ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA TELECOMUNICACIONES

"DISEÑO DE UNA RED DE PROVEEDORES DE SERVICIO CON SDN Y SEGMENT ROUTING PARA MEJORAR EL RENDIMIENTO DE LA RED"

Trabajo de titulación

Tipo: Proyecto Integrador

Presentado para optar al grado académico de:

INGENIERO EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES

AUTOR:

JHON JAVIER LOZA LLUCO

Riobamba - Ecuador

2022

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA TELECOMUNICACIONES

"DISEÑO DE UNA RED DE PROVEEDORES DE SERVICIO CON SDN Y SEGMENT ROUTING PARA MEJORAR EL RENDIMIENTO DE LA RED"

Trabajo de titulación

Tipo: Proyecto Integrador

Presentado para optar al grado académico de:

INGENIERO EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES

AUTOR: JHON JAVIER LOZA LLUCO

DIRECTOR: ING. ALBERTO LEOPOLDO ARELLANO AUCANCELA, MSC

Riobamba - Ecuador

2022

Yo, Jhon Javier Loza Lluco, declaro que el presente trabajo de titulación es de mi autoría y los resultados del mismo son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autor asumo la responsabilidad legal y académica de los contenidos de este trabajo de titulación. El patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo

Riobamba, 22 de febrero de 2022

Jhon Javier Loza Lluco
060485592-4

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA TELECOMUNICACIONES

El tribunal de trabajo de titulación certifica que: El trabajo de titulación: Tipo: Proyecto integrador, "**DISEÑO DE UNA RED DE PROVEEDORES DE SERVICIO CON SDN Y SEGMENT ROUTING PARA MEJORAR EL RENDIMIENTO DE LA RED**", de responsabilidad del señor **JHON JAVIER LOZA LLUCO**, ha sido minuciosamente revisado por los Miembros del Tribunal del trabajo de titulación, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el tribunal autoriza su presentación.

NOMBRE	**FIRMA**	**FECHA**
Ing. Franklin Geovanni Moreno Montenegro. **PRESIDENTE DE TRIBUNAL**	FRANKLIN GEOVANNI MORENO MONTENEGRO Firmado digitalmente porFRANKLIN GEOVANNI MORENO MONTENEGRO DN: cn=FRANKLIN GEOVANNI MORENO MONTENEGRO c=EC o=SECURITY DATA S.A. 1 ou=ENTIDAD DE CERTIFICACION DE INFORMACION Motivo:Soy el autor de este documento Ubicación: Fecha:2022-02-22 16:04-05:00	22/02/2022
Ing. Alberto Leopoldo Arellano Aucancela Msc. **DIRECTOR DEL TRABAJO DE TITULACIÓN**	ALBERTO LEOPOLDO ARELLANO AUCANCELA Firmado digitalmente por ALBERTO LEOPOLDO ARELLANO AUCANCELA Fecha: 2022.02.18 15:20:48 -05'00'	22/02/2022
Ing. Jefferson Alexander Ribadeneira Ramirez. Phd. **MIEMBRO DEL TRIBUNAL**	Firmado electrónicamente por: JEFFERSON ALEXANDER RIBADENEIRA RAMIREZ	22/02/2022

DEDICATORIA

El presente trabajo de titulación se lo dedico a mis padres, Rocío Lluco y Mariano Loza, por el apoyo brindado en mi vida estudiantil. A mi hermana, abuelos, tíos y familia en general que siempre estuvo dispuesta a ayudarme cuando lo he necesitado.

Javier

AGRADECIMIENTO

En estas líneas quiero agradecer a mis padres por darme lo necesario para poder alcanzar este objetivo, a mis abuelitos y tíos ya que sin ellos no sería la persona que soy ahora. A todas las personas que alguna vez me brindaron un consejo o una enseñanza para la vida.

A mis amigos, los cuales son mi segunda familia y con ellos hemos compartido experiencias, las cuales a donde sea que vayamos siempre llevaremos en el recuerdo con agrado.

A todos los docentes de la ESPOCH que han compartido su conocimiento y gracias al cual he podido formarme como profesional. Especial mención al Ing. Alberto Arellano por toda la enseñanza en su área durante la carrera, la cual me ha inspirado para desarrollar el presente y futuros trabajos, de igual manera por su apoyo para culminar con éxito el trabajo de titulación.

Javier

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

ÍNDICE DE ABREVIATURAS

SDN:	Software Defined Network (Redes definidas por software)
IGP:	Interior Gateway Protocol (Protocolo de enlace interior)
OSPF:	Open Shortest Path First (Ruta abierta más corta)
MPLS:	Multiprotocol Label Switch (Protocolo de conmutación de etiqutas)
SR:	Segment Routing (Enrutamiento de segmentos)
SRGB:	Segment Routing Global Block (Bloque global SR)
SRLB:	Segment Routing Local Block (Bloque local SR)
CPS:	Conections Per Second (Conexiones por segundo)
PPS:	Packets Per Second (Paquetes por segundo)
FIB:	Forwarding Information Base (Base de información de reenvío)
RIB:	Routing Information Base (Base de información de reenvío)
TI-LFA:	Topology Independ Loop Free Alternate
URI:	Uniform Resource Identifier (Identificador de recurso uniforme)
URN:	Uniform Resource Name (Nombre de recurso uniforme)
URL:	Uniform Resource Locator (Localizador de recurso uniforme)
XML:	Extensible Markup Language (Lenguaje de marcado extensible)

RESUMEN

El presente trabajo tuvo como objetivo el diseño de una red de proveedores de servicio definida por software (SDN) y enrutamiento por segmentos (Segment Routing) para mejorar el rendimiento de la red, se analizaron distintos protocolos de enrutamiento, formas de integrar ambas tecnologías y arquitecturas. Para determinar el rendimiento se realizaron pruebas de estrés con distintas cantidades de tráfico mediante el software generador de tráfico TRex, el capturador Wireshark y el analizador Omnipeek, se observó el comportamiento en base a parámetros de red para concluir que enrutamiento es mejor. Para integrar SDN a la red con Segment Routing se ha implementado el controlador Opendaylight, el cual es un software libre que puede comunicarse con los dispositivos de red, para obtener toda la información sobre ella y configurarla con el protocolo de configuración de red (NETCONF). Una vez se obtuvieron los valores para los parámetros de red, se hizo un análisis estadístico con la prueba Wilconxon para determinar si existía una diferencia significativa entre las tecnologías de enrutamiento. Con el controlador SDN, se comunicó la red para poder administrarla y configurarla de mejor manera gracias a las diferentes herramientas nativas que ofrece el software. Se concluye que Segment Routing tiene mejor rendimiento que el enrutamiento por el protocolo de camino más corto primero (OSPF) para el despliegue de proveedores de servicio, y que Opendaylight es un software mediante el cual se puede administrar y configurar la red en tiempo real de manera más sencilla debido a su interfaz gráfica. Se recomienda estudiar las interfaces Northbound del controlador Opendaylight que gracias a su comunicación con interfaces de programación de aplicaciones (APIs), se puede utilizar herramientas de programación de aplicaciones como POSTMAN para publicar archivos de configuración y con ello automatizar la red.

