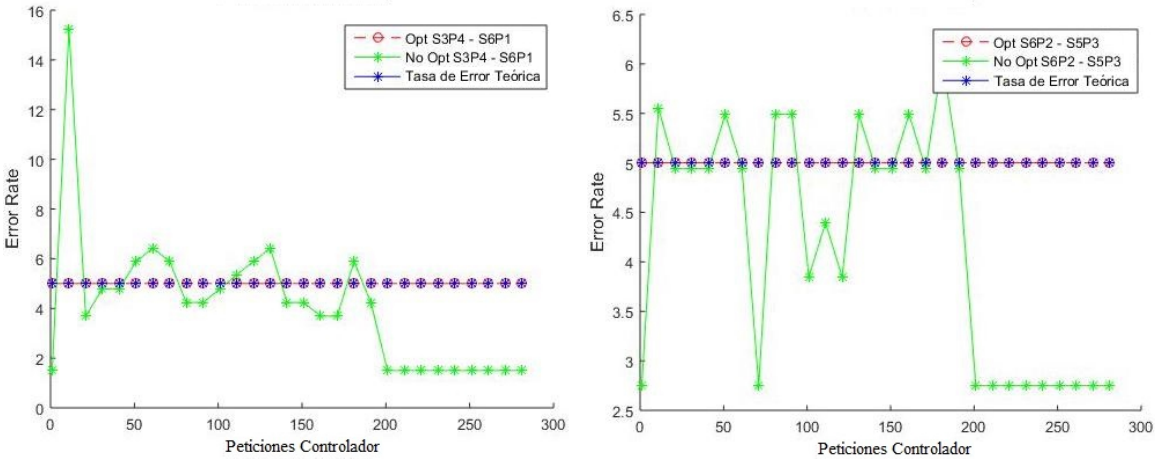
(a) Tasa de Datos del Enlace $s_3 - s_6$ (b) Tasa de Datos del Enlace $s_3 - s_6$

Figura 4.19: Tasa de Datos de Monitorización Optimizada (Esperanza-Maximización) respecto a No Optimizada

Continuando con la siguiente métrica, las Figuras 4.20a y 4.20b muestran la tasa de error en los datos del tráfico a través de la ruta de transmisión. La Figura 4.20a describe la tasa de error en el flujo de datos (en bps) a través de los enlaces $s_3 - s_6$ ($switch : s_3 - puerto : p_4, switch : s_6 - puerto : p_1$) y Figura 4.20b el enlace entre $s_6 - s_5$ ($switch : s_6 - puerto : p_2, switch : s_5 - puerto : p_3$). Las líneas continuas (líneas verdes) muestran la tasa de error obtenida utilizando el método no optimizado, las líneas punteadas (líneas rojas) muestran la tasa de error con el algoritmo optimizado y la línea continua punteada (líneas azules) muestra el porcentaje netem configurado en Mininet (5 % de pérdida de paquetes). Como se esperaba, los enlaces pueden detectar la pérdida de información entre los enlaces debido a las características de estos enlaces.

Del mismo modo, las simulaciones de la tasa de datos que aplican el algoritmo Basado en Densidad muestran un resultado de 1041 consultas (1038 menos) que supone una

relación de reducción del 50 % de solicitudes respecto de la simulación de monitorización no optimizada. La monitorización de resultados utilizando el algoritmo Basado en Densidad se detalla en la Tabla 4.13.



(a) Tasa de Error del enlace $s_3 - s_6$

(b) Tasa de Error del enlace $s_6 - s_5$

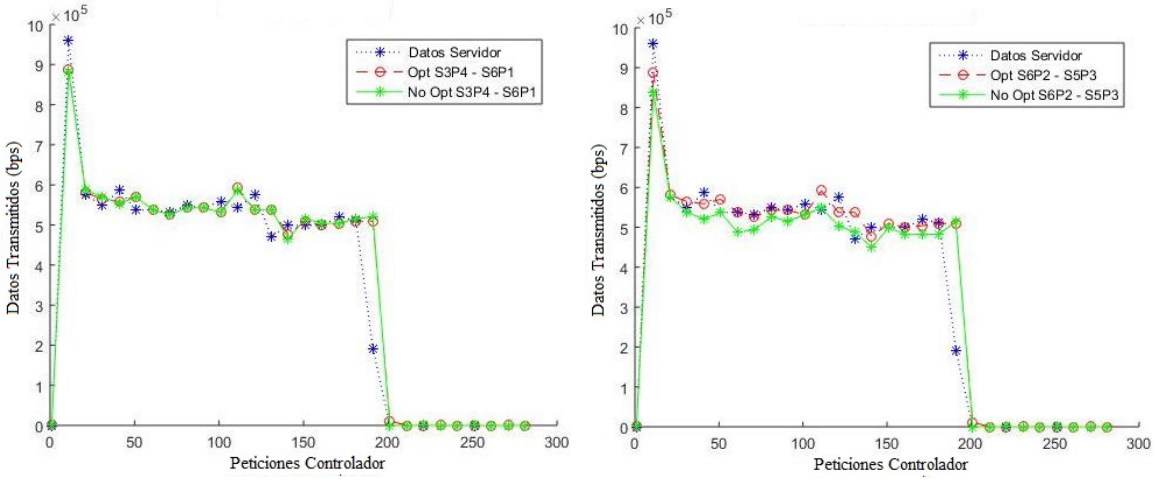
Figura 4.20: Tasa de Error de Monitorización Optimizada (Esperanza-Maximización) respecto a No Optimizada

Tabla 4.13: Resultados usando Métodos Optimizado (Basado en Densidad) respecto a No Optimizado para Tasa de Datos dentro del Caso de Estudio A.

Método	Número de Peticiones	Reducción	Reducción (%)	Enlace	Diferencia
Basado en Densidad	1041	1038	50 %	Enlace s_3-s_6	25.52 Kbps
No Optimizado	2079			Enlace s_6-s_5	37.51 Kbps

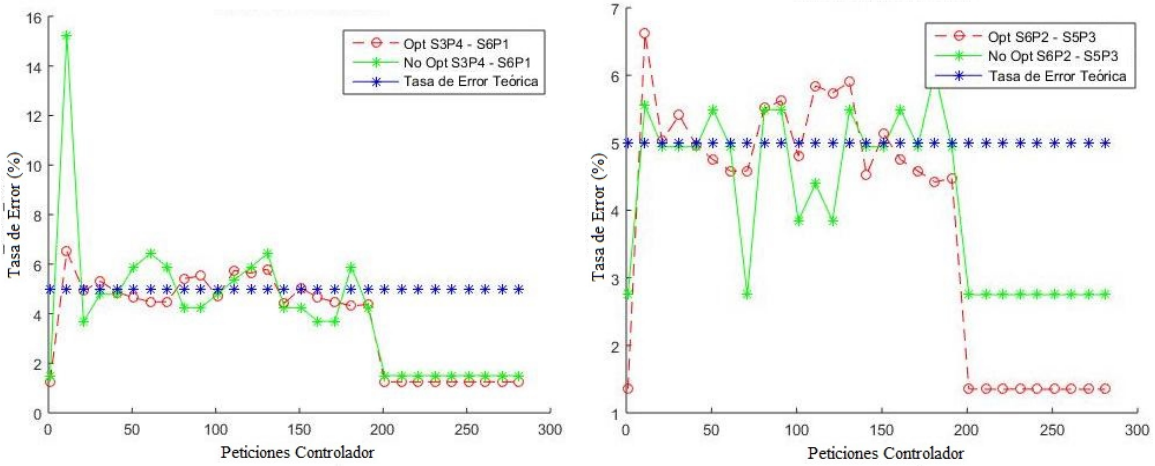
Aplicando el algoritmo de agrupamiento Basado en Densidad, la diferencia entre los enlaces $s_3 - s_6$ con/sin la optimización es de 25,52 Kbps (Figura 4.21a) y 37,51 Kbps en el enlace $s_6 - s_5$ (Figura 4.21b). Estos resultados confirman que la optimización mediante el algoritmo de agrupamiento Basado en Densidad proporciona una mayor precisión y reduce la cantidad de solicitudes en el plano de datos con respecto a los algoritmos K-medias y Esperanza-Maximización.

Continuando con la siguiente métrica, las Figuras 4.22a y 4.22b muestran la tasa de error en los datos de tráfico sobre el camino seguido en la transmisión de vídeo. La Figura 4.22a describe la tasa de datos en el flujo de datos (in bps) a través de enlace $s_3 - s_6$ ($switch : s_3 - puerto : p_4, switch : s_6 - puerto : p_1$) y la Figura 4.22b del enlace $s_6 - s_5$ ($switch : s_6 - puerto : p_2, switch : s_5 - puerto : p_3$).



(a) Tasa de Datos del enlace $s_3 - s_6$ (b) Tasa de Datos del enlace $s_6 - s_5$

Figura 4.21: Tasa de Datos de Monitorización Optimizada (Basado en Densidad) respecto a No Optimizada



(a) Tasa de Error del enlace $s_3 - s_6$ (b) Tasa de Error del enlace $s_6 - s_5$

Figura 4.22: Tasa de Error de Monitorización Optimizada (Basado en Densidad) respecto a No Optimizada

4.5.4. Evaluación del Caso de Estudio B

El segundo estudio, el Caso de Estudio B se centra en la escalabilidad utilizando una topología más compleja que la simulada en el Caso de Estudio A. Las simulaciones de este caso de estudio se realizan utilizando los tres algoritmos de agrupamiento descritos anteriormente (K-medias, Esperanza-Maximización y Basado en Densidad) y la tasa de datos y tasa de error como métricas.

4.5.4.1. Definición del Escenario

Las simulaciones del Caso de Estudio B son realizadas usando la topología descrita en la Figura 4.23 la cual está compuesta de 19 OF-Switches ($s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10}, s_{11}, s_{12}, s_{13}, s_{14}, s_{15}, s_{16}, s_{17}, s_{18} y s_{19}$) y 2 host ($h_1$ y h_2) conectados a s_1 y s_{19} respectivamente. Los enlaces $[switch : s_{15} - puerto : p_4, switch : s_{16} - puerto : p_1]$ y $[switch : s_{16} - puerto : p_2, switch : s_{19} - puerto : p_3]$ están configurados con valores de tasa de datos máxima de (1Mbps de ancho de banda) y pérdida de paquetes (5 % de los paquetes serán perdidos).

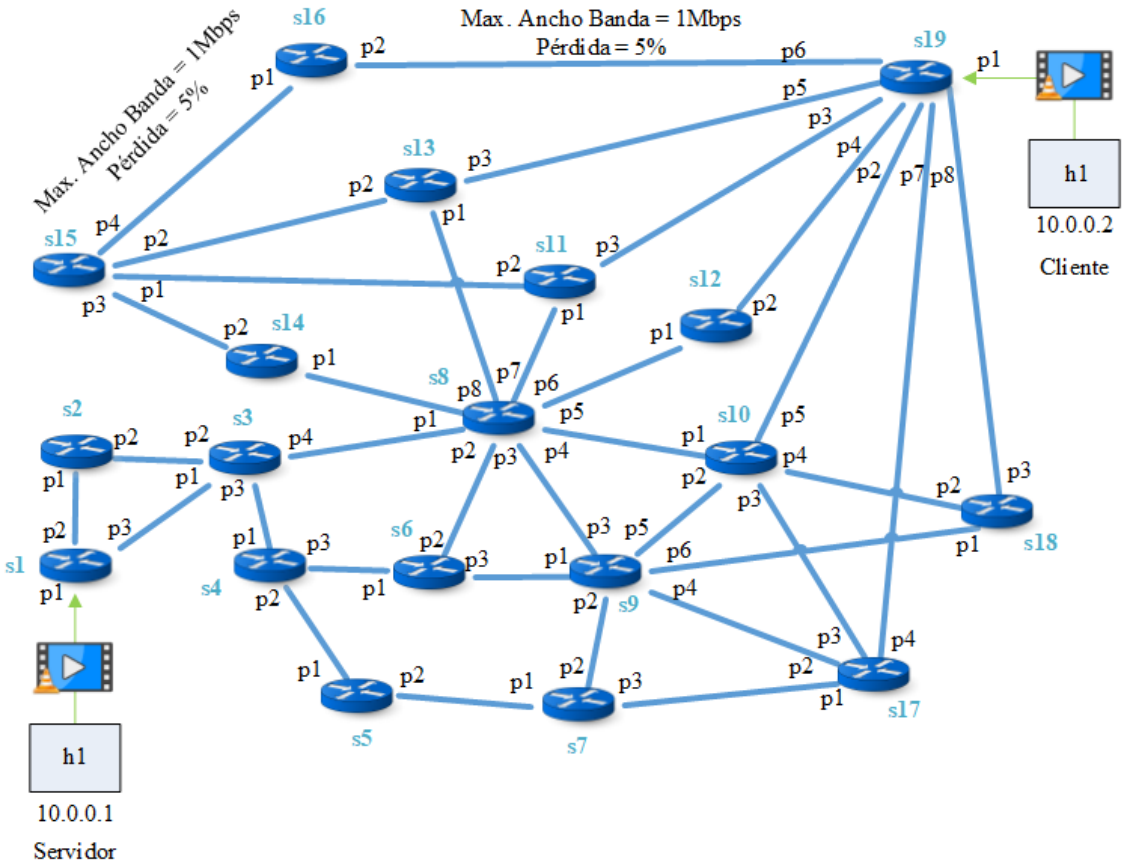


Figura 4.23: Topología de Prueba del Caso de Estudio B

La conexión entre el host cliente y el host servidor es una ruta fija compuesta por $[switch : s_1 - puerto : p_1, switch : s_1 - puerto : p_2, switch : s_2 - puerto : p_1, switch : s_2 - puerto : p_2, switch : s_3 - puerto : p_2, switch : s_3 - puerto : p_4, switch : s_8 - puerto : p_1, switch : s_8 - puerto : p_8, switch : s_{14} - puerto : p_1, switch : s_{14} - puerto : p_2, switch : s_{15} - puerto : p_3, switch : s_{15} - puerto : p_4, switch : s_{16} - puerto : p_1, switch : s_{16} - puerto : p_2, switch : s_{19} - puerto : p_6, switch : s_{19} - puerto : p_1]$.

4.5.4.2. Simulaciones y Resultados

Antes de comenzar a monitorizar los switches de red, cada algoritmo crea sus propios grupos en función de sus características. La Tabla 4.14 muestra la asignación de cada switch en un determinado grupo utilizando los algoritmos de agrupamiento.

Tabla 4.14: Asignación de Grupos en el Caso de Estudio B

Grupo	K-medias	Esperanza-Maximización	Basado en Densidad
Grupo 0 (c_0)	s_{10}	$s_2 s_6 s_{11} s_{13} s_{18}$	$s_8 s_9 s_{19}$
Grupo 1 ($c1$)	$s_1 s_4 s_6 s_7 s_{11} s_{13} s_{18}$	$s_8 s_{19}$	$s_1 s_4 s_5 s_7$
Grupo 2 (c_2)	$s_3 s_{15} s_{17}$	$s_1 s_3 s_4 s_7 s_{15} s_{17}$	$s_2 s_6 s_{11} s_{13} s_{18}$
Grupo 3 (c_3)	$s_2 s_5 s_{12} s_{14} s_{16}$	s_5	$s_3 s_{10} s_{15} s_{17}$
Grupo 4 (c_4)	$s_8 s_{19}$	$s_{12} s_{14} s_{16}$	s_{16}
Grupo 5 (c_5)	s_9	$s_9 s_{10}$	$s_{12} s_{14}$

Como se muestra en la tabla, la asignación de los switches no es la misma ya que cada algoritmo de agrupamiento tiene sus propias características para construir los grupos.

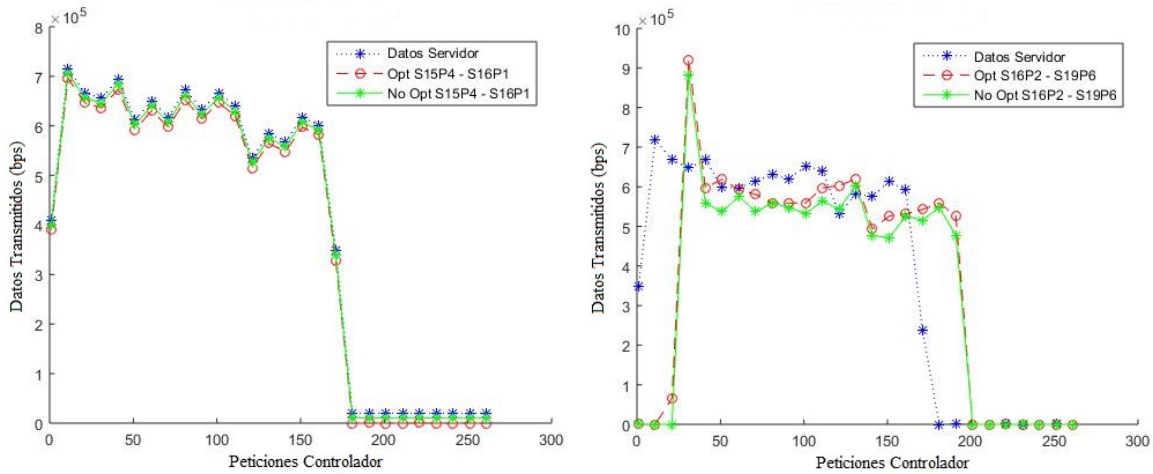
Aplicando el algoritmo de agrupamiento de K-medias, el número de consultas de monitorización fue de 2538, mientras que en el método no optimizado se obtuvieron 5225 solicitudes. Supone una relación de reducción del 52 % en las solicitudes con respecto a una simulación de monitorización no optimizada como se muestra en la Tabla 4.15.

Tabla 4.15: Resultados usando Métodos Optimizado (K-medias) respecto a No Optimizado para Tasa de Datos dentro del Caso de Estudio B

Método	Número de Peticiones	Reducción	Reducción (%)	Enlace	Diferencia
K-medias	2538	2687	52 %	Enlace s_{15}-s_{16}	23.32 Kbps
Non-Optimized	5225			Enlace s_{16}-s_{19}	48.61 Kbps

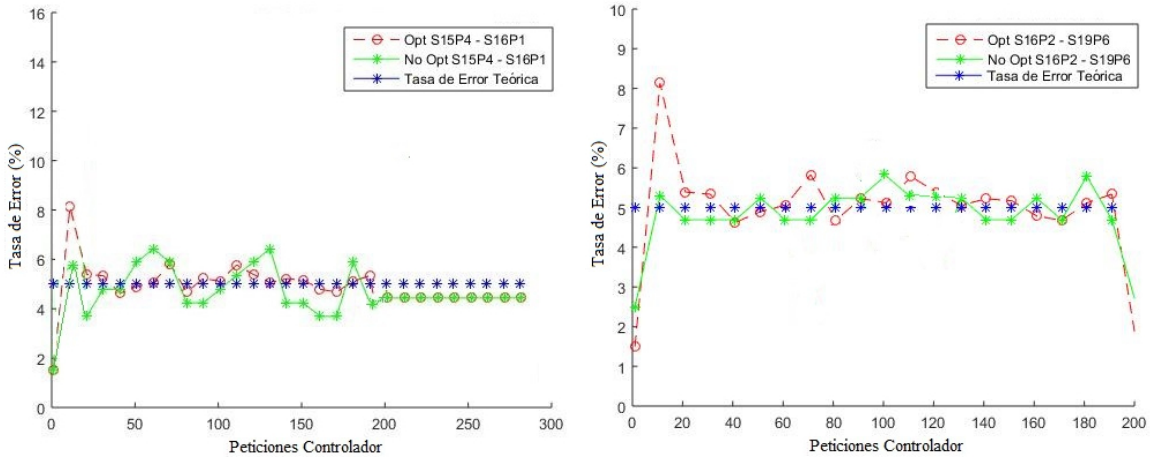
Las Figuras 4.24a y 4.24b muestran el flujo de tráfico entre dos simulaciones en las cuales una aplica la optimización usando el algoritmo K-medias respecto a otra que no la usa. La Figura 4.24a describe el flujo de datos (en bps) a través de los enlaces $s_{15} - s_{16}$ $(switch : s_{15} - puerto : p_4, switch : s_{16} - puerto : p_1)$ y la Figura 4.24b los enlaces entre $s_{16} - s_{19}$ $(switch : s_{16} - puerto : p_2, switch : s_{19} - puerto : p_6)$. La diferencia entre los enlaces $s_{15} - s_{16}$ con/sin la optimización es 23,32 Kbps y 48,61 Kbps en el enlace $s_{16} - s_{19}$.

Con respecto a la métrica de la tasa de error, las Figuras 4.25a y 4.25b muestran la tasa de error en los datos de tráfico sobre la ruta de transmisión. La Figura 4.25a describe la tasa de error en el flujo de datos (en bps) a través de los enlaces $s_{15} - s_{16}$ $(switch : s_{15} - puerto : p_4, switch : s_{16} - puerto : p_1)$ y la Figura 4.25b el enlace entre $s_{16} - s_{19}$ $(switch : s_{16} - puerto : p_2, switch : s_{19} - puerto : p_6)$.



(a) Tasa de Datos del Enlace $s_{15} - s_{16}$ (b) Tasa de Datos del Enlace $s_{16} - s_{19}$

Figura 4.24: Tasa de Datos de Monitorización Optimizada (K-medias) respecto a No Optimizada



(a) Tasa de Error del Enlace $s_{15} - s_{16}$ (b) Tasa de Error del Enlace $s_{16} - s_{19}$

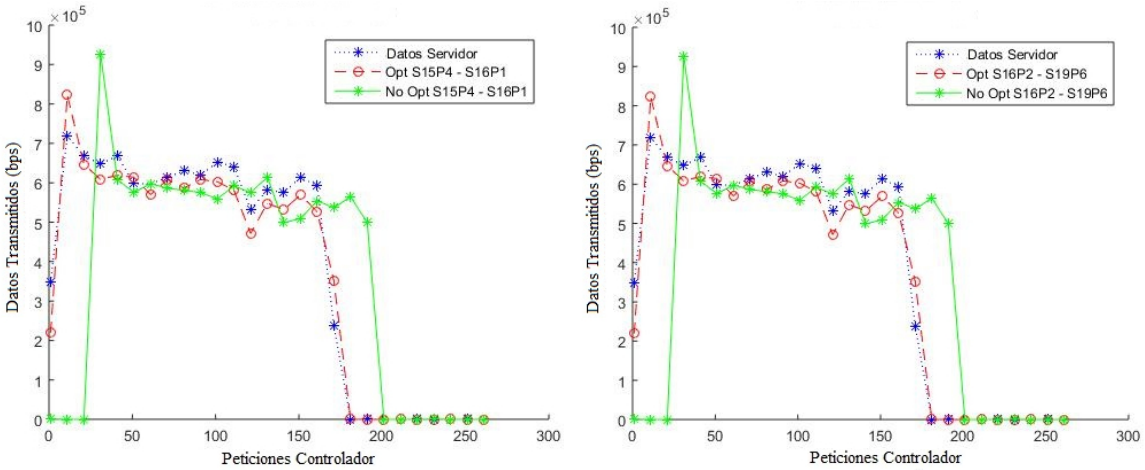
Figura 4.25: Tasa de Error de Monitorización Optimizada (K-medias) respecto a No Optimizada

Continuando con las simulaciones del algoritmo de Esperanza-Maximización, el número de consultas de monitorización es 1460. La diferencia de 3765 consultas (con el algoritmo no optimizado que es la solicitud 5225) muestra una relación de reducción del 72 % de solicitudes con respecto a una simulación de monitorización no optimizada como se resume en la Tabla 4.16.

Tabla 4.16: Resultados usando Métodos Optimizado (Esperanza-Maximización) respecto a No Optimizado para Tasa de Datos dentro del Caso de Estudio B

Método	Número de Peticiones	Reducción	Reducción (%)	Enlace	Diferencia
EM	1460	3765	72 %	Enlace s_{15}-s_{16}	47.51 Kbps
Non-Optimized	5225			Enlace s_{16}-s_{19}	57.38 Kbps

Las Figuras 4.26a y 4.26b muestran el flujo de tráfico entre dos simulaciones en las que una aplica la optimización utilizando el algoritmo Esperanza-Maximización respecto de otro que no lo utiliza. La Figura 4.26a describe el flujo de datos (en bps) a través de los enlaces $s_{15} - s_{16}$ $(switch : s_{15} - puerto : p_4, switch : s_{16} - puerto : p_1)$ y la Figura 4.26b los enlaces entre $s_{16} - s_{19}$ $(switch : s_{16} - puerto : p_2, switch : s_{19} - puerto : p_6)$. Del mismo modo, en promedio, la diferencia entre los enlaces $s_{15} - s_{16}$ con/sin la optimización es 47.51 Kbps y 57.38 Kbps en el enlace $s_{16} - s_{19}$.

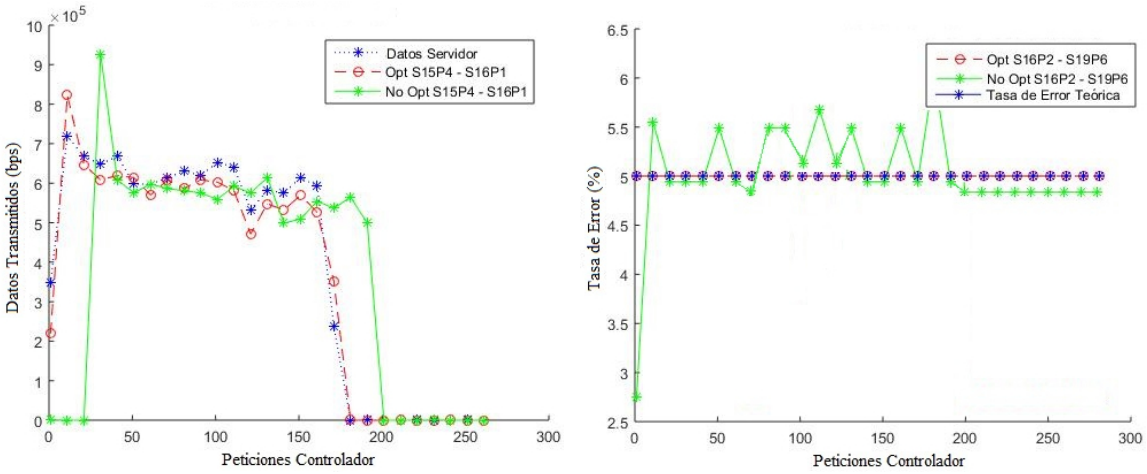


(a) Tasa de Datos del Enlace $s_{15} - s_{16}$

(b) Tasa de Datos del Enlace $s_{16} - s_{19}$

Figura 4.26: Tasa de Datos de Monitorización Optimizada (Esperanza-Maximización) respecto a No Optimizada

Continuando con la siguiente métrica, las Figuras 4.27a y 4.27b muestran la tasa de error en los datos de tráfico sobre la ruta de transmisión. La Figura 4.27a describe la tasa de error en el flujo de datos (en bps) a través de los enlaces $s_{15} - s_{16}$ $(switch : s_{15} - puerto : p_4, switch : s_{16} - puerto : p_1)$ y Figura 4.27b el enlace entre $s_{16} - s_{19}$ $(switch : s_{16} - puerto : p_2, switch : s_{19} - puerto : p_6)$. Como era de esperar, ambos enlaces pueden detectar la pérdida de información entre los enlaces debido a las características de estos enlaces.



(a) Tasa de Error del Enlace $s_{15} - s_{16}$

(b) Tasa de Error del Enlace $s_{16} - s_{19}$

Figura 4.27: Tasa de Error de Monitorización Optimizada (Esperanza-Maximización) respecto a No Optimizada

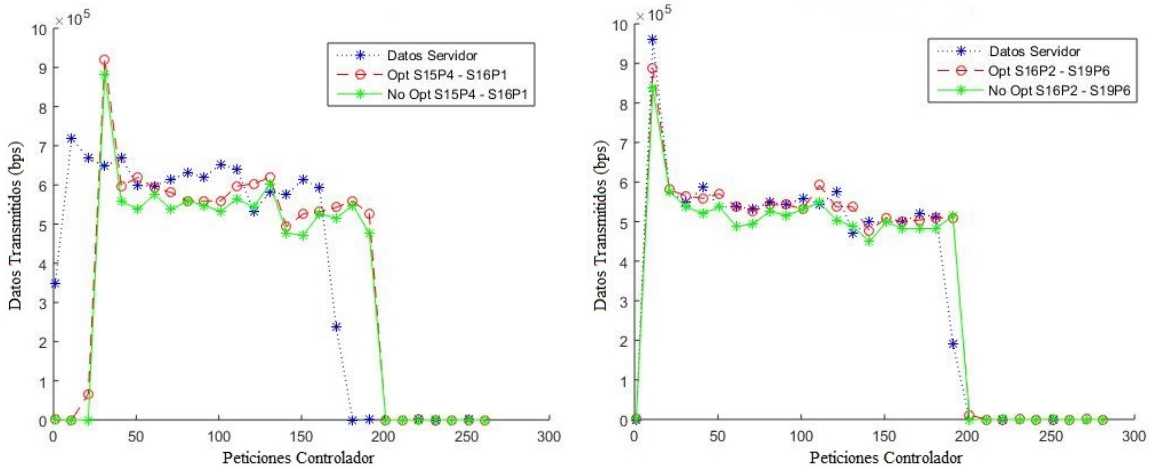
Finalmente, las simulaciones de la tasa de datos que aplican el algoritmo Basado en Densidad mostraron un resultado total de 1884 consultas (3341 menos) que supone una relación de reducción del 64 % de las solicitudes respecto a las simulaciones de monitorización no optimizada. Los resultados de monitorización usando el algoritmo Basado en Densidad se detalla en la Tabla 4.17.

Tabla 4.17: Resultados usando Métodos Optimizado (Basado en Densidad) respecto a No Optimizado para Tasa de Datos dentro del Caso de Estudio B

Método	Número de Peticiones	Reducción	Reducción (%)	Enlace	Diferencia
Basado en Densidad	1884	3341	64 %	Enlace s_{15}-s_{16}	38.14 Kbps
No Optimizado	5225			Enlace s_{16}-s_{19}	42.3 Kbps

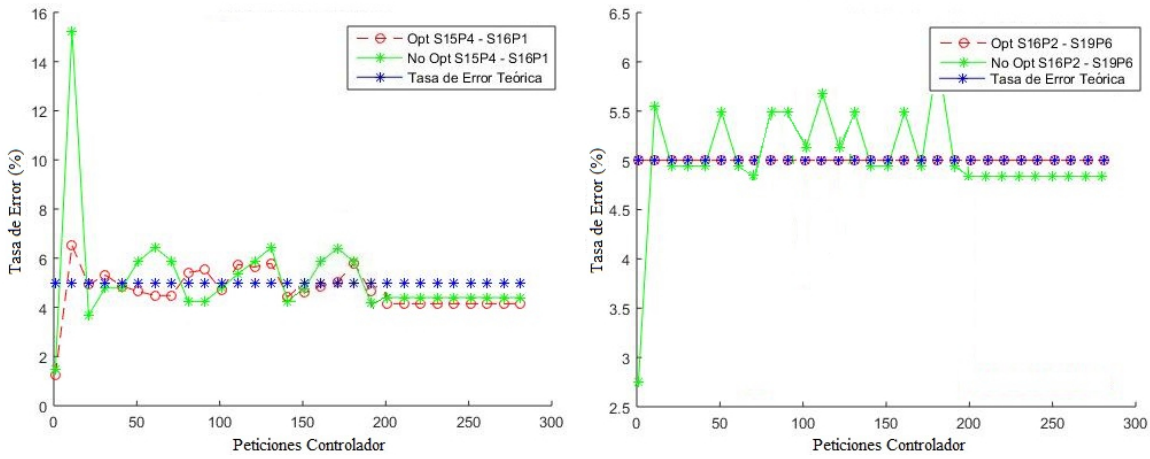
Aplicando el algoritmo de agrupamiento Basado en Densidad, la diferencia entre los enlaces $s_{15} - s_{16}$ con/sin la optimización es 38.14 Kbps y 42.3 Kbps en el enlace $s_{16} - s_{19}$. Los resultados proporcionados en las Figuras 4.28a y 4.28b confirman que la optimización mediante el algoritmo de agrupación basada en la densidad proporciona una mayor precisión y reduce el número de solicitudes en el plano de datos con respecto a los algoritmos K-medias y Esperanza-Maximización.

Los resultados, Figuras 4.29a y 4.29b muestran la tasa de error en los datos de tráfico a través de la ruta de transmisión. La Figura 4.29a describe la tasa de error en el flujo de datos (en bps) a través de los enlaces $s_{15} - s_{16}$ ($switch : s_{15} - puerto : p_4, switch : s_{16} - puerto : p_1$) y Figura 4.29b el enlace entre $s_{16} - s_{19}$ ($switch : s_{16} - puerto : p_2, switch : s_{19} - puerto : p_6$). Como era de esperar, ambos enlaces pueden detectar la pérdida de información entre los enlaces debido a las características de estos enlaces.



(a) Tasa de Datos del Enlace $s_{15} - s_{16}$ (b) Tasa de Datos del Enlace $s_{16} - s_{19}$

Figura 4.28: Tasa de Datos de Monitorización Optimizada (Basado en Densidad) respecto a No Optimizada



(a) Tasa de Error del Enlace $s_{15} - s_{16}$ (b) Tasa de Error del Enlace $s_{16} - s_{19}$

Figura 4.29: Tasa de Error de Monitorización Optimizada (Basado en Densidad) respecto a No Optimizada

Tomando estos resultados como referencia, se puede afirmar que el algoritmo Basado en Densidad mejora los resultados extraídos de los K-medias, pero no son mejores que los valores obtenidos utilizando el algoritmo Esperanza-Maximización.