PALABRAS CLAVE: <TELECOMUNICACIONES Y REDES>, <RED DE PROVEEDOR>, <REDES DEFINIDAS POR SOFTWARE (SDN)>, <ENRUTAMIENTO POR SEGMENTOS (SR)>, <OPENDAYLIGHT>, <NETCONF>, <CAMINO MÁS CORTO PRIMERO (OSPF)>, <INTERFAZ DE PROGRAMACIÓN DE APLICACIONES (API)>.

SUMMARY

The objective of this work was to design a network of service providers defined by software (SDN) and segment routing (Segment Routing) to improve network performance, different routing protocols, ways to integrate both technologies and architects. To determine the performance, stress tests were carried out with different amounts of traffic using the TRex traffic generator software, the Wireshark capturer and the Omnipeek analyzer, the behavior was observed based on network parameters to conclude which routing is better. To integrate SDN into the network with Segment Routing, the Opendaylight controller has been implemented, which is free software that can communicate with network devices to obtain all the information about it and configure it with the network configuration protocol (NETCONF). Once the values for the network parameters were obtained, a statistical analysis was done with the Wilconxon test to determine if there was a significant difference between the routing technologies. With the SDN controller, the network was communicated to be able to manage and configure it in a better way thanks to the different native tools offered by the software. It is concluded that Segment Routing has better performance than routing by the shortest path first (OSPF) protocol for the deployment of service providers and that Opendaylight is a software through which the network can be managed and configured in real time. easier due to its graphical interface. It is recommended to study the Northbound interfaces of the Opendaylight controller that, thanks to their communication with application programming interfaces (APIs), application programming tools such as POSTMAN can be used to publish configuration files and thereby automate the network.

Keywords: <TELECOMUNICATIONS>, <PROVIDER NETWORK>, <SOFTWARE DEFINED NETWORKS (SDN)>, <SEGMENTAL ROUTING (SR)>, <OPENDAYLIGHT>, <NETCONF>, <APPLICATION PROGRAMMING INTERFACE (API)>.

MSc. Wilson G. Rojas
NOMBRE Y FIRMA PROFESOR

INTRODUCCIÓN

El crecimiento que ha tenido el contenido multimedia, la computación en la nube, el aumento de usuarios en la telefonía móvil, el crecimiento del IoT y las continuas mejoras que desean implementar las empresas para reducir los costos mientras mantienen fijo los ingresos. Son factores que las empresas proveedoras de servicio deben satisfacer y para ello deben mirar más allá del modelo tradicional de las redes empresariales.

Es por ello que muchas de las grandes empresas recurren a tecnologías como SDN para revolucionar las operaciones y el diseño de la red. Las redes definidas por software son el nuevo paradigma en las redes y permite a la misma ser controlada de una forma inteligente y central, es decir puede ser programada utilizando aplicaciones de software, ayudando a los operadores de la red que se encargan de la gestión, a que lo hagan de una forma constante e integral, independientemente de la tecnología subyacente.

De la misma forma es necesario que las redes soporten todo el tráfico generado por los usuarios sin presentar problemas en el transporte del mismo, es por ello que en los últimos años se está aplicando Segment Routing en el core de las redes, debido a que simplifica el encapsulamiento de los paquetes, ya que elimina cabeceras tanto de LDP, como RSVP a la vez que hace más sencillo implementar ingeniería de tráfico para mejorar el transporte de datos. Actualmente se despliega SR sobre el plano de datos MPLS y sobre algún IGP (OSPF o IS-IS) sin embargo se está realizando el estudio para el despliegue sobre IPv6 llamado SRv6, el cual promete ser más eficiente que SR sobre MPLS.

El presente trabajo de titulación consiste en proponer un diseño de red que integre SDN y SR que son tecnologías que buscan solucionar los problemas mencionados y mostrar que pueden coexistir y conjuntamente mejorar el rendimiento de la red. Para ello se ha diseñado una red que transporta tráfico de servicios Web, Streaming y VoIP, asemejando el tráfico que tendría una red real, variando el mismo hasta un punto en el cual se pueda observar el rendimiento de SR. A la vez se ha integrado un controlador SDN llamado Opendaylight para la gestión de los dispositivos de red.

ANTECENDENTES

En el mundo actual la mayoría de la información, entretenimiento, comunicación, y demás contenido que consumen las personas es transportado en una red de datos que ofrece un proveedor de servicios, debido a la pandemia el contenido multimedia y streaming ha crecido de una forma significativa, la explosión de la computación en la nube, el internet de las cosas, virtualización de servicios entre otras nuevas tecnologías, han hecho que el tráfico en la red del proveedor crezca hasta convertirse en un problema que puede llegar a congestionar la red y provocar graves problemas. Debido a este elevado crecimiento de usuarios conectados a la red de internet y la multitud de aplicaciones, se propone un nuevo paradigma para las redes de datos, las redes definidas por software SDN. Esta nueva forma de entender las redes trata de solucionar problemas de las redes tradicionales, las cuales son muy complejas y difíciles de gestionar, es muy difícil reconfigurar una red ya implementada, para solventar problemas de fallas, carga y cambios en la topología. Su complejidad aumenta ya que las redes actuales están integradas de forma vertical, el plano de control y de datos están agrupadas de forma conjunta.

Las redes SDN tienen como objetivo romper con estas dificultades y cambiar la estructura vertical de una red, separando el control lógico de la red de los routers subyacentes y switches, promoviendo una centralización lógica del control de la red, e introduciendo la habilidad de programar la red. La separación entre las definiciones de políticas, la implementación en la conmutación de hardware y el reenvío de tráfico es clave para la flexibilidad que se busca, mediante la separación del problema de control de la red en pequeñas piezas más manejables. SDN consigue esto de manera fácil, creando e introduciendo nuevas abstracciones de la red, simplificando la gestión de la misma y facilitando que esta pueda evolucionar.