Las principales tendencias extraídas del estudio del escenario de prueba A y el escenario de prueba B son las siguientes:

- El algoritmo K-medias ha convergido cuando las asignaciones ya no cambian.
- La asignación de grupos usando el algoritmo Basado en Densidad es muy similar a la asignación del K-medias en una topología pequeña.

- El mejor algoritmo de agrupamiento para el escenario de prueba A es el Esperanza-Maximización como se indica en los resultados. Esto se debe a que la asignación de los grupos no es tan uniforme como los algoritmos K-medias y Basado en Densidad.
- La tasa de error obtenida usando el algoritmo K-medias tiende al valor teórico de error configurado en la topología.
- La asignación de grupos después de aplicar el algoritmo Basado en Densidad se distribuye uniformemente.

Por lo tanto, teniendo en cuenta estas conclusiones, las ventajas de aplicar algoritmos de agrupamiento para monitorizar redes SDN son beneficiosas ya que disminuyen las consultas de estadísticas para reducir la sobrecarga de la red y liberar el controlador de tráfico.

Capítulo 5

Conclusiones

En este trabajo se han desarrollado tres técnicas de monitorización para mejorar el número de peticiones estadísticas en la monitorización de switches de una red SDN.

Se ha comenzado con una presentación de conceptos generales, como de la tecnología sobre la que se ha centrado toda la contribución de esta tesis doctoral, los componentes de dicha arquitectura y tecnologías subyacentes. Además, se ha puesto de manifiesto el concepto de virtualización. Dicho concepto persigue el aislamiento de múltiples redes lógicas, cada una de ellas con sus propios mecanismos de direccionamientos y reenvío, pero compartiendo la infraestructura física. SDN provee la oportunidad de alcanzar la capa de abstracción de un hardware. SDN es una arquitectura innovadora que propone la separación entre el plano de datos y plano de control, permitiendo su independiente evolución y desarrollo. Actualmente, la principal cristalización de SDN es OpenFlow, una arquitectura abierta diseñada inicialmente para realizar experimentos en redes heterogéneas, sin afectar el tráfico de usuarios reales. OpenFlow reúne las capacidades homogéneas del hardware de red actuales y las libera. La especificación OpenFlow establece las reglas de comunicación entre el plano de datos y un plano de control centralizado y permite el control del comportamiento de toda la red a través de aplicaciones de software API.

Las tecnologías de red se han convertido en un elemento crítico en prácticamente todas las actividades humanas. El número de dispositivos, así como la cantidad de tráfico transportándose a través del Internet continúa creciendo exponencialmente. Sin embargo, el desarrollo de nuevos protocolos y servicios (movilidad, QoS, transmisión de vídeo digital) ha sido limitado, principalmente por la fuerte unión entre el hardware especializado en la transmisión de paquetes y sistemas operativos ejecutando cientos de protocolos estáticos (algunos de ellos privados) y permitiendo al administrador de red únicamente la configuración del equipo de red.

Así, se han analizado las aplicaciones de las SDN sobre distintos ámbitos como las redes domésticas, redes móviles y contenido multimedia entre otros. De la misma forma, se ha analizado su contribución a la hora de resolver desafíos o problemas de otras tecnologías

como las redes de sensores WSN y las redes móvilas inalámbricas WCN.

Tras revisar los principales trabajos relacionados sobre las distintas técnicas de monitorización en redes SDN, se ha realizado una agrupación de los mismos en dos tipos atendiendo a si utiliza métodos activos o pasivos en la obtención de las estadísticas. Aunque hay algunas diferencias en mayor parte de los trabajos previos en el tratamiento de la información o el tipo de topología donde se aplican, se evalúa las estadísticas sobre los flujos de información que se producen en las mismas y la distribución de la información recolectada para su posterior procesamiento.

Del estudio de la literatura se han obtenido distintos enfoques y aproximaciones de las técnicas actuales que se utilizan para monitorizar redes SDN. La principal búsqueda de las herramientas de monitorización es la obtención de estadísticas lo más precisas posibles intentando no sobrecargar la red con mensajes al controlador para obtener dichas estadísticas. Con respecto a dichos resultados se han proporcionado las distintas contribuciones de esta tesis doctoral buscando la reducción del número de peticiones estadísticas en los switches de la red (disminuyendo así el flujo de mensajes al controlador). Por tanto, se ha extraído que el problema de optimización en el flujo de mensajes de control en una red SDN es una cuestión muy importante ya que distintas aplicaciones de esta red requieren de una alta demanda de computación y recursos.

A continuación, se han detallado las contribuciones de esta tesis proponiendo diferentes arquitecturas para reducir el número de mensajes de monitorización en una red SDN. Así, en primer lugar, se ha descrito el método desarrollado e implementado para la monitorización básica de una red SDN. El término monitorización básica se corresponde con la obtención de estadísticas de todos los switches de la topología. En esta contribución se han detallado todos los componentes funcionales básicos sobre los que se han apoyado las siguientes contribuciones de esta tesis doctoral. Esta contribución se ha evaluado con una transmisión de video entre dos puntos de una topología monitorizando todos los switches. Los resultados obtenidos de esta contribución son los más precisos posibles ya que el valor de las estadísticas obtenidas se corresponden con los del envío del video. Sin embargo, monitorizar todos los switches de una red no es nada rentable desde el punto de vista de sobrecarga y la escalabilidad.

A continuación y en segundo lugar, se ha mejorado la arquitectura inicial aportando una optimización que reduce el número de peticiones total eliminando las consultas a los switches con dos puertos siguiendo el principio de la Ley de Conservación de Flujo. Esta optimización se basa en que todo el tráfico entrante por el puerto de un switch será el mismo que el reenviado al puerto de salida del mismo switch. Los resultados obtenidos sobre las simulaciones realizadas usando esta optmización han reducido el número de consultas respecto a la monitorización básica en un 33 % y una diferencia de 30.5 Kbps entre los datos monitorizados y los valores del host transmisor.

En tercer lugar, se ha propuesto otra optimización para reducir el número de peticiones consultando sólo determinados switches agrupados en grupos. Antes de proceder al

cálculo de las estadísticas, se construyen los grupos usando el algoritmo de agrupamiento K-medias. Una vez que ya están construidos los grupos, se asigna una técnica de monitorización distinta a cada grupo de forma que: los grupos con switches de grados dos se les aplicará la técnica de la Ley de Conservación de Flujo expuesta en la segunda contribución de esta tesis doctoral. Por otro lado, los switches con grado tres o mayor se monitorizan en cada periodo de monitorización de forma aleatoria manteniendo una lista de switches monitorizados para que siempre se monitoricen todos los switches de cada grupo aunque sea de forma aleatoria. Terminada la etapa de asignación de monitorizarción, se procede a las simulaciones usando distintos videos para comprobar la factibilidad de la contribución obteniendo resultados muy satisfactorios manteniendo la precisión de los datos y disminuyendo el número de peticiones a los switches en un 57 %.

En cuarto y último lugar, se ha extendido la arquitectura de la tercera contribución con distintos algoritmos de agrupamiento (Esperanza-Maximización y Basado en Densidad). Se han dividido las simulaciones en dos escenarios (Caso de Estudio A y Caso de Estudio B) utilizando diferentes topologías en términos de tamaño, número de puertos y múltiples rutas para realizar la transmisión de vídeo desde el servidor al servidor del cliente. Esta propuesta se ha realizado para verificar la escalabilidad de la herramienta de monitorización propuesta en la tercera contribución y las ventajas/desventajas de aplicar un cierto algoritmo de agrupamiento o no. Una vez realizadas las simulaciones se han obtenido mejores resultados atendiendo a la construcción de los grupos usando el algoritmo de agrupamiento Esperanza-Maximización frente al Basado en Densidad.

Una vez obtenidos los resultados de las simulaciones usando cada una de las arquitecturas de monitorización se concluye que las contribuciones descritas en esta tesis doctoral han optimizado el objetivo principal de ésta, que es la optimización de las estadísticas en la monitorización de redes SDN.

5.1. Trabajos Futuros

En la actualidad, SDN se ha posicionado en gran lugar para ocupar las redes del futuro. A pesar de ofrecer múltiples ventajas con respecto a arquitecturas tradicionales, existen retos pendientes para completar su implementación en redes comerciales.

En primer lugar, se encuentra el encaminamiento entre diferentes dominios SDN y la coexistencia con arquitecturas tradicionales. Para solventar este problema, se propone una estructura jerárquica de controladores de tal manera que en vez de que la comunicación entre los controladores que rigen sus respectivos sistemas autónomos (conexión intra-controlador), habilitar un nuevo controlador en un nivel superior que maneje estos dos (conexión inter-controlador). De esta forma, el retardo aumentaría ligeramente por el incremento de mensajes con esta cantidad de controladores, pero se compensaría con la escalabilidad que proporcionaría este modelo.

Uno de los problemas relacionados con el control centralizado es la vulnerabilidad de

la red. Un fallo en el controlador puede afectar negativamente a la resistencia de la red y por tanto comprometiendo la información que está viajando en ella. Actualmente se está trabajando en arquitecturas de backup para que el controlador siga dando servicio en caso de fallos. Actualmente, la configuración de cada uno de los controladores de respaldo es posible pero no el mecanismo conexión y comunicación entre ellos. De igual manera, debido a que el control se encuentra centralizado, la posibilidad de ataques de denegación de servicio es mayor. Debido a la existencia única de un controlador SDN en las topologías clásicas de esta red y la demanda de mensajes que utiliza una herramienta de monitorización, introducir un controlador de respaldo (como un controlador "espejo") que esté sincronizado en todo momento con el controlador principal o maestro. Este hecho se puede realizar fijando un proceso/daemon que vuelque toda la información que contiene el controlador primario al secundario. Además, dicho controlador secundario analizaría los datos obtenidos por el controlador primario y los enviaría directamente al controlador original aumentando el flujo de mensajes entre los dos, pero liberando al controlador principal del procesamiento y carga de los datos que ya ha hecho el secundario.

Como se acaba de exponer, el punto débil de esta tecnología es la caída o fallo del controlador SDN. Siguiendo la línea de fallos en la red, ¿Qué pasaría si falla un enlace? Este desafío es muy importante en las comunicaciones multimedia y QoE (por ejemplo audio o transmisión de vídeo). El fin que se persigue es recuperar el camino de red cuando uno o varios enlaces han caído. Actualmente, se ha conseguido solventar este problema con un controlador que incorpora un mecanismo de recuperación de red con un camino alternativo cuando se produce un fallo. Por otra parte, se ha llegado a predecir o anticipar un posible fallo antes de que se produzca en la red basándose en la obtención de dos caminos disjuntos (en el caso que falla uno, utilizar el otro sin que haya ningún problema en la comunicación). El retardo que se produce calculando los dos caminos disjuntos iniciada ya la transmisión, se podría solventar calculando caminos alternativos junto con el camino principal antes de proceder al envío.

Respecto al controlador Floodligt y aprovechando que es una herramienta de código abierto, sería interesante incorporar algunas funciones predefinidas su API como por ejemplo una que obtenga el ancho de banda actual de un enlace en concreto. Dicha función se implementaría utilizando los métodos activos que se han descrito en esta tesis doctoral para obtener el ancho de banda y asignárselo al enlace en cuestión.

Por otro lado, un trabajo futuro muy importante sería facilitar la disponibilidad de las estadísticas a través de un bus común de información (bus Kafka) u otro tipo de pipeline de datos en tiempo real de tipo cliente-consumidor. De esta forma, se libera al controlador de más procesamiento de monitorización de los switches ya que la información de las estadísticas estaría disponible en el bus listo para ser recogido por la herramienta de monitorización dejando al controlador de la red aprovechable para otras tareas. Además, se pueden incluir motores en tiempo real de análisis de datos que estén consumiendo del bus común de información.

Otro reto que se plantea respecto a la contribución de esta tesis es ¿Cómo se pueden llevar lo que se ha aprendido con esta propuesta a otros tipos de redes emergentes como 5G?. la arquitectura 5G que dependerán en gran medida de tecnologías como SDN, NFV, *Multi-Access Edge Computing* (MEC) y *Fog Computing* (FC). Satisfacer las demandas de QoS y QoE de sus usuarios, fuerza a los operadores a desplegar nuevos tipos de infraestructura y tecnologías carrier grade, así como sistemas de antenas MIMO (con la capacidad de emitir en múltiples direcciones en el mismo momento), sistemas distribuidos (DAS), small cells y cabezas de radio remotas (RRH), entre otros. De este modo, la arquitectura propuesta en esta tesis doctoral puede ser extrapolada a la de 5G sustituyendo los switches por estaciones base de antenas, cambiando los host por dispositivos móviles que se desplazan en la topología.

Por último, un trabajo futuro a tener en cuenta es el tema de la seguridad. Las compañías despliegan una amplia variedad de mecanismos de seguridad en sus data centers para proteger sus aplicaciones de posibles ataques. Estos mecanismos son empleados a menudo con otras aplicaciones que realizan balanceo de carga, almacenamiento en caché y aceleración de aplicaciones. De esta forma, se propone añadir un mecanismo de seguridad en la herramienta de monitorización firmando y/o añadiendo un campo hash en el objeto de estadísticas de los paquetes que se envían al controlador para su posterior procesamiento. Una vez llegados los mensajes con los paquetes de estadísticas en el controlador, se verifica la firma o se calcula el hash y se comprueba con uno guardado de forma fija con el valor para la aplicación en cuestion y por tanto, dichos mensajes serán tratados como genuinos.

Apéndice A

Protocolo OpenFlow

A.1. Introducción

OpenFlow es un estándar creado por la Universidad de Stanford, inicialmente diseñado para permitir a los investigadores ejecutar protocolos experimentales en las redes de un campus, que provee un mecanismo estandarizado para ejecutar experimentos sin requerir la exposición de la estructura interna de los dispositivos de red. Actualmente, OpenFlow tiene soporte en switches Ethernet comerciales, routers y puntos de accesos inalámbricos.

A.2. Arquitectura OpenFlow

La arquitectura OpenFlow propone la existencia de un controlador, un switch OpenFlow y un protocolo seguro de comunicación entre ellos. Dichos elementos se muestran en la Figura A.1.

Cada switch OpenFlow está formado por tablas de flujo que son administradas desde el controlador. Cada tabla de flujo consta de tres elementos: packet header, action y statistics. El packet header es una máscara encargada de seleccionar los paquetes que van a ser procesados por el switch. Los campos que se utilizan para la comparación pueden ser indistintamente de la capa 2, 3 o 4 de la arquitectura TCP/IP. En otras palabras, no existe una separación entre capas como sucede en las arquitecturas actuales. Todos los paquetes que llegan al switch son filtrados por medio de este método. El número de campos que el switch puede procesar depende de la versión del protocolo OpenFlow utilizado. En la versión OpenFlow v1.0 [PLHea11], que es la versión más utilizada, existen 12 campos, mientras que la última versión disponible OpenFlow v1.3 define la existencia de 40 campos incluyendo soporte para IPv6.

Una vez que la cabecera de un paquete entrante coincide con el packet header del flowtable, las acciones correspondientes para esa máscara son ejecutadas por el switch. Existen acciones principales y opcionales. Las acciones principales son: reenviar el paquete a un puerto determinado, encapsular el paquete y enviarlo hacia el controlador y descartar

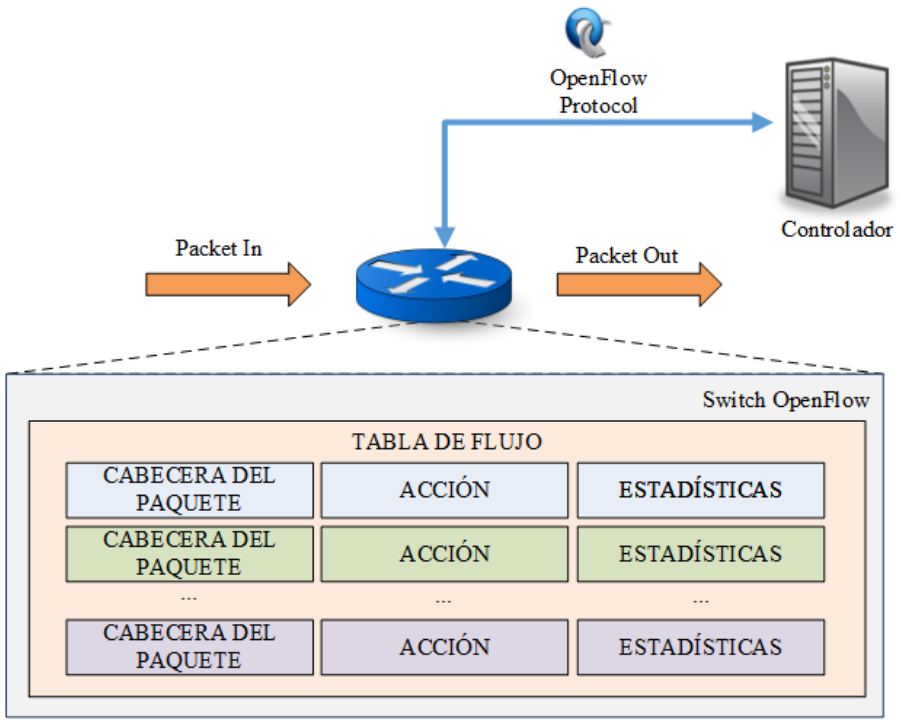


Figura A.1: Elementos de la Arquitectura OpenFlow

el paquete. Finalmente, el campo de statistics contabiliza entre otros la información del número de paquetes por cada flujo y se utiliza para fines de administración. En el caso de que la cabecera de un paquete entrante no coincide con el packet header del flowtable, el switch (según su configuración) envía dicho paquete hacia el controlador para su análisis y tratamiento.