Un ejemplo de red SDN ha sido diseñado en la Universidad Politécnica de Valencia, en dónde han destinado esta tecnología para una red de videovigilancia IP. Para la simulación utilizaron el controlador SDN ONOS, para centralizar y gestionar el control de la red. En el trabajo se puede observar cómo a medida que el tráfico de un canal va creciendo, la calidad en la imagen se degrada, y muestra cómo al aplicar SDN para que reconfigure ciertos parámetros en la red, el tráfico es distribuido de mejor forma, ofreciendo siempre la misma calidad de transmisión aún cuando el tráfico incremente de forma espontánea. (Jiménez, 2020)

Actualmente la empresa CISCO, desarrolla la implementación de redes SDN con Segment Routing. Debido a que el alto tráfico que van a soportar las redes y la acogida que va teniendo las

redes SDN, es necesario que el núcleo de la red de transporte sea mucho más rápida y eficaz, consumiendo la menor cantidad de recursos de red y se acople a las demandas de tráfico que tendrán los proveedores en los próximos años. (Perrin, 2017)

En Ecuador ya se han realizado diseños de este tipo de red, en la Universidad Católica de Santiago de Guayaquil, se diseñó la red de un tipo de servicio mediante Segment Routing para comunicar a dos clientes, dando como conclusión que una red que implemente Segment Routing con MPLS, ofrecerá mayor eficiencia ya que la red cuenta con menos protocolos de señalización que una red MPLS tradicional, además de que Segment Routing es la puerta para aplicar tecnologías de última generación como lo es SDN. (Parra, 2020)

En otro trabajo de la ESPE, en el que se analiza la factibilidad técnica y económica de una red SD-WAN, concluye que al aplicar una red SDN, en este caso para una red WAN, se puede tener ventajas como comunicación segura, balanceo de tráfico, calidad de servicio, convergencia de la red y alta disponibilidad. De igual forma, esta tesis de cuarto nivel propone el estudio de Segment Routing MPLS para las redes que utilicen SDN.

FORMULACIÓN DEL PROBLEMA

¿Es posible diseñar una red con SDN y Segment Routing para mejorar el rendimiento de la red de los proveedores de servicio?

SISTEMATIZACIÓN DEL PROBLEMA

¿Cuáles son las principales aplicaciones de SDN y Segment Routing en una red de Proveedor de Servicios?
¿Cuáles son las ventajas que aporta la incorporación de SDN y Segment Routing en la red de un Proveedor de Servicios?
¿En qué porcentaje se reduce el tráfico de control en una red con SDN y Segment Routing comparado con una red MPLS tradicional?
¿Cómo la implementación de una red SDN con Segment Routing mejora el rendimiento de una red tradicional?

JUSTIFICACIÓN TEÓRICA

En términos de crecimiento porcentual del tráfico de datos móviles, Oriente Medio y África acumularon la mayor tasa de incremento regional en los últimos años.

- Oriente Medio y África tuvo una tasa de incremento interanual del 72% (multiplicándose por 15,3).
- Europa Central y Oriental acumuló una ratio de incremento interanual del 71% (multiplicándose por 14,4).
- Asia-Pacífico experimentó un crecimiento interanual del 58% (multiplicándose por 9,7).
- Latinoamérica tuvo una tasa de incremento interanual del 59% (multiplicándose por 10,1).
- Norteamérica acumuló una tasa de incremento interanual del 47% (multiplicándose por 6,8).
- Europa Occidental experimentó un crecimiento interanual del 48% (multiplicándose por 7,1).

En términos de generación de tráfico de datos móviles, la región Asia-Pacífico se situó en primer lugar.

- Asia-Pacífico: tuvo 9,5 Exabytes mensuales para 2019.
- Norteamérica: tuvo 3,8 Exabytes mensuales para 2019.
- Europa Occidental: tuvo 2,4 Exabytes mensuales para 2019.
- Europa Central y Oriental: tuvieron 3,5 Exabytes mensuales para 2019.
- Oriente Medio y África: tuvieron 3 Exabytes mensuales para 2019.
- Latinoamérica: tuvo 2 Exabytes mensuales para 2019.

La resiliencia de las arquitecturas de internet y telecomunicaciones, que otorgan una prima a la confiabilidad, incluido el aprovisionamiento por encima de la capacidad, ha sido capaz de manejar este aumento de tráfico. Con el tiempo, los proveedores de servicios de internet continuarán aumentando la capacidad de enrutamiento de fibra y núcleo para mantenerse por delante de las demandas, como el gran crecimiento en video. Esto puede alentar a los proveedores a evaluar nuevos diseños convergentes de enrutamiento óptico y enrutamiento de software. La situación de covid-19 destaca los requisitos para que TI entregue una red confiable, segura y de alto rendimiento. También muestra que los patrones de tráfico de la red pueden cambiar significativamente debido a las condiciones comerciales y sociales.

Los eventos de 2020 alteraron permanentemente el entorno de red corporativa con inversiones requeridas en conectividad remota y un cambio continuo de aplicaciones a la nube. La flexibilidad, la agilidad y la automatización deben ser los factores clave en la planificación de la red a largo plazo. El aumento de los dispositivos móviles, las conexiones máquina a máquina para IoT y el 5G son impulsores clave del crecimiento exponencial de tráfico de datos móviles, este entorno ofrece a los proveedores de servicio la oportunidad única para proporcionar servicios innovadores y experiencias móviles a medida que el IoT se va haciendo real. (hayCanal, 2021)

Telefónica registró un incremento en el tráfico de internet en el 2020 durante la pandemia, equivalente a todo el tráfico del año 2019. Llegando a un crecimiento en un mes, entre el 10 de marzo y el 12 de abril del 35% en comparación del 30% que registró durante todo el 2019. El tráfico de la red de Telefónica, en un año anual es estable, a principios de año y navidad, el tráfico empieza con picos y a partir de ahí se establece y crece entorno a un 5% hasta mediados de año, en donde por vacaciones, el tráfico disminuye, volviendo a crecer cuando empiezan clases y épocas festivas. Debido a este comportamiento, las redes de datos de los proveedores deben ser dinámicas y adaptarse en cada momento a los requerimientos, para optimizar los recursos, pero ofreciendo la mayor calidad posible. El tráfico estimado para 2022 rondará el zettabyte, y se acumularán 12 mil millones de dispositivos móviles y conexiones IoT. (Telefónica, 2020)

JUSTIFICACIÓN APLICATIVA

Una red de un proveedor de servicios está formada por equipos CTE (Connected Telecommunications Equipment) dentro del core para conformar la red de transporte de datos, estos equipos son CISCO IOSXR, los cuales los podemos simular en el software GNS3, y soportan la tecnología Segment Routing. Se van a utilizar servidores para simular distintos tráficos, como FTP, HTTP, VoIP, Streaming entre otros. De esta forma se podrá tener distintos tráficos y se asemejará más a una red real.

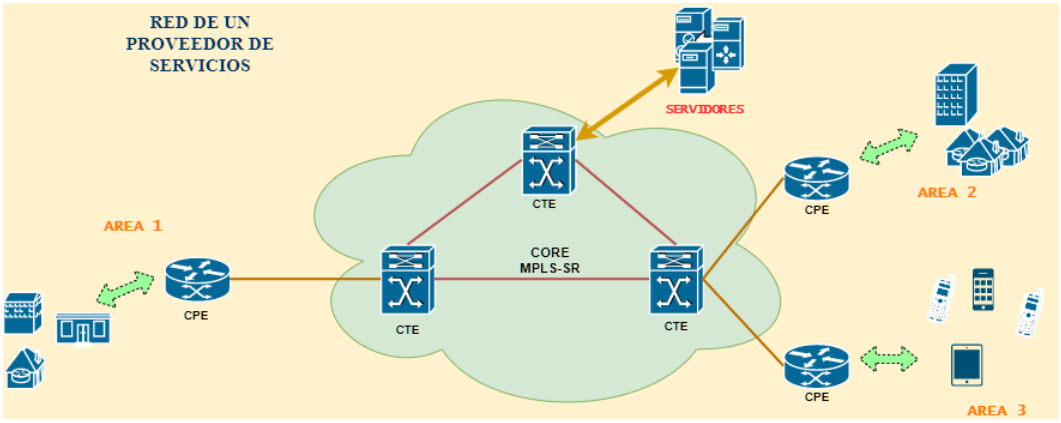


Figura 1. Topología de la red de un proveedor de servicios

Fuente: Loza, J. 2021

Para obtener los datos del rendimiento que tiene la red, es necesario que se transporte tráfico que tendría una red de un proveedor de servicios, el cual puede ser algo básico como un servidor HTP, FTP, etc. Los cuales son muy comúnes en todo tipo de red, sin embargo las redes actuales transportan todo tipo de tráfico, en esta época de pandemia el tráfico de streaming es el que más ha crecido, pero no se puede dejar de lado los servicios de VoIP ya que el despliegue del 5G va a seguir creciendo. Para tener este tráfico en la red se utilizan generadores de tráfico como puede ser Iperf u Ostinato para simular tráfico de HTTP, FTP, ICMP, etc. VLC se utiliza para generar tráfico de streaming y DGIT para el tráfico de VoIp. En el enrutamiento se puede realizar mediante OSPF o ISIS tanto para el core como para los enlaces hacia los CPE de cada área, dentro del core se implementa Segment Routing que puede ser sobre MPLS.

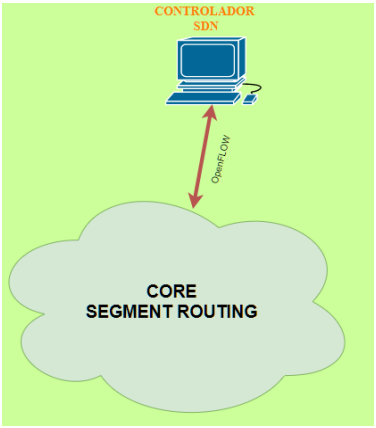


Figura 2. Diagrama de la red con SDN

Fuente: Loza, J. 2021

Para añadir la característica de SDN a la red se debe utilizar un controlador, este controlador debe conectarse a uno de los routers de la red de CORE, el controlador que puede utilizarse es

Opendaylight el cual es de código abierto y está basado en Linux, por lo cual se puede utilizar en una máquina virtual. Para la comunicación entre la red de CORE y el controlador SDN, se puede utilizar Open vSwitch, el cual es un conmutador virtual de código abierto y permite la automatización del tráfico de la red, soporta los protocolos OpenFlow, sFlow, NetFlow, CLI, entre otros.

OBJETIVOS

OBJETIVO GENERAL

Diseñar una red de proveedores de servicio con SDN y Segment Routing para mejorar el rendimiento de la red.

OBJETIVOS ESPECIFICOS

- Estudiar la arquitectura general de las redes SDN en el ámbito corporativo.
- Determinar el protocolo de enrutamiento más adecuado para implementar Segment Routing.
- Analizar los mecanismos de integración de redes SDN con Segment Routing.
- Evaluar mediante distintos escenarios de uso de la red la capacidad que se tendrá con el diseño propuesto y una red tradicional.
- Comparar el rendimiento y gestión de la red para los servicios de VoIP, Streaming y Web entre el diseño propuesto y la red tradicional.

CAPÍTULO I

1. MARCO TEÓRICO

1.1 SDN

1.1.1 Introducción

Software Defined Network desacopla los planos de control y de datos, la inteligencia y el estado de la red están centralizados lógicamente y la infraestructura de red subyacente se abstrae de las aplicaciones. Ofrece una arquitectura dinámica y flexible que protege las inversiones ya existentes y prepara la red para el futuro, ya que la red estática actual puede evolucionar hasta convertirse en una plataforma de prestación de servicios extensible capaz de responder rápidamente a las necesidades cambiantes de la empresa, el usuario final y el mercado de las telecomunicaciones. SDN brinda beneficios sustanciales tanto a empresas como a operadores, tales como:

- Gestión y control centralizados de dispositivos de red de varios proveedores.
- Automatización y administración mejoradas al utilizar API comunes para abstraer detalles de la red subyacente de los sistemas y aplicaciones de orquestación y aprovisionamiento.
- Rápida innovación gracias a las nuevas capacidades y servicios de red sin la necesidad de configurar dispositivos individuales o esperar los lanzamientos de los proveedores.
- Programabilidad por parte de operadores, empresas, proveedores de software independientes y usuarios utilizando entornos de programación comunes, lo que brinda nuevas oportunidades de ingresos e impulsa la diferenciación en los servicios.
- Mayor confiabilidad y seguridad de la red debido a la implementación de una administración centralizada y automatizada de dispositivos de red, aplicación uniforme de políticas y menos errores de configuración.
- Control de red más granular con la capacidad de aplicar políticas integrales y de amplio alcance a nivel de sesión, usuario, dispositivo y aplicación.
- Mejora la experiencia del usuario final, ya que las aplicaciones aprovechan la información centralizada del estado de la red para adaptar sin problemas el comportamiento de la red a las necesidades del usuario. (Open Networking Foundation, 2012, pp.2-3)

1.1.2 Arquitectura de SDN

Los elementos de la arquitectura SDN se encuentran representadas en la figura 1-1, dependiendo de la implementación, ciertos planos pueden ser colocados junto a otros planos o pueden estar separados físicamente. SDN se basa en el concepto de separación entre una entidad controlada y la entidad controladora, el controlador manipula la entidad controlada mediante una interfaz.
SDN abarca varios planos, *forwarding*, *operational*, *control*, *management* y *application*.