A.3. Switch OpenFlow

Un switch OpenFlow consiste en una tabla de flujo (flow table) y un canal externo (secure channel) que se conecta al controlador. Estos componentes se pueden apreciar en la parte inferior de la Figura A.1.

El controlador maneja el comportamiento del switch a través del canal seguro utilizando el protocolo OpenFlow. El controlador puede añadir, actualizar y borrar información de la tabla de flujo, tanto reactivamente (en respuesta a paquetes) como proactivamente (generando acciones).

Cada tabla de flujo en el switch contiene un conjunto de entradas (flow entries). Éstas, a su vez, consisten en valores de cabecera (header values), contadores de actividad y un conjunto de cero o más acciones para aplicar a los paquetes. Cada vez que entra un paquete, el switch compara la cabecera del paquete con las entradas de la tabla de flujo. Si los campos coinciden, las instrucciones asociadas a ese flujo se ejecutan y, en caso contrario,

se envía el paquete al controlador por medio del canal seguro. Por tanto, el controlador es responsable de determinar cómo se manejan los paquetes sin entrada de flujo válida. Dichas instrucciones se envían al switch para reconfigurar la tabla de flujo, permitiendo que se envíen directamente los siguientes paquetes. Las acciones asociadas a las entradas son: envío del paquete por un puerto determinado, re-escritura de la cabecera del paquete y descartar paquete.

A.4. Tablas OpenFlow

En este apartado se describe los componentes de las tablas de flujo, además del mecanismo de comprobación de coincidencia y manejo de las acciones.

A.4.1. Tabla de Flujo

Una tabla de flujo, tal y como se muestra en la Tabla A.1, es una estructura que contiene 3 campos:

- Campos de Cabecera: se usan para hacer la comprobación de coincidencia de los paquetes entrantes.
- Contadores: se utiliza para registrar el número de paquetes coincidentes.
- Instrucciones: determinan las acciones que se ejecutarán con los paquetes cuyas cabeceras son idénticas a los campos coincidentes.

Tabla A.1: Cabeceras de una Tabla de Flujo

Campo de Cabecera	Contadores	Instrucciones

La Tabla A.2 muestra los campos de cabecera que pueden ser utilizados para la comparación con los paquetes entrantes. Cada entrada contiene un valor específico o el valor ANY para un valor arbitrario. Los campos coincidentes pueden ser indistintamente de la capa 2, 3 o 4 de la arquitectura TCP/IP.

Tabla A.2: Longitud de los Campos y la Manera que son Aplicados a las Tablas de Flujo

Campo	Bits	Aplicable a	Notas
Ingress Port	(depende de la implementación)	Todos los paquetes	Representación numérica del puerto de entrante, empezando en 1.
Ethernet source address	48	Todos los paquetes en puertos habilitados	
Ethernet destination address	48	Todos los paquetes en puertos habilitados	
Ethernet Type	16	Todos los paquetes en puertos habilitados	Un switch OpenFlow es requerido para la comprobar la coincidencia del tipo tanto en el estándar Ethernet como 802.2 con una cabecera SNAP y OUI de valor 0x000000. El valor especial 0x05FF es usado para coincidir con todos los paquetes 802.3 sin cabeceras SNAP.
VLAN id	12	Todos los paquetes del tipo Ethernet 0x8100	
VLAN priority	3	Todos los paquetes del tipo Ethernet 0x8100	
IP source address	32	Todos los paquetes IP y ARP	Puede ser enmascarado por subred.
IP destination address	32	Todos los paquetes IP y ARP	Puede ser enmascarado por subred.
IP protocol	8	Todos los paquetes IP e IP sobre Ethernet. Paquetes ARP	Solo los 8 bits menos significativos son usados para el ARP opcode.
IP ToS bits	6	Todos los paquetes IP	Especifica un valor de 8 bits y esta localizado en los 6 bits superiores de ToS.
Transport source port / ICMP type	16	Todos los paquetes TCP, UDP e ICMP	Solo los 8 bits menos significativos son usados para tipo ICMP.
Transport destination port / ICMP code	16	Todos los paquetes TCP, UDP e ICMP	Solo los 8 bits menos significativos son usados para tipo ICMP.

El tratamiento que un paquete recibe cuando entra en el switch se describe en la Figura A.2.

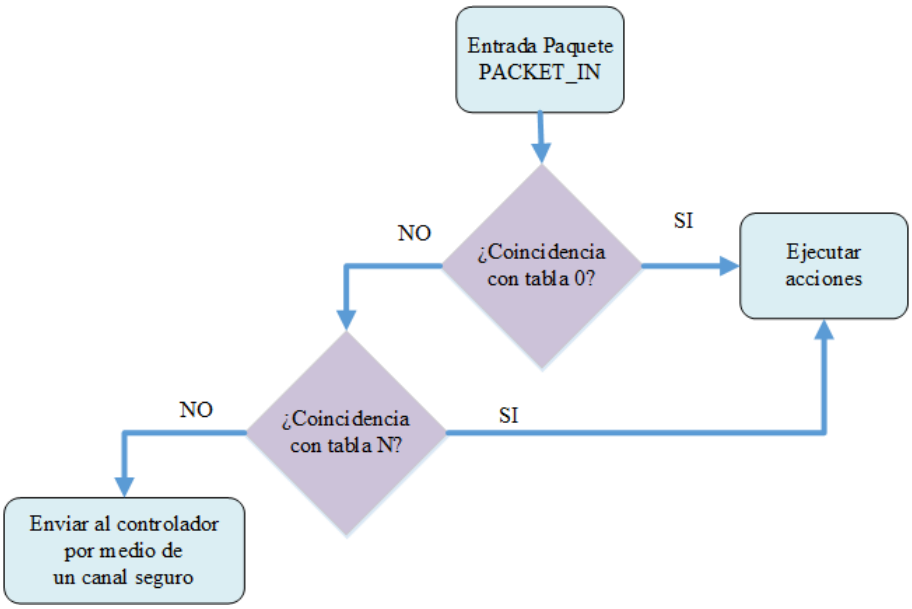


Figura A.2: Procesamiento de un Paquete en un Switch OpenFlow

Como se explicó anteriormente, el switch compara el paquete con los campos de la tabla de flujo; en caso de coincidencia ejecuta las acciones, actualiza contadores y busca en la siguiente tabla. En caso de que el paquete sea desconocido, el switch (según su configuración) descarta o encapsula el paquete y lo envía al controlador. Las diferentes acciones que se pueden ejecutar se dividen en requeridas y opcionales:

- Acción Requerida: Forward. Esta acción envía un paquete por un puerto específico (tanto físico como virtual). Los puertos estándar son definidos como: puertos físicos, virtuales (definidos por el controlador) y el puerto LOCAL. Además, los switches OpenFlow soportan estas opciones adicionales de envío: - ALL: envía el paquete de salida a todos los puertos estándares, menos el puerto de entrada. - CONTROLLER: encapsula y envía el paquete al controlador. - LOCAL: envía el paquete a la pila de red del switch local. - TABLE: realiza acciones en la tabla de flujo. Sólo para mensajes Packet-Out. - IN PORT: envía el paquete al puerto de entrada.

- Acción Opcional: Forward. El switch tiene la opción de soportar las siguientes acciones en los puertos virtuales: - NORMAL: procesa el paquete usando algoritmos tradicionales (no-OpenFlow) del switch. El switch comprueba el campo VLAN para determinar si se puede o no enviar el paquete a través de la ruta normal de procesamiento. En caso contrario, el switch envía un mensaje indicando que no soporta esta acción. - FLOOD: inunda el paquete sobre toda la red, excluyendo la interfaz de entrada.

- Acción Opcional: Enqueue. Este tipo de acción envía un paquete a través de la

cola adjunta a un puerto. El comportamiento de envío está determinado por la configuración de la cola y suele proveer soporte básico QoS.

- Acción Requerida: Drop. El switch descarta todos los paquetes que coinciden con una tabla de flujo configurada sin acciones.

- Acción Opcional: Modify-Field. Este tipo de acción modifica los valores de las respectivas cabeceras en un paquete.

A.5. Canal OpenFlow

El canal OpenFlow es la interfaz que conecta el switch OpenFlow con el controlador. Dicha interfaz es conocida como protocolo OpenFlow y por medio de ella se realizan las siguientes acciones:

- Configura y actualiza el switch.

- Recibe eventos procedentes del switch.

- Envía paquetes al switch.

A.5.1. Protocolo OpenFlow

El protocolo OpenFlow define los siguientes tipos de mensajes entre el switch y el controlador: controller to switch, symmetric y asynchronous. Los mensajes tipo controller to switch gestionan el estado del switch, los asynchronous actualizan el control de los eventos de la red y cambios al estado del switch. Los symmetric son enviados ya sea por el controlador o por el switch para iniciar la conexión o intercambio de mensajes. De igual manera, se definen 2 tipos de switches: OpenFlow-only y OpenFlow-enabled, según tengan la capacidad de trabajar únicamente con OpenFlow o puedan también procesar el paquete utilizando algoritmos tradicionales de conmutación o de encaminamiento.

En resumen, el protocolo OpenFlow [PLHea11] soporta tres tipos de mensajes. Éstos son:

- Controlador a switch: iniciado por el controlador y usado para inspeccionar el estado del switch.

- Asíncronos: iniciado en el switch y usado para actualizar la información del controlador cuando se producen eventos en la red y cambios de estado en el switch.

- Simétricos: iniciados por ambos y enviados sin petición.

A.5.1.1. Mensajes Controlador a Switch

Este tipo de mensajes son iniciados por el controlador y pueden o no tener respuesta por parte del switch. Estos tipos de mensajes son:

- Features: el controlador solicita las capacidades de un switch enviando una petición de características (features request). El switch responde con una respuesta a la petición (features reply).
- Configuration: el controlador está capacitado para cambiar y hacer peticiones sobre parámetros de configuración en el switch. El switch sólo responde a una petición generada desde el controlador.
- Modify-State: estos mensajes son enviados por el controlador para administrar estados en los switches. Su propósito principal es añadir, eliminar o modificar flujos en las tablas OpenFlow. Además, permiten cambiar las propiedades de los puertos del switch.
- Read-State: son usados por el controlador para coleccionar estadísticas del switch.
- Send-Packet: estos mensajes son usados por el controlador para enviar paquetes por un puerto específico del switch y para reenviar paquetes previamente recibidos.
- Barrier: los mensajes de petición/respuesta de barrera son usados por el controlador para asegurar dependencias que haya tenido o para recibir notificaciones de operaciones completadas.

A.5.1.2. Mensajes Asíncronos

Estos mensajes son enviados sin la petición del controlador al switch. Estos mensajes denotan la llegada de un paquete, cambios de estado en el switch o errores. Los cuatro principales tipos de mensajes asíncronos son:

- Packet-in: Este mensaje encapsula un paquete y lo envía al controlador para su procesamiento. Este mensaje es enviado cuando no existe una tabla asociada a la cabecera del paquete.
- Flow-removed: Informa al controlador que un elemento de la tabla de flujo ha sido eliminado del switch.
- Port-status: el switch envía mensajes de este tipo al controlador para informar sobre cambios de estados en la configuración del puerto.
- Error: el switch notifica al controlador de problemas existentes.

A.5.1.3. Mensajes Simétricos

Estos mensajes son enviados bidireccionalmente sin petición. Los mensajes simétricos son:

- Hello: son intercambiados entre el switch y el controlador a la hora de establecer la conexión.
- Echo: los mensajes Echo pueden ser enviados tanto por el controlador como por el switch y se envían siempre en respuesta a una petición. Son usados para medir la latencia o el ancho de banda de una conexión controlador- switch.
- Vendor: este tipo de mensajes proveen a los switches OpenFlow una forma estándar de ofrecer funcionalidad adicional dentro del espacio del mensaje OpenFlow.

A.5.2. Mensajes Lectura de Estadísticas

La sección 5.3.5 Read State Messages de [Ope10] describe la estructura de los mensajes de petición y respuesta sobre las estadísticas a un switch. El mensaje $OFPT\_STATS\_REQUEST$ contiene los siguientes campos:

struct $ofp\_stats\_request$ {
    struct ofp_header header;
    uint16_t type; * One of the OFPST_* constants. */
    uint16_t flags; * $OFPSF\_REQ\_*$ flags (none yet defined). */
    uint8_t body[0]; * Body of the request. */
};

OFP_ASSERT(sizeof(struct $ofp\_stats\_request$) == 12);

El switch responde con uno o varios mensajes OFPT_STATS_REPLY:
struct ofp_stats_reply {
    struct $ofp\_headerheader$;
    uint16_t type; One of the OFPST_* constants.
    uint16_t flags; $OFPSF\_REPLY\_*$ flags.
    uint8_t body[0]; Body of the reply.
};

OFP_ASSERT(sizeof(struct ofp_stats_reply) == 12);

El único valor definido para las banderas en una respuesta es si más respuestas seguirán a esta, éstas tendrán el valor 0x0001. Para facilitar la implementación, el switch puede

enviar respuestas sin entradas adicionales. Sin embargo, siempre debe enviar otra respuesta después de un mensaje con el conjunto de banderas. Los identificadores de transacción (xid) de las respuestas siempre deben coincidir con la solicitud que los solicitó.

Tanto en la solicitud como en la respuesta, el campo de tipo especifica el tipo de información que se pasa y determina cómo el campo cuerpo es interpretado:

```
enum ofp_stats_types {
    /* Description of this OpenFlow switch.
    * The request body is empty.
    * The reply body is struct ofp_desc_stats. */
    OFPST_DESC,
    /* Individual flow statistics.
    * The request body is struct ofp_flow_stats_request.
    * The reply body is an array of struct ofp_flow_stats. */
    OFPST_FLOW,
    /* Aggregate flow statistics.
    * The request body is struct ofp_aggregate_stats_request.
    * The reply body is struct ofp_aggregate_stats_reply. */
    OFPST_AGGREGATE,
    /* Flow table statistics.
    * The request body is empty.
    * The reply body is an array of struct ofp_table_stats. */
    OFPST_TABLE,
    /* Physical port statistics.
    * The request body is struct ofp_port_stats_request.
    * The reply body is an array of struct ofp_port_stats. */
    OFPST_PORT,
    /* Queue statistics for a port
    * The request body defines the port
    * The reply body is an array of struct ofp_queue_stats */
    OFPST_QUEUE,
    /* Vendor extension.
    * The request and reply bodies begin with a 32-bit vendor ID, which takes
    * the same form as in "struct ofp_vendor_header". The request and reply
    * bodies are otherwise vendor-defined. */
    OFPST_VENDOR = 0xffff
};
```

A.6. Ventajas de OpenFlow

Como se ha comentado anteriormente, OpenFlow fue inicialmente propuesto como alternativa para el desarrollo de protocolos experimentales en campus universitarios, donde se puedan probar nuevos algoritmos sin necesidad de interrumpir o interferir con el funcionamiento normal del tráfico de otros usuarios. Hoy en día, la Open Networking Foundation ONF [ONF17] es la organización encargada de la publicación del protocolo OpenFlow, entre otros protocolos SDN como, por ejemplo, OF-Config [Ope13].

La ventaja que OpenFlow presenta con respecto a protocolos SDN previos radica en que OpenFlow aprovecha elementos y funciones de hardware ya disponibles en la mayoría de los equipos de red. Estos elementos son las tablas de encaminamiento y las funciones comunes como leer la cabecera, enviar el paquete a un puerto, descartar paquete, entre otros. OpenFlow abre estos elementos y funciones para que puedan ser controlados externamente. Esto implica que basta con una actualización de firmware para que el mismo hardware pueda ser ya compatible con OpenFlow. De esta manera las empresas no necesitan realizar un cambio completo de su hardware para implementar SDN en sus productos y servicios.