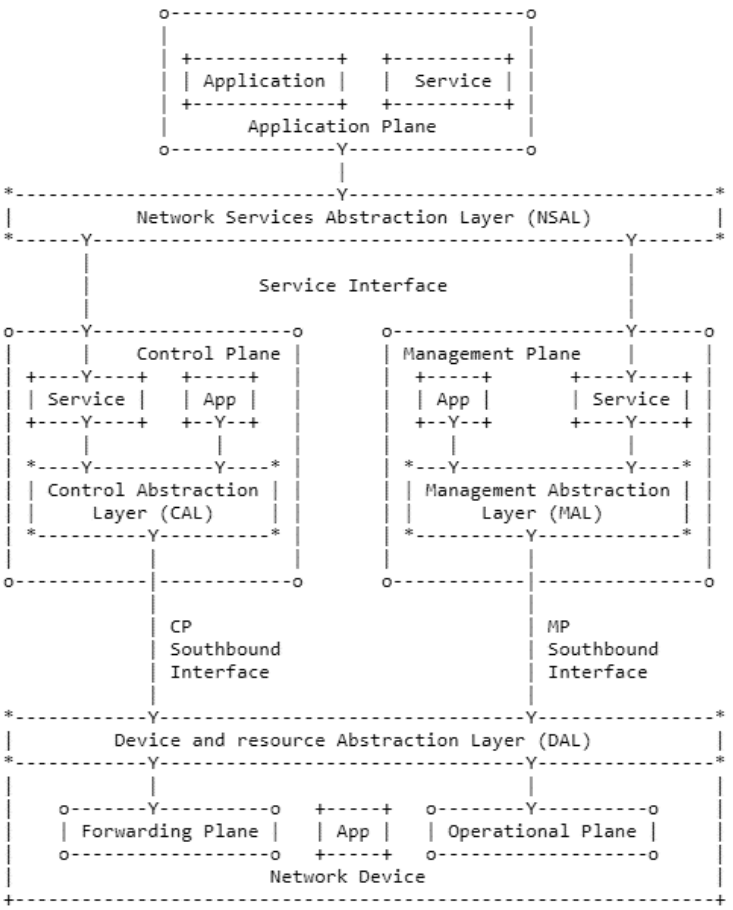


Figura 1-1: Esquema de la arquitectura de SDN

Fuente: Haleplidis et al., 2015, p.8

1.1.2.1 Forwarding Plane

Este plano es el responsable del manejo de los paquetes en el plano de datos en base a las instrucciones recibidas desde el plano de control. Las acciones del plano de reenvío son descartar, cambiar y reenviar paquetes. El plano de reenvío usualmente es el punto de terminación de los servicios y aplicaciones del plano de control.

1.1.2.2 Operational Plane

El plano responsable de la gestión del estado operativo del dispositivo de red; si el dispositivo está activo o inactivo, el número de puertos disponibles, el estado de cada puerto, entre otros. El plano de operación normalmente es el punto de terminación de los servicios y aplicaciones del plano de administración (*management plane*).

1.1.2.3 Control Plane

El plano responsable de la toma de decisiones sobre cómo los paquetes deberían ser reenviados por uno o más dispositivos de la red y de enviar dichas decisiones a los dispositivos de red para su ejecución. El plano de control mayormente se enfoca en el plano de reenvío y en menor medida en el plano operacional del dispositivo, este plano puede estar interesado en la información del plano operacional el cual puede incluir el estado actual de un puerto en particular o sus capacidades. El trabajo principal del plano de control es ajustar las tablas de reenvío, según la topología de la red o las peticiones de servicios externos.

1.1.2.4 Management Plane

El plano responsable de monitorear, configurar y mantener los dispositivos de red. Normalmente tiene mayor enfoque sobre el plano operacional del dispositivo y en menor medida sobre el plano de reenvío. Este plano puede ser utilizado para configurar el plano de reenvío a través de un enfoque más completo que el plano de control, aunque se usa con poca frecuencia para ello. El plano de gestión (*management plane*) puede configurar todas o una parte de las reglas de reenvío a la vez, aunque se espera que esta acción se tome con moderación.

1.1.2.5 Application Plane

El plano de aplicación es el lugar en donde se encuentran las aplicaciones y servicios que definen el comportamiento de la red. Las aplicaciones que apoyan directamente el funcionamiento del plano de reenvío no se consideran parte del plano de aplicación. Las aplicaciones pueden estar implementadas de forma modular o distribuida, por lo que las aplicaciones pueden abarcar varios planos. (Haleplidis et al., 2015, pp. 9-10)

1.1.3 Arquitectura de las redes SDN en el ámbito corporativo

SDN muestra todo su potencial en redes de gran tamaño, por lo que son ideales para redes empresariales.

Las redes SDN empresariales son redes de producción, las cuales tienen una infraestructura física grande y los dispositivos que la conforman se encuentran distribuidos en diferentes nodos a lo largo de una zona geográfica extensa. Una característica necesaria de una red empresarial es que debe contar con tecnologías de virtualización y funciones en la nube, ya que siempre una empresa tiene objetivo la disminución de costos sin perder capacidades, por lo que la integración de estas tecnologías es importante. Sin embargo, también deben permitir la coexistencia o integración con las redes actuales, ya que se debe mantener el servicio que ya existe a los clientes.
Estas redes se caracterizan por ofrecer soporte de gran cantidad de usuarios, estabilidad, escalabilidad, seguridad, alto rendimiento, resiliencia, entre otros. (Quimbayo Rodríguez, 2020, p.24)

Esta arquitectura se representa en la figura 2-1, donde la capa de aplicación está conformada por el plano de aplicación y la capa NSAL, se priorizan o se da mayor importancia a la implementación de las aplicaciones de negocios, como puede ser la gestión de movilidad, el control de acceso, monitoreo de tráfico y seguridad. Estas aplicaciones se comunican con la capa de control por la interfaz Northbound y mediante protocolos como RESTful, Procera, Frenetic, entre otros.

La capa de control lo conforma el controlador SDN, el plano de control, servicios y aplicaciones para las funciones de control, la capa CAL y MAL, existen multitud de controladores tanto comerciales como de código libre.

La capa de infraestructura mejor conocida como plano de datos o reenvío, está conformada por los dispositivos SDN físicos y virtuales de la red, el plano de reenvío y el plano operacional, las distintas aplicaciones de red y la capa DAL. La comunicación entre esta capa y la de control se realiza mediante la interfaz Southbound y protocolos como OpenFlow que es un estándar para SDN, Netconf/YANG, sFlow-RT, entre otros.

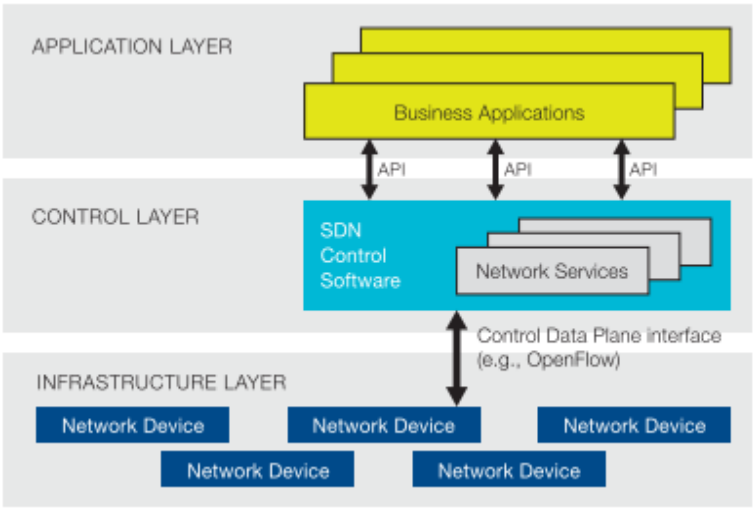


Figura 2-1: Capas de la arquitectura SDN en redes empresariales

Fuente: Open Networking Foundation, 2012, p.7

1.1.4 Integración SDN y Segment Routing

Existen múltiples formas en la que se integra una red con SDN sin embargo para la integración mediante Segment Routing existen dos formas, por medio de PCEP y utilizando el propio protocolo de enrutamiento para la comunicación.

Segment Routing como protocolo de comunicación, en principio el protocolo diseñado para comunicar un controlador con la red es OpenFlow, mediante el cual se pretende programar el estado de flujo directamente en la FIB desde el controlador, por ello SDN requiere separación entre la RIB y FIB. En lugar de un protocolo de enrutamiento que configure la RIB, con SDN y por medio de una API se puede programar la RIB desde las interfaces Northbound y por medio de un protocolo como REST.

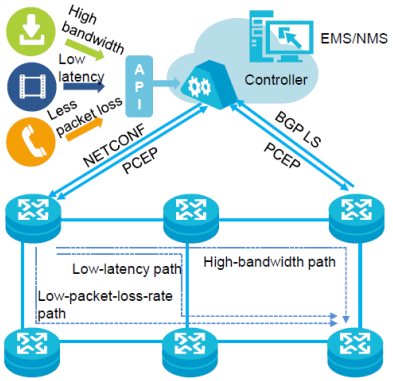


Figura 3-1: Integración de una red SDN

Fuente: ciscolive.com

El problema con OpenFlow es que provoca que las redes tengan estado, mientras que los routers modernos trabajan sin estado, reenvían el tráfico basado en las etiquetas entrantes o la IP de destino. Este tipo de reenvío necesita de dos búsquedas en la FIB, la primera búsqueda se realiza en la tarjeta de línea de entrada para determinar la tarjeta de línea de salida. La otra búsqueda se encarga de resolver la adyacencia en capa 2 y reescribe el encabezado. En una red con SDN y OpenFlow, la búsqueda de FIB coincide con 3-Tuple o 7-Tuple, lo que provoca una programación bidireccional de estado de flujo en la FIB a través de la red. Un nodo de red puede parecer simple como el plano de control eliminado, sin embargo, el plano de datos se vuelve exponencialmente complicado ya que necesita mantener los estados de flujo para cada aplicación en la red. Además, una SDN clásica no tiene la capacidad para realizar estos desafíos.
Cuando se despliega SDN, la filosofía de la red pasa a ser: no programar el flujo en la red, programar los paquetes. Es por ello, que se propone a Segment Routing como una alternativa genuina para OpenFlow.

Segment Routing en el plano de control utiliza las extensiones del protocolo IGP/BGP para la distribución de etiquetas salto a salto, mientras que en el plano de datos puede utilizar MPLS o IPv6, por lo que no es necesario realizar cambios en la FIB. SR con plano de datos IPv6 se llama SRv6. Una conexión a gran escala con SR puede admitir varios casos de uso de SDN que se extienden a diferentes partes de una red de proveedores de servicio. SR como tecnología de enrutamiento de origen puede coexistir con cualquier paradigma de enrutamiento de destino y cambio de etiquetas. (Abuhayat, 2020)(Paraschis, 2019, pp.68-76)

Otra forma de integrar SDN y SR es mediante el protocolo PCEP, el cual es uno de los protocolos de comunicación para redes extensas y controladores SDN. Lo primero a tener en cuenta es que se debe utilizar BGP-LS ya que permite inyectar información del IGP en BGP, y mediante este protocolo llevar la información del estado del enlace de la red al controlador SDN y después mediante PCEP calcular la ruta a través de la red. En esta arquitectura intervienen los elementos PCC el cual corresponde al cliente del cálculo de ruta que viene a ser un router dentro de la red y el PCE que es el elemento de cálculo de ruta, es decir el controlador SDN. En la figura 4-1, se puede observar la integración de SDN y el core de una red con Segment Routing mediante el protocolo de comunicación PCEP y BGP-LS, los cuales se encargan conjuntamente de informar el estado de la red, realizar el cálculo de la ruta e inyectarla en los routers correspondientes. (At, 2020, pp.1-23)(Paraschis, 2019, pp.21-31)

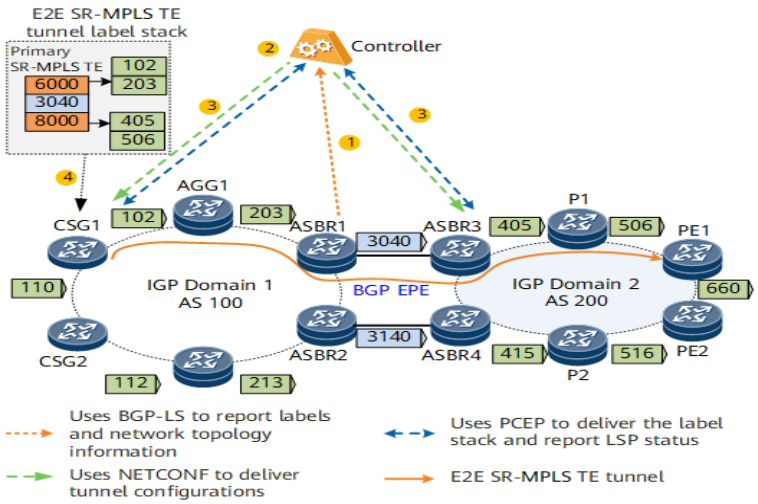


Figura 4-1: Integración de una red SR y SDN mediante diferentes protocolos

Fuente: Paraschis, 2019, pp.23

1.1.5 Controlador SDN

Es el elemento principal al momento de implementar SDN en una red ya que dependiendo de sus características se puede desplegar las distintas tecnologías, servicios y bondades que ofrece SDN en una red, existen multitud de controladores tanto comerciales como de software libre. Los controladores más populares son ONOS, OpendayLight y Floodlight.

1.1.5.1 Opendaylight

Es un controlador para SDN desarrollado por la *Linux Foundation*, es una plataforma abierta modular que tiene como objetivo personalizar y automatizar las redes de cualquier tamaño y escala. Fue diseñada desde el principio como base para soluciones comerciales que abordan una variedad de casos de uso en entornos de red existentes.

ODL es la plataforma de controlador SDN de código abierto más implementada y en sus 8 años de vida, cuenta ya con 13 versiones y una comunidad en constante desarrollo.