El controlador recibe la información de los diferentes switches y configura remotamente las tablas de flujo de los switches. Es en el controlador, donde el usuario puede literalmente programar el comportamiento de la red. A diferencia de las redes activas que proponían un Node Operating System, OpenFlow abre la noción de un NOS. En este aspecto, en [FRZ14] se define al NOS como el software que abstrae la instalación del estado en los switches de red de la lógica y aplicaciones que controlan el comportamiento de la red. En los últimos años los NOS han ido evolucionando según las necesidades y aplicaciones de los investigadores y administradores de red.

Apéndice B

Sistema Operativo de Red

B.1. Introducción

Un NOS describe a todas las herramientas de software que permiten crear aplicaciones y controlar el comportamiento de la red. Los NOS se clasifican según el lenguaje de programación que utilizan. Cada lenguaje tiene sus ventajas en función de administración de la memoria, soporte multiplataforma y rendimiento. En la Tabla B.1 se resumen los principales NOS.

Tabla B.1: NOS en Función del Lenguaje de Programación

Lenguaje	Nombre del Controlador
C/C++	NOX, Trema, MUL
Haskell	Nettle, McNettle, NetCore
Java	Maestro, Floodlight, Beacon
OCaml	Mirage, Frenetic
Python	POX, Pyretic, RYU

B.2. Evolución de los Sistemas Operativos de Red

El concepto de (NOS) se basa en la función de un sistema operativo en el campo de computación, es decir, el sistema operativo permite al usuario crear aplicaciones usando abstracción de alto nivel de los recursos de información y de hardware. En SDN algunos autores [RFR+12] [SSHC+13] [KF13] han clasificado las abstracciones de los recursos de red como interfaces Southbound y Northbound (Figura B.1).

Las interfaces tipo Southbound tienen la función de abstraer la funcionalidad de los switches programables y conectarse con el software controlador. Un claro ejemplo de interfaz Southbound es OpenFlow. Sobre las interfaces Southbound se ejecuta un NOS. Ejemplos de NOS son: NOX [GKP+08], Beacon [Eri13], Floodlight [Flo17], entre otros. Por otro lado, las interfaces Northbound permiten crear aplicaciones o políticas de red de

alto nivel y transmiten dichas tareas al NOS. Ejemplos de estas interfaces son: Frenetic [FHF+11] [FGR+13], Procera [VKF12] [KF13], Netcore [MFHW12]; McNettle [VW12]. A continuación se analizan los principales NOS e interfaces Northbound.

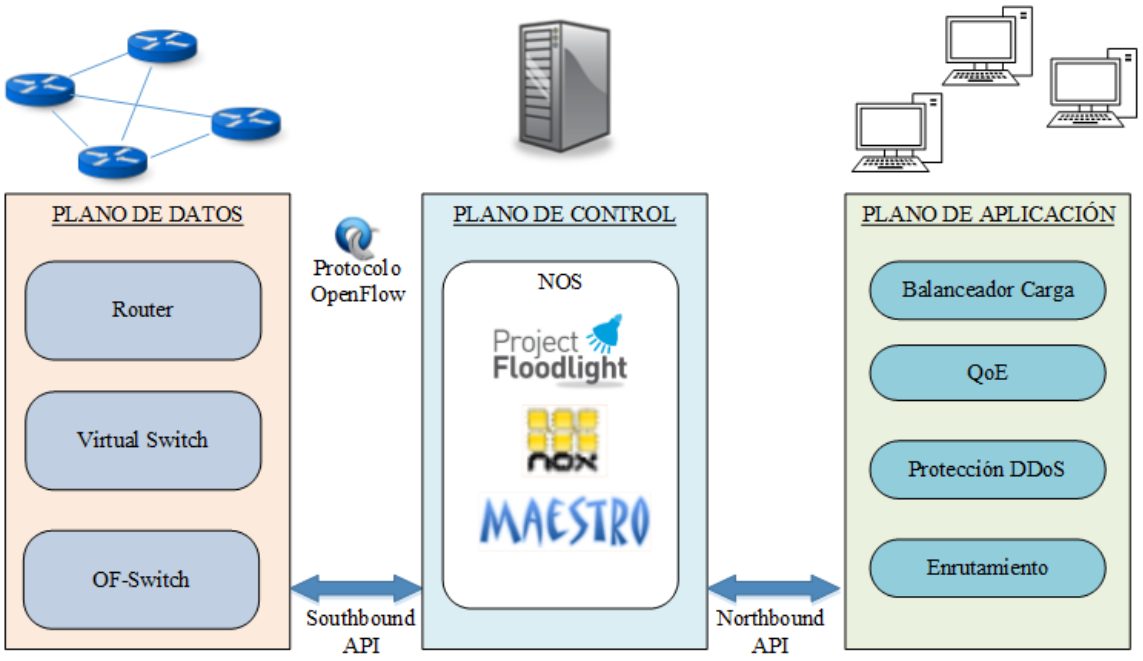


Figura B.1: NOS e Interfaces Northbound y Southbound

El software NOX [GKP+08] es el primer NOS para OpenFlow y está constituido por 2 elementos: procesos del controlador y una visión global de la red. En función del estado actual de la red, el usuario puede tomar decisiones y configurar el comportamiento de la red por medio de dichos procesos. En NOX el tráfico se maneja a nivel de flujos, es decir, todos los paquetes con la misma cabecera, son tratados de manera similar. El controlador inserta, elimina entradas y lee los contadores que se encuentran en las tablas de flujo (flow tables) de los switches. Por otro lado, debido a la naturaleza dinámica del tráfico NOX usa eventos (event handlers) que son registrados con diferentes prioridades para ejecutarse cuando se produce un evento específico en la red. Los eventos más utilizados son: switch join, switch leave, packet received y switch statistics received. Asimismo, NOX incluye system libraries con implementaciones y servicios de red comunes. Finalmente, NOX es implementado en C++ ofreciendo alto rendimiento. Existe una implementación enteramente en Python denominado POX, que proporciona un lenguaje de desarrollo más amigable.

Beacon [Eri13] es un controlador OpenFlow basado en Java. Su interfaz es simple y sin restricciones, es decir, el usuario puede usar libremente los constructores disponibles en Java (threads, timers, sockets, ...). Por otro lado, Beacon es un NOS basado en eventos, es decir, el usuario configura los sucesos que monitoriza el controlador. La interacción

con los mensajes OpenFlow del switch se realiza mediante la librería OpenFlowJ, una implementación del protocolo OpenFlow 1.0 [Ope10], y la interfaz IBeaconProvider que contiene los listeners IOFSwitchListener, IOFInitializerListener y IOFMessageListener. Beacon tiene soporte multithreading y facilita implementaciones APIs importantes (Device Manager, Topology, Routing, Web UI), así como la capacidad de iniciar, agregar y terminar aplicaciones sin terminar completamente un proceso (runtime modularity).

OpenDaylight [MVTG14] es un NOS que utiliza ingeniería de software basada en modelos (MDSE) y gestión de red basada en modelos para proporcionar flexibilidad y escalabilidad en el desarrollo de aplicaciones SDN. De esta manera, los usuarios pueden incluir múltiples servicios y aplicaciones utilizando los complementos Southbound de OpenDaylight. Se utilizan lenguajes de modelado como YANG junto con los protocolos NETCONF / RESTCONF para facilitar a los desarrolladores el diseño de servicios/aplicaciones de red. Por su parte, ONOS [BGH+14] es un NOS enfocado en proporcionar una plataforma de control SDN distribuida. Para este propósito, propone una arquitectura distribuida para garantizar la disponibilidad y la ampliación. Del mismo modo, proporciona una vista de red global y un control centralizado lógicamente a pesar de que los servidores se pueden distribuir en múltiples ubicaciones.

A pesar que un NOS puede manejar las tablas de flujo de los switches, existen algunos problemas que pueden ocasionar el mal funcionamiento de la red [RFR+12] [SSHC+13] [GRF13]. Por ejemplo, el controlador recibe el primer paquete que llega al switch y que no tiene un flow table asignado. Luego, el controlador lo analiza, asigna acciones y reenvía esas instrucciones al switch para que los demás paquetes similares tengan el mismo camino. Sin embargo, durante ese tiempo puede llegar el segundo, tercer o cuarto paquete similar al controlador y ocasionar un funcionamiento errático. En otras palabras, virtualmente existen dos procedimientos en ejecución: uno en el controlador y otro en el switch, y dichos procedimientos no se encuentran completamente sincronizados.

Otra limitación es la composición, es decir, si el usuario desea configurar dos servicios diferentes en el mismo switch (por ejemplo, encaminamiento y monitorización), se tiene que combinar manualmente ambas acciones en el switch, asignar prioridades, mantener la semántica según cada elemento de la red. Esto hace muy difícil el diseño, coordinación y reutilización de librerías. Además, el switch tiene que manejar 2 tipos de mensajes simultáneamente: paquetes y mensajes de control. Cualquier descoordinación puede ocasionar que un paquete sea procesado con una política inválida y, consecuentemente, causar un problema de seguridad importante en la red. Por ejemplo, si en una tabla de flujo existen dos entradas con la misma prioridad, el comportamiento del switch podría ser no determinista, ya que la ejecución dependería del diseño del hardware del switch. Para superar este tipo de inconvenientes, la comunidad investigadora ha trabajado en el desarrollo de interfaces más simples que interactúen y coordinen el correcto funcionamiento en el switch (Northbound).

Procera [VKF12] [KF13] es un framework que permite expresar políticas o

configuraciones de red de alto nivel. Esta arquitectura establece diferentes dominios de control y acciones con las que el usuario programa el comportamiento de la red. Los principales dominios de control son: Time, Data Usage, Status y Flow. Con estos dominios el usuario puede determinar un comportamiento dependiendo, por ejemplo, de la hora del día, cantidad de datos transmitidos, privilegios o grupos de usuarios, tipo de tráfico transmitido, etc. Las acciones pueden ser temporales o reactivas y están expresadas en un lenguaje de alto nivel basado en *Functional Reactive Programming* (FRP) y Haskell. En [KF13] se encuentran los detalles de este lenguaje, así como ejemplos del uso en aplicaciones de monitorización y control de usuarios en un campus.

Frenetic [FHF+11] [FGR+13] es un lenguaje de alto nivel para redes SDN desarrollado en Python. Está estructurado en 2 sub-lenguajes: Network Query Language y Reactive Network Policy Management Library. Network Query Language permite al usuario leer el estado de la red. Esta tarea se realiza mediante la instalación de reglas de bajo nivel (low-levels rules) en el switch que no afectan al funcionamiento normal de la red. Por otro lado, el Network Policy Management Library es diseñado en base a un lenguaje para robots, Yampa [CNP03] y librerías para programación web en Flapjax [MGB+09]. Las acciones usan un constructor tipo Rule que contiene un patrón o filtros y lista de acciones como argumentos. Las acciones principales son: enviar a un puerto determinado, enviar paquete al controlador, modificar la cabecera del paquete, y acción en blanco, que se interpreta como descartar el paquete. La instalación de estas políticas se realiza mediante la generación de policy events (similar a querys), primitive events (Seconds, SwitchJoin SwitchExit, PortChange) y listener (Print, Register). El resultado de experimentos [FHF+11] muestra que Frenetic proporciona simplicidad, así como un ahorro significativo en código y menor consumo de los recursos de la red en comparación a NOX.

Una de las ventajas adicionales de este lenguaje es la composición, es decir, se pueden escribir módulos funcionales independientes y el runtime system coordina su correcto funcionamiento en el controlador y en el switch. Existen 2 tipos de composición: secuencial y paralela. En la composición secuencial la salida de un módulo es la entrada del siguiente. Por ejemplo, un balanceador de carga que primeramente modifica el destino IP de un paquete y luego busca el puerto de salida en función de la nueva cabecera IP. En la composición paralela ambos módulos son ejecutados virtualmente al mismo tiempo en el controlador. Por ejemplo, si el balanceador envía un paquete con destino IP A hacia el puerto 1 y el paquete con destino IP B hacia el puerto 2. Esta composición resultaría en una función que envía los paquetes entrantes por los puertos 1 y 2.

McNettle [VW12] es un controlador diseñado especialmente para ofrecer alta escalabilidad a la red SDN. Esto se logra mediante un conjunto de manejadores de mensaje (message handlers), uno por cada switch, que tienen una función que gestiona las variables switch-local y network state y administra las acciones de suministro de los flujos de la red. El principio es que los mensajes de un switch individual se manejen secuencialmente,

mientras que los mensajes de switches diferentes sean manejados concurrentemente. De igual manera, McNettle intenta que cada mensaje sea procesado en un único core CPU. De esta manera, se reduce al máximo el número de conexiones y sincronizaciones inter-cores, mejorándose el rendimiento. Las pruebas realizadas en [VW12] muestran que McNettle tienen un desempeño multicore considerable en comparación a NOX o Beacon.

El controlador propuesto en [GRF13] se basa en la verificación de las políticas establecidas, en lugar de buscar bugs monitorizando el funcionamiento del controlador. Para realizar la verificación, en primer lugar se utiliza el lenguaje de alto nivel denominado Netcore [MFHW12], en donde se expresa únicamente el comportamiento de la red, más no su modo de implementación en el controlador. Luego, el NetCore Compiler expresa dichas políticas en configuraciones a nivel de switch (tablas de flujo). La información de las tablas de flujo es analizada nuevamente por el Verified Run-time System, que traduce dicha configuración en un nivel de abstracción más bajo denominado Featherweght OpenFlow. Featherweght OpenFlow es un modelo que permite asegurar que las reglas instaladas en el switch son consistentes con la tabla de flujo y que, gracias a primitivas de sincronización, aseguran que el funcionamiento del switch sea el correcto. Asimismo, en [RFR+12] se presenta la herramienta Kinetic, que ofrece mantener consistencia de actualizaciones en la red mediante dos mecanismos: de paquete y de flujo. En el primero de ellos se garantiza que cuando existe una actualización el paquete que circula en por la red se procesa por una misma configuración. En el segundo se asegura que todos los paquetes pertenecientes al mismo flujo (por ejemplo, la misma conexión TCP) sean tratados de forma similar por los switches de la red.

B.3. NOX/POX

NOX es uno de los primeros controladores OpenFlow de código abierto. En este apartado se agrupan NOX y POX debido a que tienen estructura similar, pero un entorno diferente de implementación (NOX es desarrollado en C++ y POX en Python).

NOX incluye una vista sobre la red en la que incluye la topología a nivel de switches, la localización de los usuarios, hosts, middleboxes, servicios (por ejemplo, *HyperText Transfer Protocol* (HTTP) y otros elementos de la red. La vista también incluye todos los enlaces entre nombres y direcciones. La interfaz de programación de NOX es simple, centrándose sobre eventos, espacio de nombres y vista sobre la red.

- Eventos: las redes en general no son estáticas, es decir, los flujos llegan y se van, igual que los usuarios y los enlaces. Para alcanzar este cambio de eventos, las aplicaciones de NOX usan un conjunto de handlers o manejadores de eventos que están registrados para ejecutarse cuando una interrupción particular sucede. Estos handlers son ejecutados en el orden de prioridades (especificado durante el registro del handler). El handler devuelve un valor indicado a NOX si hay que parar

la ejecución de ese evento, continuarlo o pasar dicho evento al siguiente handler registrado. Algunos eventos son generados directamente por los mensajes OpenFlow como switch join, switch leave, packet received y switch statistics received. Otros eventos son generados por las aplicaciones NOX como resultado de un procesamiento de eventos de bajo nivel. Por ejemplo, NOX incluye aplicaciones que autenticarán a un usuario a través de la redirección de tráfico HTTP con el evento packet received.

- Vista de la red y espacio de nombres: NOX incluye un número de aplicaciones base las cuales construyen la vista de la red y mantienen un espacio de nombres de alto nivel que puede ser usado por otras aplicaciones. Estas aplicaciones manejan la autenticación del usuario y del host para obtener los nombres de los hosts a través de una monitorización DNS. La vista de la red debe ser consistente y hacerla disponible para todas las instancias de controladores NOX, por lo que la escritura dirigida a ellos se torna compleja. Debido a esto, las aplicaciones NOX sólo deben avisar cuando se detecta un cambio en la red y no para todos los paquetes recibidos.

- Servicios de alto nivel: NOX incluye un sistema de librerías para proveer implementaciones eficientes de funciones comunes a muchas aplicaciones de la red. Éstas incluyen un módulo de encaminamiento, clasificación rápida de paquetes, servicios estándar como *Dynamic Host Control Protocol* (DHCP) y DNS, y un módulo de filtrado basado en políticas de la red.

B.4. Floodlight

Floodlight [Flo17] es un controlador OpenFlow desarrollado en Java. Aparece como evolución del controlador Beacon [Eri13], por lo que igualmente comparten similar estructura. Java brinda soporte multiplataforma y sencillez de programación. El usuario puede hacer uso de herramientas típicas de Java, como son threads, temporizadores, sockets, etc. Floodlight incluye la librería OpenFlowJ para trabajar con mensajes OpenFlow. Esta librería es una implementación Java orientada a objetos de la especificación de la versión 1.0 de OpenFlow. Los listeners se utilizan para notificar cuando los switches son añadidos o eliminados (IOFSwitchListener), y para recibir distintos tipos de mensajes específicos OpenFlow (IOFMessageListener). Además, Floodlight contiene aplicaciones básicas de referencia las cuales constituyen el core. Esta API adicional es la que se muestra en la Figura B.2.

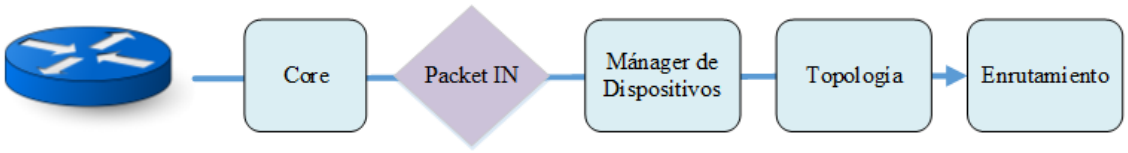


Figura B.2: Pipeline del Thread IOFMessageListener de Beacon

Las aplicaciones básicas disponibles son:

- Device Manager: muestra los dispositivos que aparecen en la red, incluyendo sus direcciones (Ethernet e IP), fecha de último uso, el switch y el puerto que han sido vistos por última vez. El Device Manager provee una interfaz (IDeviceManager) para buscar dispositivos conocidos y la habilidad de registrarlos para recibir eventos cuando nuevos dispositivos sean incluidos, actualizados o eliminados.
- Topology: descubre los enlaces entre los switches OpenFlow. Su interfaz (ITopology) permite la recuperación de una lista de los enlaces y el registro de eventos para ser notificados cuando los enlaces son incluidos o eliminados.
- Routing: provee enrutamiento de la capa L2 con el camino más corto entre dos dispositivos de la red. Esta aplicación exporta la interfaz IRoutingEngine, permitiendo implementaciones con diferentes lógicas de encaminamiento. La implementación incluida usa todos los métodos de computación de camino más corto entre dos pares. Esta aplicación depende tanto de la Topology como del Device Manager.
- Web: provee una Web User Interface (UI) para Floodlight. La aplicación web provee la interfaz IWebManageable, permitiendo a los desarrolladores añadir sus propios elementos UI.

Por otro lado, Floodlight contiene funciones adicionales como se muestra en la Figura B.3. Se presenta un Java API para el desarrollo de aplicaciones que residen dentro del controlador y requieren alta eficiencia de procesamiento (por ejemplo, procesamiento de paquetes tipo PACKET_IN). Adicionalmente, el REST API (por las siglas de Representational State Transfer) está disponible para configuración remota (puerto 8080 por defecto) de los diferentes servicios del controlador. De esta manera, los usuarios pueden crear aplicaciones que invoquen servicios del controlador mediante peticiones web (HTTP REST).

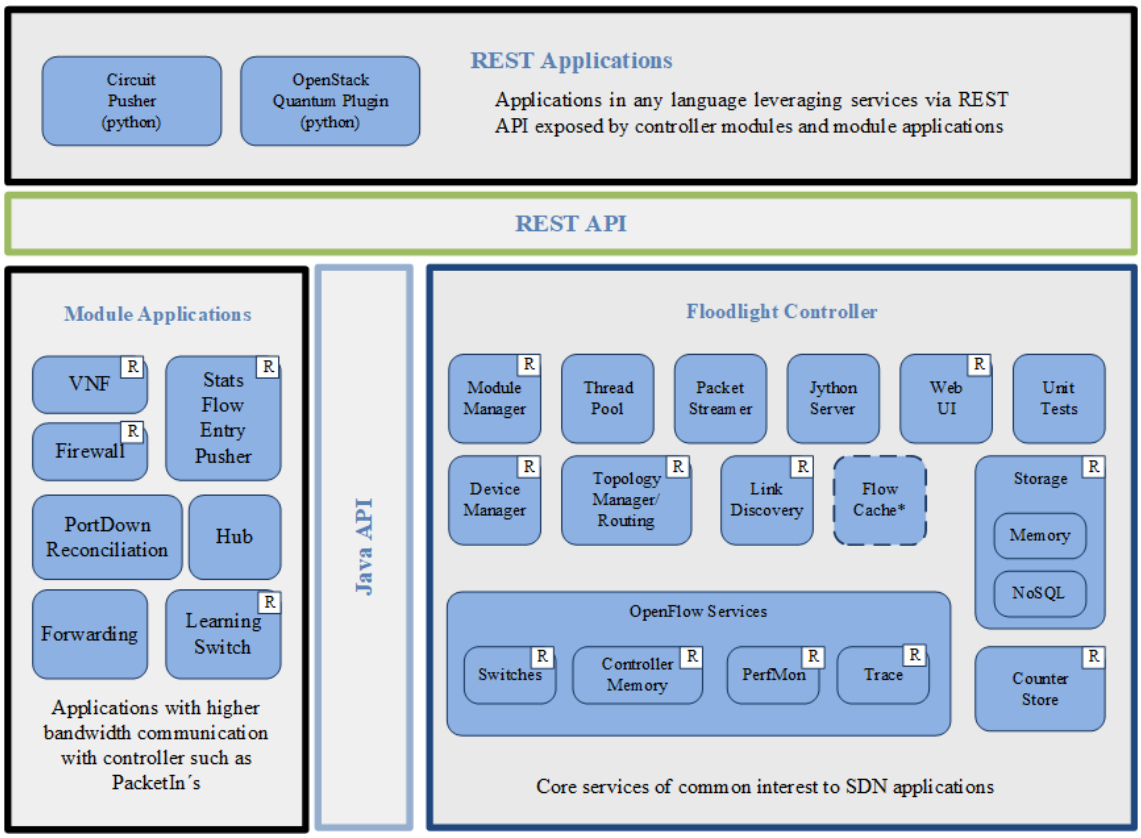


Figura B.3: API de Floodlight

B.4.1. Modularidad en Tiempo de Ejecución

Muchos de los controladores OpenFlow tienen la habilidad de seleccionar qué aplicaciones compilar (modularidad en tiempo de compilación) y qué aplicaciones ejecutar cuando el controlador comienza a actuar (modularidad en tiempo de inicio). Floodlight tiene la capacidad adicional de comenzar y finalizar aplicaciones mientras se está ejecutando, además de añadirlas y eliminarlas (modularidad en tiempo de ejecución) sin que el proceso de Floodlight se termine.

B.5. Pyretic

Pyretic [MRF+13] es uno de los miembros de la familia de lenguajes de programación SDN. Surge como resultado de uno de los lenguajes de programación de redes como es Frenetic [Fre17] pero con sintaxis de Python, ofreciendo una manera simple para expresar políticas de alto nivel, las cuales compilan las reglas de coincidencia OpenFlow. Permite a los programadores especificar políticas en lo que son paquetes localizados. Una política está basada en la combinación de un paquete y su localización en la red.

B.5.1. Características

Pyretic ofrece muchas características entre las cuales se encuentran la habilidad de escribir políticas de red como funciones. En otras palabras, Pyretic permite al programador escribir una función que recoge como entrada un paquete, devolviéndolo en diferentes localizaciones en la red. Pyretic provee predicados booleanos en contraste a otros controladores. Además, a diferencia de manejar reglas de acciones por medio de coincidencias, Pyretic permite la creación de políticas usando conjunciones y predicados como and y not. En la Tabla B.2 se muestra una lista de algunas políticas del lenguaje Pyretic.

Tabla B.2: Políticas Atómicas del Lenguaje Pyretic

Sintaxis	**Descripción**
None	Devuelve el conjunto vacío.
Identidad	Devuelve el paquete original.
Match	Devuelve la identidad si el campo del paquete coincide con un valor particular, en otro caso, devuelve el conjunto vacío.
Mod	Devuelve el mismo paquete pero cambia el campo de la cabecera virtual del paquete a un valor específico o envía el tráfico a través de un puerto de salida particular modificando solamente el atributo que se encarga de dicho puerto en el paquete.
Flood	Devuelve un paquete por cada puerto en el árbol de recubrimiento de la red.

Por otro lado, Pyretic ofrece registros o campos virtuales en la cabecera del paquete, que habilitan al programador a referirse tanto a las cabeceras actuales como a las virtuales.

Por último, Pyretic provee operadores de composición tanto paralelos como secuenciales para hacer posible que el operador de red escriba políticas complejas compuestas por otras más sencillas.

Bibliografía

[3gs00] 3G Specification. https://www.3gpp.org, 2000.

[AAKS98] D. Alexander, W. Arbaugh, A. Keromytis, and J. Smith. A Secure Active Network Environment Architecture: Realization in SwitchWare. *IEEE Network*, 12(3):37–45, May 1998.

[AATF19] M. Abujubbeh, F. Al-Turjman, and M. Fahrioglu. Software-defined wireless sensor networks in smart grids: An overview. *Sustainable Cities and Society*, 51:101754, November 2019.

[AATK13] J. I. Agbinya, M. C. Aguayo-Torres, and R. Klempous. *4G wireless communication networks: Design Planning and Applications*. River Publishers, 2013.

[ABF+13] B. Anwer, T. Benson, N. Feamster, D. Levin, and J. Rexford. A slick control plane for network middleboxes. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 147–148, Hong Kong, China, August 2013.

[ACDF10] G. Anastasi, M. Conti, and M. Di Francesco. A comprehensive analysis of the MAC unreliability problem in IEEE 802.15. 4 wireless sensor networks. *IEEE Transactions on Industrial Informatics*, 7(1):52–65, October 2010.

[AEH19] Y. Al-Eryani and E. Hossain. The D-OMA Method for Massive Multiple Access in 6G: Performance, Security, and Challenges. *IEEE Vehicular Technology Magazine*, 14(3):92–99, July 2019.

[ANLC16] I. F. Akyildiz, S. Nie, S. Lin, and M. Chandrasekaran. 5G roadmap: 10 key enabling technologies. *Computer Networks*, 106:17–48, September 2016.

[ARS16] M. Agiwal, A. Roy, and N. Saxena. Next generation 5G wireless networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 18(3):1617–1655, February 2016.

[AT19] F. Al-Turjman. A novel approach for drones positioning in mission critical applications. *Transactions on Emerging Telecommunications Technologies*, page e3603, April 2019.

[ATAMM20] F. Al-Turjman, M. Abujubbeh, A. Malekloo, and L. Mostarda. UAVs assessment in software-defined IoT networks: An overview. *Computer Communications*, 150:519–536, January 2020.

[AWL15] I. F. Akyildiz, P. Wang, and S. Lin. SoftAir: A software defined networking architecture for 5G wireless systems. *Computer Networks*, 85(5):1–18, July 2015.

[BCS13] K. Benton, L. J. Camp, and C. Small. OpenFlow vulnerability assessment. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 151–152, Hong Kong, China, August 2013.

[BDC18] S. Biswas, R. Das, and P. Chatterjee. Energy-efficient connected target coverage in multi-hop wireless sensor networks. In *Industry interactive innovations in science, engineering and technology*, pages 411–421. Springer, 2018.

[BdSMP10] R. Braga, E. de Souza Mota, and A. Passito. Lightweight DDoS flooding attack detection using NOX/OpenFlow. In *LCN*, pages 408–415, October 2010.

[BGH+14] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O Connor, P. Radoslavov, and W. Snow. ONOS: Towards an Open, Distributed SDN OS. In *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, pages 1–6, August 2014.

[BPC+07] P. Baronti, P. Pillai, V. WC. Chook, S. Chessa, A. Gotta, and Y. F. Hu. Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and ZigBee standards. *Computer communications*, 30(7):1655–1695, May 2007.

[BZZ+14] R. Beckett, X. K. Zou, S. Zhang, S. Malik, J. Rexford, and D. Walker. An assertion language for debugging SDN applications. In *Proceedings of the third workshop on Hot topics in software defined networking*, pages 91–96, August 2014.

[Cal99] K. Calvert. Architectural framework for active networks version 1.0. *Active Network Working Group Draft*, 158, 1999.

[CBAB14] S. R. Chowdhury, M. F. Bari, R. Ahmed, and R. Boutaba. PayLess: A Low Cost Netowrk Monitoring Framework for Software Defined Networks. In *Proceedings of the IEEE/IFIP Network Operations and Management Symposium*, pages 1–9, Krakow,Poland, May 2014.

[CCF+05] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and J. Van der Merwe. Design and Implementation of a Routing Control Platform. In *Proceedings of the 2nd Conference on Symposium on Networked Systems Design and Implementation*, pages 15–28, Berkeley, CA, USA, May 2005.

[CEQM+04] M. Castillo-Effer, D. H Quintela, W. Moreno, R. Jordan, and W. Westhoff. Wireless sensor networks for flash-flood alerting. In *Proceedings of the Fifth IEEE International Caracas Conference on Devices, Circuits and Systems, 2004.*, pages 142–146, Punta Cana, Dominican Republic, November 2004.

[CFP+07] M. Casado, M. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker. Ethane: Taking Control of the Enterprise. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 1–12, Stockholm, Sweden, August 2007.

[CFSD90] J.D. Case, M. Fedor, M.L. Schoffstall, and J. Davin. RFC 1157 - Simple Network Management Protocol (SNMP). https://tools.ietf.org/html/rfc1157, May 1990.

[CGA+06] M. Casado, T. Garfinkel, A. Akella, M. J. Freedman, D. Boneh, N. McKeown, and S. Shenker. SANE: A Protection Architecture for Enterprise Networks. In *USENIX Security Symposium*, page 50, Vancouver, B.C., Canada, August 2006.

[CGMP12] S. Costanzo, L. Galluccio, G. Morabito, and S. Palazzo. Software defined wireless networks (sdwn): Unbridling sdns. In *European workshop on software defined networking*, pages 1–6, Darmstadt, Germany, December 2012.

[Cis11] T. Cisco. Cisco visual networking index: global mobile data traffic forecast update, 2010–2015. White Paper, Technical Report, Cisco Systems Inc. 2011., February 2011.

[Cis17] Cisco. The Zettabyte Era: Trends and Analysis. Technical Report, Cisco Systems, June 2017.

[CK03] C. Chong and S. P. Kumar. Sensor networks: evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, August 2003.

[CKY+14] T. Choi, S. Kang, S. Yoon, S. Yang, S. Song, and H. Park. SuVMF: Software-defined unified virtual monitoring function for SDN-based large-scale networks. In *Proceedings of The Ninth International Conference on Future Internet Technologies*, page 4, Tokyo Japan, June 2014.

[Cla04] B. Claise. RFC 3954 - Cisco Systems NetFlow Services Export Version 9. https://tools.ietf.org/html/rfc3954, October 2004.

[CNP03] A. Courtney, H. Nilsson, and J. Peterson. The Yampa Arcade. In *Proceedings of the ACM Workshop on Haskell*, pages 7–18, Uppsala Sweden, August 2003.

[CSSK+15] R. Chávez-Santiago, M. Szydełko, A. Kliks, F. Foukalas, Y. Haddad, K. E. Nolan, M. Y. Kelly, M. T. Masonta, and I. Balasingham. 5G: The convergence of wireless communications. *Wireless Personal Communications*, 83(3):1617–1642, March 2015.

[CZCT14] T. Chen, H. Zhang, X. Chen, and O. Tirkkonen. SoftMobile: Control evolution for future heterogeneous mobile networks. *IEEE Wireless Communications*, 21(6):70–78, 2014.

[DB18] K. David and H. Berndt. 6G vision and requirements: Is there any need for beyond 5G? *ieee vehicular technology magazine*, 13(3):72–80, September 2018.

[DGAM14] A. De Gante, M. Aslan, and A. Matrawy. Smart wireless sensor network management based on software-defined networking. In *2014 27th Biennial Symposium on Communications (QBSC)*, pages 71–75, Kingston, ON, Canada, June 2014.

[DH98] S. Deering and R. Hinden. RFC 2460 - Internet protocol, version 6 (IPv6) specification. https://tools.ietf.org/html/rfc2460, 1998.

[DMW+11] A. Damnjanovic, J. Montojo, Y. Wei, T. Ji, T. Luo, M. Vajapeyam, T. Yoo, O. Song, and D. Malladi. A survey on 3GPP heterogeneous networks. *IEEE Wireless communications*, 18(3):10–21, June 2011.

[DS07] C. Douligeris and D. N. Serpanos. *Network security: current status and future directions*. John Wiley & Sons, February 2007.

[DT13] K. Dhamecha and B. Trivedi. Sdn issues-a survey. *International Journal of Computer Applications*, 73(18):30–35, July 2013.

[EBSB11] R. Enns, M. Bjorklund, J. Schoenwaelder, and A. Bierman. RFC 6241 - Network Configuration Protocol (NETCONF). https://tools.ietf.org/html/rfc6241, June 2011.

[EDBT12] H.E. Egilmez, S.T. Dane, K.T. Bagci, and A.M. Tekalp. OpenQoS: An OpenFlow Controller Design for Multimedia Delivery with End-to-end Quality of Service over Software-Defined Networks. In *Proceedings of the Asia-Pacific Signal Information Processing Association Annual Summit and Conference*, pages 1–8, Hollywood, CA, USA, December 2012.

[Eri11] L. Ericsson. More than 50 billion connected devices. *White Paper*, 14(1):124, 2011.

[Eri13] D. Erickson. The Beacon Openflow Controller. In *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, pages 13–18, New York, NY, USA, August 2013.

[Eri15] Ericcson. Ericsson Mobility Report. https://www.ericsson.com/assets/local/mobility-report/documents/2015/ericsson-mobility-report-june-2015.pdf, June 2015.

[FBMP12] P. Fonseca, R. Bennesby, E. Mota, and A. Passito. A Replication Component for Resilient OpenFlow-based Networking. In *Proceedings of the IEEE Network Operations and Management Symposium*, pages 933–939, Maui, HI, USA, April 2012.

[Fer13] M. P. Fernandez. Comparing openflow controller paradigms scalability: Reactive and proactive. In *2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)*, pages 1009–1016, Barcelona, Spain, June 2013.

[FGR+13] N. Foster, A. Guha, M. Reitblatt, A. Story, M. Freedman, N. Katta, C. Monsanto, J. Reich, J. Rexford, C. Schlesinger, D. Walker, and R. Harrison. Languages for Software Defined Networks. *IEEE Communications Magazine*, 51(2):128–134, February 2013.

[FHF+11] N. Foster, R. Harrison, M. Freedman, C. Monsanto, J. Rexford, A. Story, and D. Walker. Frenetic: A Network Programming Language. In *Proceedings of the 16th ACM International Conference on Functional Programming*, volume 46, pages 279–291, Tokyo Japan, September 2011.

[FHTG10] M. Fiedler, T. Hossfeld, and P. Tran-Gia. A generic quantitative relationship between quality of experience and quality of service. *IEEE Network*, 24(2):36–41, March 2010.

[Flo17] Floodlight. http://www.projectfloodlight.org/, April 2017.

[Fre17] Frenetic. https://github.com/frenetic-lang/frenetic, April 2017.

[FRZ14] N. Feamster, J. Rexford, and E. Zegura. The Road to SDN: An Intellectual History of Programmable Networks. *ACM SIGCOMM Computer Communication Review*, 44(2):87–98, December 2014.

[FSYM13] Seyed Kaveh Fayazbakhsh, Vyas Sekar, Minlan Yu, and Jeffrey C Mogul. Flowtags: Enforcing network-wide policies in the presence of dynamic middlebox actions. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 19–24, Hong Kong China, August 2013.

[GBMP13] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami. Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions. *Future Generation Computer Systems - The International Journal of Grid Computing and Science*, 29(7):1645–1660, September 2013.

[GEB+13] P. Georgopoulos, Y. Elkhatib, M. Broadbent, M. Mu, and N. Race. Towards Network-wide QoE Fairness Using Openflow-assisted Adaptive Video Streaming. In *Proceedings of the 2013 ACM SIGCOMM Workshop on Future Human-centric Multimedia Networking*, pages 15–20, Hong Kong, China, August 2013.

[GGW+06] T. Gao, D. Greenspan, M. Welsh, R. R. Juang, and A. Alm. Vital signs monitoring and patient tracking over a wireless network. In *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, pages 102–105, Shanghai, China, January 2006.

[GHM+05a] A. Greenberg, G. Hjaimtysson, D. Maltz, A. Myers, J. Rexford, G. Xie, H. Yan, J. Zhan, and H. Zhang. A Clean Slate 4D Approach to Network Control and Management. *ACM SIGCOMM Computer Communication Review*, 35(5):41–54, October 2005.

[GHM+05b] A. Greenberg, G. Hjalmtysson, D. A Maltz, A. Myers, J. Rexford, G. Xie, H. Yan, J. Zhan, and H. Zhang. A clean slate 4D approach to network control and management. *ACM SIGCOMM Computer Communication Review*, 35(5):41–54, October 2005.

[GJSB00] J. Gosling, B. Joy, G. Steele, and G. Bracha. *The Java language specification*. Addison-Wesley Professional, April 2000.

[GKP+08] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker. NOX: Towards an Operating System for Networks. *ACM SIGCOMM Computer Communication Review*, 38(3):105–110, July 2008.

[GMMP15a] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo. Reprogramming Wireless Sensor Networks by using SDN-WISE: A hands-on demo. In *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pages 19–20, Hong Kong, China, April 2015.

[GMMP15b] L. Galluccio, S. Milardo, G Morabito, and S. Palazzo. SDN-WISE: Design, prototyping and experimentation of a stateful SDN solution for WIreless SEnsor networks. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 513–521, Kowloon, Hong Kong, April 2015.

[GNBC10] A. Goodney, S. Narayan, V. Bhandwalkar, and Y. H. Cho. Pattern based packet filtering using NetFPGA in DETER infrastructure. In *1st Asia NetFPGA developers workshop. Daejeon, Korea*, Daejeon, Korea, June 2010.

[GPLK13] A. Gudipati, . Perry, L. E. Li, and S. Katti. SoftRAN: Software defined radio access network. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 25–30, Hong Kong China, August 2013.

[GRF13] A. Guha, M. Reitblatt, and N. Foster. Machine-verified network controllers. In *ACM SIGPLAN Notices*, pages 483–494, New York, NY, United States, June 2013.

[GVSOTCBA09] L. J. García Villalba, A. Sandoval Orozco, A. Trivino Cabrera, and C. Barenco Abbas. Routing protocols in wireless sensor networks. *sensors*, 9(11):8399–8421, October 2009.

[HSM12] B. Heller, R. Sherwood, and N. McKeown. The Controller Placement Problem. In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, pages 7–12, New York, NY, USA, August 2012.

[HW79] J. A Hartigan and M. A. Wong. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[HWZ13] S. Hartman, M. Wasserman, and D. Zhang. Security requirements in the software defined networking model. *Internet Engineering Task Force, Internet-Draft draft-hartman-sdnsec-requirements-01*, pages 1–11, April 2013.

[IoT17] European Lighthouse Integrated Project, Internet of Things Architecture. http://www.iot-a.eu/public/, April 2017.

[JASD12] J. H. Jafarian, E. Al-Shaer, and Q. Duan. Openflow random host mutation: transparent moving target defense using software defined networking. In *Proceedings of the first workshop on Hot topics in software defined networks*, pages 127–132, Helsinki Finland, August 2012. ACM.

[JLVR13] X. Jin, E. Li, L. Vanbever, and J. Rexford. Softcell: Scalable and flexible cellular core network architecture. In *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*, pages 163–174, Santa Barbara California USA, December 2013.

[KAMH17] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke. A survey on software-defined wireless sensor networks: Challenges and design requirements. *IEEE access*, 5:1872–1899, February 2017.

[KAMH18] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke. Fragmentation-based distributed control system for software-defined wireless sensor networks. *IEEE transactions on industrial informatics*, 15(2):901–910, April 2018.

[KF13] H. Kim and N. Feamster. Improving Network Management with Software Defined Networking. *IEEE Communications Magazine*, 51(2):114–119, February 2013.

[KKSZ11] H. Kriegel, P. Kröger, J. Sander, and A. Zimek. Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231–240, April 2011.

[KMN+02] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 24(7):881–892, August 2002.

[Koh90] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, September 1990.

[KRV13] D. Kreutz, F. Ramos, and P. Verissimo. Towards secure and dependable software-defined networks. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 55–60, Hong Kong China, August 2013.

[KSC+11] H. Kim, S. Sundaresan, M. Chetty, N. Feamster, and W. Edwards. Communicating with Caps: Managing Usage Caps in Home Networks. In *Proceedings of the ACM SIGCOMM 2011 Conference*, pages 470–471, Toronto, ON, Canada, August 2011.

[KSKD+12] A. Kassler, L. Skorin-Kapov, O. Dobrijevic, M. Matijasevic, and P. Dely. Towards QoE-driven Multimedia Service Negotiation and Path Optimization with Software Defined Networking. In *Proceedings of the 20th International Conference on Software, Telecommunications and Computer Networks*, pages 1–5, Split, Croatia, September 2012.

[KW03] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: Attacks and countermeasures. *Ad hoc networks*, 1(2-3):293–315, September 2003.

[LCM03] Z. Liu, R. H. Campbell, and M D. Mickunas. Active security support for active networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 33(4):432–445, November 2003.

[LCMP12a] P. Le Callet, S. Möller, and A. Perkis. Qualinet White Paper on Definitions of Quality of Experience. White Paper, European Network on Quality of Experience in Multimedia Systems and Services, March 2012.

[LCMP12b] P. Le Callet, S. Möller, and A. Perkis. Qualinet White Paper on Definitions of Quality of Experience. *European Network on Quality of Experience in Multimedia Systems and Services*, 3:1–24, March 2012.

[LDXZ18] S. Li, L. Da Xu, and S. Zhao. 5G Internet of Things: A survey. *Journal of Industrial Information Integration*, 10:1–9, June 2018.

[LFZ18] B. Li, Z. Fei, and Y. Zhang. UAV communications for 5G and beyond: Recent advances and future trends. *IEEE Internet of Things Journal*, 6(2):2241–2263, December 2018.

[Li18] Z. Li. Towards the next generation of multi-criteria recommender systems. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 553–557, Vancouver British Columbia Canada, October 2018.

[LLM10] Taeyoung Lee, Melvin Leok, and N Harris McClamroch. Geometric tracking control of a quadrotor UAV on SE (3). In *49th IEEE conference on decision and control (CDC)*, pages 5420–5425, Atlanta, GA, USA, December 2010.

[LMFJ+04] K. Lorincz, D. J. Malan, T. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton. Sensor networks for emergency response: challenges and opportunities. *IEEE pervasive Computing*, 3(4):16–23, December 2004.

[LMR12] L. E. Li, Z. M. Mao, and J. Rexford. Toward software-defined cellular networks. In *2012 European Workshop on Software Defined Networking*, pages 7–12, Darmstadt, Germany, October 2012. IEEE.

[LMZ+16] S. Lin, F. Miao, J. Zhang, G. Zhou, L. Gu, T. He, J. A. Stankovic, S. Son, and G. J. Pappas. Atpc: adaptive transmission power control for wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 12(1):1–31, March 2016.

[LNR+04] T. Lakshman, T. Nandagopal, R. Ramjee, K. Sabnani, and T. Woo. The SoftRouter Architecture. In *Proceedings of the ACM SIGCOMM Workshop on Hot Topics in Networking*, pages 1–6, New York, NY, USA, November 2004.

[LTQ12] T. Luo, H. Tan, and T. Quek. Sensor OpenFlow: Enabling software-defined wireless sensor networks. *IEEE Communications letters*, 16(11):1896–1899, November 2012.

[MA15] R. N. Mitra and D. P. Agrawal. 5g mobile technology: A survey. *ICT Express*, 1(3):132–137, December 2015.

[MAB+08] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. OpenFlow: Enabling Innovation in Campus Networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, March 2008.

[MFHW12] C. Monsanto, N. Foster, R. Harrison, and D. Walker. A Compiler and Run-time System for Network Programming Languages. *ACM SIGPLAN NOTICES*, 47(1):217–230, January 2012.

[MGB+09] L. Meyerovich, A. Guha, J. Baskin, G. Cooper, M. Greenberg, and A. Bromfield. Flapjax: A Programming Language for Ajax Applications. In *Proceedings of the 24th ACM Conference on Object Oriented Programming Systems Languages and Applications*, pages 1–20, Orlando Florida USA, October 2009.

[min17] Mininet. https://github.com/mininet/mininet/wiki/Introduction-to-Mininet#limits, April 2017.

[MLMAM18] K. M. Modieginyane, B. B. Letswamotse, R. Malekian, and A. M. Abu-Mahfouz. Software defined wireless sensor networks application opportunities for efficient network management: A survey. *Computers & Electrical Engineering*, 66:274–287, February 2018.

[MLP+01] S. Murphy, E. Lewis, R. Puga, R. Watson, and R. Yee. Strong security for active networks. In *2001 IEEE Open Architectures and Network Programming Proceedings. OPENARCH 2001 (Cat. No. 01EX484)*, pages 63–70, Anchorage, AK, USA, USA, April 2001.

[MM18] H. Mostafaei and M. Menth. Software-defined wireless sensor networks: A survey. *Journal of Network and Computer Applications*, 119:42–56, October 2018.

[Moo96] T. K. Moon. The expectation-maximization algorithm. *IEEE Signal processing magazine*, 13(6):47–60, November 1996.

[MR11] A. Mahmud and R. Rahmani. Exploitation of OpenFlow in wireless sensor networks. In *Proceedings of 2011 International Conference on Computer Science and Network Technology*, pages 594–600, Harbin, China, December 2011.

[MRF+13] C. Monsanto, J. Reich, N. Foster, J. Rexford, and D. Walker. Composing Software Defined Networks. In *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, pages 1–14, Lombard, IL, USA, April 2013.

[MVTG14] J. Medved, R. Varga, A. Tkacik, and K. Gray. Opendaylight: Towards a model-driven sdn controller architecture. In *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, pages 1–6, Sydney, NSW, Australia, October 2014.

[Mye99] A. C. Myers. JFlow: Practical Mostly-static Information Flow Control. In *Proceedings of the 26th ACM Symposium on Principles of Programming Languages*, pages 228–241, San Antonio, Texas, USA, January 1999.

[MZD+15] Z. Ma, Z. Zhang, Z. Ding, P. Fan, and H. Li. Key techniques for 5G wireless communications: network architecture, physical layer, and MAC layer perspectives. *Science China information sciences*, 58(4):1–20, February 2015.

[NCC10] E. Ng, Z. Cai, and A.L. Cox. Maestro: A System for Scalable OpenFlow Control. Technical Report, Rice University, December 2010.

[NHAM17] M. Ndiaye, G. Hancke, and A. Abu-Mahfouz. Software defined networking for improved wireless sensor network management: A survey. *Sensors*, 17(5):1031, May 2017.

[NJR+12] U. Niethammer, M. James, S. Rothmund, J. Travelletti, and M. Joswig. UAV-based remote sensing of the Super-Sauze landslide: Evaluation and results. *Engineering Geology*, 128:2–11, March 2012.

[NR14] F. Nex and F. Remondino. UAV for 3D mapping applications: a review. *Applied geomatics*, 6(1):1–15, March 2014.

[NRFC09] A. Nayak, A. Reimers, N. Feamster, and R. Clark. Resonance: Dynamic Access Control for Enterprise Networks. In *Proceedings of the 1st ACM Workshop on Research on Enterprise Networking*, pages 11–18, Barcelona Spain, August 2009.

[NS306] NS3—Network Simulator. https://www.nsnam.org/, July 2006.

[NSSG12] F. Németh, A. Stipkovits, B. Sonkoly, and A. Gulyás. Towards smartflow: case studies on enhanced programmable forwarding in openflow switches. *ACM SIGCOMM Computer Communication Review*, 42(4):85–86, August 2012.

[OLAL+17] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J Ramos-Munoz, J. Lorca, and J. Folgueira. Network slicing for 5G with SDN/NFV: Concepts, architectures, and challenges. *IEEE Communications Magazine*, 55(5):80–87, May 2017.

[ONF17] ONF. Open Networking Foundation. https://www.opennetworking.org/, April 2017.

[Ope10] OpenFlow Switch Specification v.1.0.0. http://yuba.stanford.edu/~derickso/openflow-spec-v1.0.0-cookieenhancements.pdf, September 2010.

[Ope13] Open Networking Foundation. OpenFlow Management and Configuration Protocol (OF-Config) v.1.1.1. https://www.opennetworking.org/wp-content/uploads/2013/02/of-config-1-1-1.pdf, March 2013.

[P+81] J. Postel et al. RFC 791 - Internet protocol. https://tools.ietf.org/html/rfc791, September 1981.

[PCF+15] P. A. Porras, S. Cheung, M. W Fong, K. Skinner, and V. Yegneswaran. Securing the Software Defined Network Control Layer. In *Network and Distributed System Security (NDSS) Symposium*, San Diego, California, USA, February 2015.

[PFC+10] P. S. Pisa, N. C. Fernandes, H. E. Carvalho, M. D. Moreira, M. E. M. Campista, L. H. M. Costa, and O. C. M. Duarte. OpenFlow and Xen-Based Virtual Network Migration. *Communications: Wireless in Developing Countries and Networks of the Future*, 327:170–181, September 2010.

[PFGV17a] J. A. Puente Fernández and L. J García Villalba. Aplicación de Técnicas de Agrupamiento en la Monitorización de Redes Definidas por Software. In *XXXII Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2017)*, Cartagena, Murcia, España, September 2017.

[PFGV17b] J. A. Puente Fernández and L. J García Villalba. Flow Conservation Framework for Monitoring Software Defined Networks. In *19th International Conference on Information Technology (ICIT 2017)*, Paris, France, May 2017.

[PFGV20] J. A. Puente Fernández and L. J García Villalba. Análisis y Monitorización por Agrupamiento de Contenido Multimedia en Redes Definidas por Software. In *X Congreso Iberoamericano de Seguridad Informática (CIBSI 2020)*, Bogotá, Colombia, January 2020.

[PFGVK18a] J. A. Puente Fernández, L. J. García Villalba, and T. Kim. Clustering and Flow Conservation Monitoring Tool for Software Defined Networks. *Sensors*, 18(4):1079, April 2018.

[PFGVK18b] J. A. Puente Fernández, L. J. García Villalba, and T. Kim. Software defined networks in wireless sensor architectures. *Entropy*, 20(4):225, March 2018.

[PFVCGV16] J. A. Puente Fernández, A. L. Valdivieso Caraguay, and L. J García Villalba. Seguridad en Redes Definidas por Software: Desafíos y Soluciones. In *XIV Reunión Española sobre Criptología y Seguridad de la Información (RECSI 2016)*, Maó, Menorca, Illes Balears, España, October 2016.

[PL04] P. Phaal and M. Lavine. Sflow version 5. https://sflow.org/sflow_version_5.txt, July 2004.

[Pla17] PlanetLab. https://www.planet-lab.org/, April 2017.

[PLHea11] B. Pfaff, B. Lantz, B Heller, and et. all. OpenFlow Switch Specification v.1.1.0. Specification, Open Networking Foundation, February 2011.

[PPA+09] B. Pfaff, J. Pettit, K. Amidon, M. Casado, T. Koponen, and S. Shenker. Extending Networking into the Virtualization Layer. In *Proceedings of the ACM SIGCOMM HotNets*, pages 1–6, New York, NY, USA, October 2009.

[PSW04] A. Perrig, J. Stankovic, and D. Wagner. Security in wireless sensor networks. *Communications of the ACM*, 47(6):53–57, June 2004.

[PSY+12] P. Porras, S. Shin, V. Yegneswaran, M. Fong, M. Tyson, and G. Gu. A security enforcement kernel for OpenFlow networks. In *Proceedings of the first workshop on Hot topics in software defined networks*, pages 121–126, Helsinki Finland, August 2012.

[PWH13] K. Pentikousis, Y. Wang, and W. Hu. MobileFlow: Toward Software-Defined Mobile Networks. *IEEE Communications Magazine*, 51(7):44–53, July 2013.

[QoS] QoS vs. QoE. http://www.witbe.net/technologie/qos-vs-qoe/.

[RFR+12] M. Reitblatt, N. Foster, J. Rexford, C. Schlesinger, and D. Walker. Abstractions for network update. *ACM SIGCOMM Computer Communication Review*, 42(4):323–334, August 2012.

[Riv] Riverbed. OPNET Network Simulator. https://www.riverbed.com/mx/products/steelcentral/opnet.html.

[RMTF09] A. Ramachandran, Y. Mundada, M. B. Tariq, and N. Feamster. Securing Enterprise Networks Using Traffic Tainting. Technical Report, Georgia Institute of Technology, August 2009.

[RR16] B. Rashid and M. H. Rehmani. Applications of wireless sensor networks for urban areas: A survey. *Journal of network and computer applications*, 60:192–219, January 2016.

[RSC14] D. Raumer, L. Schwaighofer, and G. Carle. MonSamp: A Distributed SDN Application for QoS Monitoring. In *Proceedings of the Federated Conference on Computer Science and Information Systems*, pages 1–9, Warsaw, Poland, September 2014.

[SAA+16] J. P. Santos, R. Alheiro, L. Andrade, A. L. Valdivieso Caraguay, L. I. Barona Lopez, M. A. Sotelo Monge, L. J. Garcia Villalba, W. Jiang, H. Schotten, J. M. Alcaraz-Calero, et al. SELFNET Framework self-healing capabilities for 5G mobile networks. *Transactions on Emerging Telecommunications Technologies*, 27(9):1225–1232, June 2016.

[SBB13] R. Skowyra, S. Bahargam, and A. Bestavros. Software-defined ids for securing embedded mobile devices. In *2013 IEEE High Performance Extreme Computing Conference (HPEC)*, pages 1–7, Waltham, MA, USA, September 2013.

[SBC19] W. Saad, M. Bennis, and M. Chen. A vision of 6G wireless systems: Applications, trends, technologies, and open research problems. *arXiv preprint arXiv:1902.10265*, pages 1–9, October 2019.

[SEN17] SENSEI, Integrated EU Project. http://www.ict-sensei.org/, April 2017.

[SG13] S. W. Shin and G. Gu. Attacking software-defined networks: A first feasibility study. In *ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 165–166, Hong Kong China, August 2013.

[SGY+09] R. Sherwood, G. Gibb, K.K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar. Flowvisor: A Network Virtualization Layer. Technical Report, OpenFlow Switch Consortium, October 2009.

[SGY+10] R. Sherwood, G. Gibb, K. Yap, G. Appenzeller, M. Casado, N. McKeown, and G. Parulkar. Can the Production Network be the Testbed? In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, pages 1–6, Vancouver, BC, Canada, October 2010.

[SPY+13] S. W. Shin, P. Porras, V. Yegneswara, M. Fong, G. Gu, and M. Tyson. Fresco: Modular composable security services for software-defined networks. In *20th Annual Network & Distributed System Security Symposium*, San Diego, CA United States, February 2013.

[SSC+12] S. Sharma, D. Staessens, D. Colle, M. Pickavet, and P. Demeester. A Demonstration of Fast Failure Recovery in Software Defined Networking. In *Proceedings of the International Conference on Testbeds and Research Infrastructures*, pages 411–414, Thessanoliki, Greece, June 2012.

[SSHC+13] S. Sezer, S. Scott-Hayward, P. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, and N. Rao. Are We Ready for SDN? Implementation Challenges for Software-Defined Networks. *IEEE Communications Magazine*, 51(7):36–43, July 2013.

[STG18] M. Seufert and P. Tran-Gia. Quality of experience and access network traffic management of HTTP adaptive video streaming. In *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*, pages 1–8, Taipei, Taiwan, July 2018.

[STH14] M. Shibuya, A. Tachibana, and T. Hasegawa. Efficient Performance Diagnosis in OpenFlow Networks Based on Active Measurements. In *Proceedings of the Thirteenth International Conference on Networks*, pages 268–273, Nice, France, February 2014.

[SVBR18] J. Suárez-Varela and P. Barlet-Ros. SBAR: SDN flow-Based monitoring and Application Recognition. In *Proceedings of the Symposium on SDN Research*, page 22, Los Angeles CA USA, March 2018.

[SWXH14] Z. Su, T. Wang, Y. Xia, and M. Hamdi. FlowCover: Low-cost flow monitoring scheme in software defined networks. In *2014 IEEE Global Communications Conference*, pages 1956–1961, Austin, TX, USA, December 2014.

[SZ16] F. K. Shaikh and S. Zeadally. Energy harvesting in wireless sensor networks: A comprehensive review. *Renewable and Sustainable Energy Reviews*, 55:1041–1054, March 2016.

[SZMM13] L. Shi, B. Zhang, H. T. Mouftah, and J. Ma. DDRP: An Efficient Data-driven Routing Protocol for Wireless Sensor Networks with Mobile Sinks. *International Journal of Communication Systems*, 26(10):1341–1355, October 2013.

[Tet97] Tetcos. NetSim—Network Simulator. https://tetcos.com/, 1997.

[TGG10] A. Tootoonchian, M. Ghobadi, and Y. Ganjali. OpenTM: Traffic Matrix Estimator for OpenFlow Networks. In *Proceedings of the International Conference on Passive and Active Network Measurement*, pages 201–210, Zurich, Switzerland, April 2010.

[vADK14] N. L. van Adrichem, C. Doerr, and F. A. Kuipers. OpenNetMon: Network Monitoring in OpenFlow Software-Defined Networks. In *Proceedings of the Network Operations and Management Symposium*, pages 1–8, Krakow,Poland, May 2014.

[VCBLGV13] A. L. Valdivieso Caraguay, L. I. Barona López, and L. J. García Villalba. Evolution and Challenges of Software Defined Networking. In *Proceedings of the Workshop on Software Defined Networks for Future Networks and Services*, pages 61–67, Trento, Italy, November 2013.

[VCPFGV15] A. L. Valdivieso Caraguay, J. A. Puente Fernández, and L. J. García Villalba. Framework for Optimized Multimedia Routing over Software Defined Networks. *Computer Networks*, 92(2):369–379, December 2015.

[VCPFGV17] A. L. Valdivieso Caraguay, J. A. Puente Fernández, and L. J. García Villalba. An Optimization Framework for Monitoring of SDN/OpenFlow Networks. *International Journal of Ad Hoc and Ubiquitous Computing*, 26(4):263–273, November 2017.

[VGS+13] P. Vlacheas, R. Giaffreda, V. Stavroulaki, D. Kelaidonis, V. Foteinos, G. Poulios, P. Demestichas, A. Somov, A. R. Biswas, and K. Moessner. Enabling Smart Cities through a Cognitive Management Framework for the Internet of Things. *IEEE Communications Magazine*, 51(6):102–111, June 2013.

[vid17a] Akiyo. http://www2.tkn.tu-berlin.de/research/evalvid/qcif.html, April 2017.

[vid17b] Bridgefar. http://www2.tkn.tu-berlin.de/research/evalvid/qcif.html, April 2017.

[vid17c] Claire. http://www2.tkn.tu-berlin.de/research/evalvid/qcif.html, April 2017.

[vid17d] Highway. http://www2.tkn.tu-berlin.de/research/evalvid/qcif.html, April 2017.

[VKF12] A. Voellmy, H. Kim, and N. Feamster. Procera: A Language for High-Level Reactive Network Control. In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, pages 43–48, Helsinki, Finland, August 2012.

[VW12] A. Voellmy and J. Wang. Scalable Software Defined Network Controllers. In *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, volume 42, pages 289–290, Helsinki Finland, August 2012.

[WALR+06] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh. Deploying a wireless sensor network on an active volcano. *IEEE internet computing*, 10(2):18–25, April 2006.

[WCH+13] X. Wen, Y. Chen, C. Hu, C. Shi, and Y. Wang. Towards a secure controller platform for openflow applications. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 171–172, Hong Kong China, August 2013.

[WHG+14] C. Wang, F. Haider, X. Gao, X. You, Y. Yang, D. Yuan, H. M. Aggoune, H. Haas, S. Fletcher, and E. Hepsaydir. Cellular architecture and key technologies for 5G wireless communication networks. *IEEE communications magazine*, 52(2):122–130, February 2014.

[WLY+17] C. Wang, C. Liang, F. R. Yu, Q. Chen, and L. Tang. Computation offloading and resource allocation in wireless cellular networks with mobile edge computing. *IEEE Transactions on Wireless Communications*, 16(8):4924–4938, May 2017.

[WS02] A. D. Wood and J. A. Stankovic. Denial of service in sensor networks. *computer*, 35(10):54–62, October 2002.

[WSYX18] B. Wang, Y. Sun, C. Yuan, and X. Xu. LESLA: A smart solution for SDN-enabled mMTC E-health monitoring system. In *Proceedings of the 8th ACM MobiHoc 2018 Workshop on Pervasive Wireless Healthcare Workshop*, page 2, Los Angeles CA USA, June 2018.

[WT01] T. Wolf and J. Turner. Design Issues for High-performance Active Routers. *IEEE Journal on Selected Areas in Communications*, 19(3):404–409, March 2001.

[WYL+17] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang. Joint computation offloading and interference management in wireless cellular networks with mobile edge computing. *IEEE Transactions on Vehicular Technology*, 66(8):7432–7445, March 2017.

[XTL+12] H. Xie, T. Tsou, D. Lopez, H. Yin, and V. Gurbani. Use cases for alto with software defined networks. *Working Draft, IETF Secretariat, Internet-Draft draft-xie-alto-sdn-extension-use-cases-01.txt*, pages 1–26, June 2012.

[XWZ16] W. Xiang, N. Wang, and Y. Zhou. An energy-efficient routing algorithm for software-defined wireless sensor networks. *IEEE Sensors Journal*, 16(20):7393–7400, July 2016.

[YBG13] G. Yao, J. Bi, and L. Guo. On the cascading failures of multi-controllers in software defined networks. In *2013 21st IEEE International Conference on Network Protocols (ICNP)*, pages 1–2, Goettingen, Germany, October 2013.

[YBSS17] 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*, 35(11):2468–2478, October 2017.

[YCEHH17] H. Yetgin, K. T. K. Cheung, M. El-Hajjar, and L. H. Hanzo. A survey of network lifetime maximization techniques in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 19(2):828–854, January 2017.

[YDAG04] L. Yang, R. Dantu, T. Anderson, and R. Gopal. Forwarding and Control Element Separation (ForCES) Framework. https://tools.ietf.org/html/rfc3746, April 2004.

[YHE02] W. Ye, J. Heidemann, and D. Estrin. An energy-efficient MAC protocol for wireless sensor networks. In *Proceedings. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies*, pages 1567–1576, New York, NY, USA, June 2002.

[YJM13] M. Yu, L. Jose, and R. Miao. Software Defined Traffic Measurement with OpenSketch. In *Presented as part of the 10th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 13)*, pages 29–42, Lombard, IL, USA, April 2013.

[YKS14] V. Yazıcı, U. C. Kozat, and M. O. Sunay. A new control plane for 5G network architecture with a case study on unified handoff, mobility, and routing management. *IEEE Communications Magazine*, 52(11):76–85, November 2014.

[YLZ+13] C. Yu, C. Lumezanu, Y. Zhang, V. Singh, G. Jiang, and H. V. Madhyastha. Flowsense: Monitoring network utilization with zero measurement cost. In *International Conference on Passive and Active Network Measurement*, pages 31–41, Hong Kong, China, March 2013.

[ZLG+15] D. Zeng, P. Li, S. Guo, T. Miyazaki, J. Hu, and Y. Xiang. Energy minimization in multi-task software-defined sensor networks. *IEEE Transactions on Computers*, 64(11):3128–3139, November 2015.

[ZMG+13] Deze Zeng, Toshiaki Miyazaki, Song Guo, Tsuneo Tsukahara, Junji Kitamichi, and Takafumi Hayashi. Evolution of software-defined sensor networks. In *2013 IEEE 9th International Conference on Mobile Ad-hoc and Sensor Networks*, pages 410–413, 2013.

[ZVS+15] H. Zhang, S. Vrzic, G. Senarath, N. Dào, Ha. Farmanbar, J. Rao, C. Peng, and H. Zhuang. 5g wireless network: Mynet and sonac. *IEEE Network*, 29(4):14–23, August 2015.

Parte II

Publicaciones

Apéndice A

Lista de Publicaciones

1. Jesús Antonio Puente Fernández, Ángel Leonardo Valdivieso Caraguay, Luis Javier García Villalba: Seguridad en Redes Definidas por Software: Desafíos y Soluciones. Actas de la XIV Reunión Española sobre Criptología y Seguridad de la Información (RECSI 2016), Maó, Menorca, Illes Balears, España, Octubre 26 – 28, 2016.

2. Jesús Antonio Puente Fernández, Luis Javier García Villalba: Flow Conservation Framework for Monitoring Software Defined Networks. In proceedings of the 19th International Conference on Information Technology (ICIT 2017), Paris, France, May 18 – 19, 2017.

3. Jesús Antonio Puente Fernández, Luis Javier García Villalba: Aplicación de Técnicas de Agrupamiento en la Monitorización de Redes Definidas por Software. Actas del XXXII Simposium Nacional de la Unión Científica Internacional de Radio (URSI 2017), Cartagena, Murcia, España, Septiembre 6 – 8, 2017.

4. Jesús Antonio Puente Fernández, Luis Javier García Villalba, Tai-Hoon Kim. Software Defined Networks in Wireless Sensor Architectures. Entropy, 20(4): 225 – 248, March 2018.

5. Jesús Antonio Puente Fernández, Luis Javier García Villalba, Tai-Hoon Kim. Clustering and Flow Conservation Monitoring Tool for Software Defined Networks. Sensors, vol. 18(4): 1079 – 1102, April 2018.

6. Jesús Antonio Puente Fernández, Luis Javier García Villalba. Análisis y Monitorización por Agrupamiento de Contenido Multimedia en Redes Definidas por Software. Actas del X Congreso Iberoamericano de Seguridad Informática (CIBSI 2020), Bogotá, Colombia, 22-24 de enero de 2020.