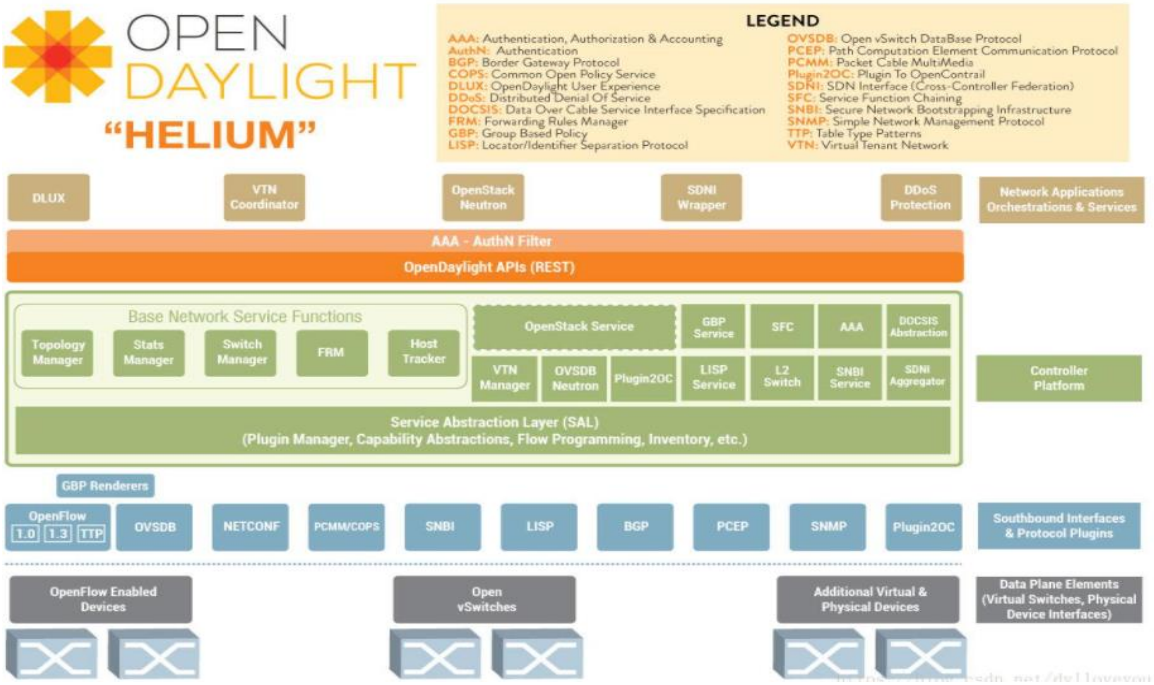


Figura 5-1: Arquitectura del controlador Opendaylight

Fuente: Marban, 2020

El núcelo de Opendaylight es la capa de abstracción de servicios basada en modelos o conocido como MD-SAL. En ODL los dispositivos de red subyacentes y las aplicaciones de red se representan como objetos o modelos cuyas interacciones se procesan dentro de la SAL.

SAL es un mecanismo de intercambio y adaptación de datos entre modelos YANG que representan dispositivos y aplicaciones de red. Los modelos YANG proporcionan descripciones generalizadas de las capacidades de un dispositivo o aplicación sin necesidad de que ninguno de los dos conozca los detalles de implementación específico del otro. En la SAL los modelos se definen por sus respectivos roles en una interacción dada. Se pueden dar dos modelos, el modelo productor implementa una API y proporciona los datos de la API, el modelo consumidor utiliza la API y consume los datos de la API. Mientras que las interfaces Northbound y Southbound brindan una visión al ingeniero de red sobre SAL, los modelos consumidor y productor son descripciones más precisas de las interacciones dentro de la SAL.

La plataforma ODL está diseñada para permitir a los usuarios intermedios y proveedores de soluciones la máxima flexibilidad en la construcción de un controlador que se adapte a sus necesidades. El diseño modular permite que cualquier persona en el ecosistema ODL aproveche los servicios creados por otros, escribir e incorporar sus propios servicios y compartirlos con la comunidad. Opendaylight brinda soporte para los protocolos en cualquier plataforma SDN, como

Openflow, OVSDB, NETCONF, BGP, PCEP y muchos más, permitiendo mejorar la programabilidad de las redes modernas y resuelven una variedad de necesidades de los usuarios.(Opendaylight, 2021)

1.2 NETCONF/YANG

NETCONF es un protocolo de comunicación creado en el 2006 por la IETF con la intención de tener un protocolo para configurar remotamente equipos de red, por lo que con NETCONF se tiene mecanismos para instalar, manipular y eliminar la configuración de los dispositivos, todo esto de forma simultánea en varios dispositivos de la red.

El protocolo utiliza un mecanismo llamado RPC (*Remote Procedure Calls*) para la comunicación entre el cliente y un servidor, en dónde el cliente puede ser un script o una aplicación que se ejecuta como parte de un administrador de red o en un ambiente SDN, el servidor es el dispositivo de red que se quiere administrar. (Enns et al., 2011, p6)

1.2.1 Arquitectura

NETCONF es un protocolo de capa 7 en base al modelo OSI y en su arquitectura se pueden diferenciar 4 capas que se muestran en la figura 6-1:

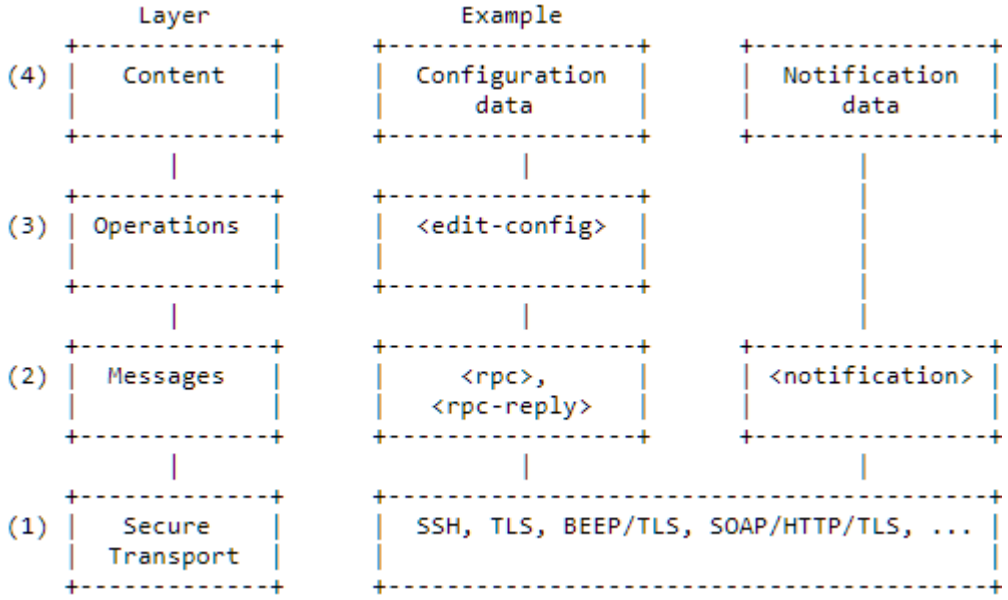


Figura 6-1: Arquitectura de NETCONF

Fuente: Enns et al., 2011, p.9

1. Transporte: se encarga de proveer la comunicación entre el cliente y servidor, y mantener activas las sesiones. NETCONF puede ser transportado por cualquier protocolo que cumpla con ser orientado a conexión, proveer autenticación, integridad de datos, confidencialidad y protección. Este protocolo debe establecer un mecanismo para indicar el tipo de sesión, ya sea cliente o servidor. El protocolo de transporte más utilizado en una implementación de NETCONF es SSH.
2. Mensajes: la capa de mensajes proporciona un mecanismo para codificar llamadas de procedimiento remoto (RPCs) y notificaciones.
3. Operaciones: en esta capa se definen las operaciones base del protocolo las cuales son invocadas por los RPC bajo los parámetros del lenguaje de codificación XML, estos mensajes deben ser *well-formed XML* codificado en UTF-8. Mediante las operaciones se pueden administrar las configuraciones de los dispositivos y recuperar información de estado de los mismos.
4. Contenido: no se define el formato que debe tener la información de configuración, el RFC 6241 define que esto se deja a consideración de cada fabricante. Sin embargo, se pretende que exista un formato estándar para esta información, y en el RFC 6020 se define el lenguaje de modelado de datos YANG, en el cual se especifica los modelos de datos y las operaciones de las capas 3 y 4 de la pila del protocolo NETCONF, por ello se puede entender que YANG sustituye estás capas y se denomina al protocolo como NETCONF/YANG. (Enns et al., 2011, p.9)

La comunicación se establece entre el cliente y servidor mediante una sesión segura, para ello se utiliza los mensajes RPC en base al formato XML, lo que permite que la jerarquía de datos compleja pueda ser expresada en un formato de texto que puede ser leído, guardado y manipulado.

1.2.2 XML

eXtensible Markup Language es un lenguaje de marcado de datos que define una sintaxis mediante un conjunto de reglas para poder escribir o incrustar metadatos. Los documentos basados en XML contienen la descripción de los datos que forman el documento, no se encuentra las referencias a la representación del mismo, lo que permite independizar el contenido de un documento de XML de su representación, lo que permite tener diferentes representaciones de un mismo documento mediante un proceso intermedio de transformación. (Barrancos Martínez, 2003, pp.2-4)

```
<?xml version="1.0" encoding="UTF-8"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <capabilities>
    <capability>urn:ietf:params:netconf:base:1.0</capability>
  </capabilities>
</hello>]]>]]>
```

Figura 7-1: Documento XML

Fuente: Andy, 2020

En la figura 7-1 se puede observar un ejemplo de documento XML utilizado por NETCONF, en los cuales se puede apreciar la presencia de la instrucción *namespace*, utilizado para definir todos los elementos del protocolo mediante el identificador URI. De igual forma en el siguiente campo se observa la instrucción *capabilites*, utilizado por NETCONF para definir el conjunto de funciones que complementa a las funciones básicas del protocolo, describiendo las operaciones adicionales como el contenido permitido dentro de las operaciones. Se pueden definir nuevas capabilities en cualquier momento en los documentos, lo que permite el conjunto de capacidades sea más grande. (Enns et al., 2011)

1.2.3 RPC

Los miembros de una sesión intercambian <rpc> y <rpc-reply> en una comunicación correcta entre ambos. El elemento <rpc> se utiliza para incluir una solicitud NETCONF enviada desde el cliente al servidor, este elemento contiene el atributo "message-id" el cual es una cadena elegida por el emisor del <rpc> que codifica un número entero que va aumentando. El receptor del <rcp> no decodifica o interpreta esta cadena, lo guarda para utilizarlo como "message-id" en el <rpc-reply> el cuál es utilizado como respuesta al <rpc>.
Dentro del <rpc> se invocan diferentes métodos del protocolo NETCONF con sus parámetros, si es que los tuviese.

Cuando se produce un error en el procesamiento de una petición <rpc> se envía un <rpc-error> dentro del <rpc-reply>. Un servidor no debe regresar información específica de un error como el nivel de aplicación o un modelo de datos específico para el cual el cliente no tiene derechos de acceso suficiente. (Enns et al., 2011, pp. 13-16)

Un elemento <rpc-error> contiene las siguientes partes:

Tabla 1-1: Mensajes de error rpc

Instrucción	**Información**
Error-type	Indica la capa en la cual ocurrió el error, puede ser la capa de transporte, de mensajes (rpc), de protocolo (operaciones) o aplicación (contenido)
Error-tag	Contiene una cadena que identifica al error determinado por el dispositivo.
Error-severity	Identifica mediante una cadena la gravedad del error, el cual puede ser como error o una advertencia.
Error-app-tag	Identifica el modelo específico de datos o el error de implementación específica. Este elemento es complementario para añadir más información por lo que puede o no estar presente en el <rcp-error>.
Error-path	Informa sobre la ruta al nodo al que un error está asociado en un <rcp-error> en particular.
Error-message	Es un mensaje que contiene una cadena para que una persona la pueda visualizar, describe la condición del error.
Error-info	Contiene información sobre errores específicos del protocolo o modelo de datos.

Fuente: Enns et al., 2011

Realizado por: Loza, J. 2021

De igual manera cuando no existen errores en el procesamiento del <rpc>, se utiliza el elemento <ok> en el mensaje <rpc-reply>.

1.2.4 Funcionamiento

El protocolo NETCONF fue diseñado para separar el estado y la configuración de los datos, y proporciona diferentes operaciones básicas y complementarias mediante los capabilities para cada uno. Estas operaciones son un elemento que está descrito dentro del mensaje <rpc> y se observan en la tabla 2-1.

Tabla 2-1: Operaciones NETCONF

Operación	**Descripción**
get-config	Recupera toda o parte de una configuración específica del datastore. Cuando se puede responder con la configuración se envía un elemento <rpc-reply> que contiene un elemento <data> donde se encuentra la configuración del dispositivo.
edit-config	Carga toda o parte de una configuración específica en el datastore. La nueva configuración puede ser expresada como archivo local, remoto o en línea, si el destino no existe en el datastore, se creará. Mediante el parámetro target se indica el nombre de la configuración a editar.
copy-config	Crea o reemplaza la configuración completa de un datastore con el contenido de otra configuración del datastore. Si el destino ya existe en el datastore, se reescribe de lo contrario se crea uno nuevo.

delete-config	Elimina una configuración del datastore.
lock	Permite al cliente bloquear todo el sistema del datastore de configuración de un dispositivo. Este bloqueo es corto y permite al cliente realizar un cambio sin miedo de una interacción con otro cliente que sea o no NETCONF o con otra persona.
unlock	Permite liberar una configuración bloqueada.
close-session	Utilizada para finalizar la sesión actual correctamente.
kill-session	Utilizada para forzar el cierre de una sesión que no puede ser la actual.

Fuente: Enns et al., 2011

Realizado por: Loza, J. 2021

Estas operaciones básicas son utilizadas para operar una o más datastores, como se muestra en la figura 8-1.

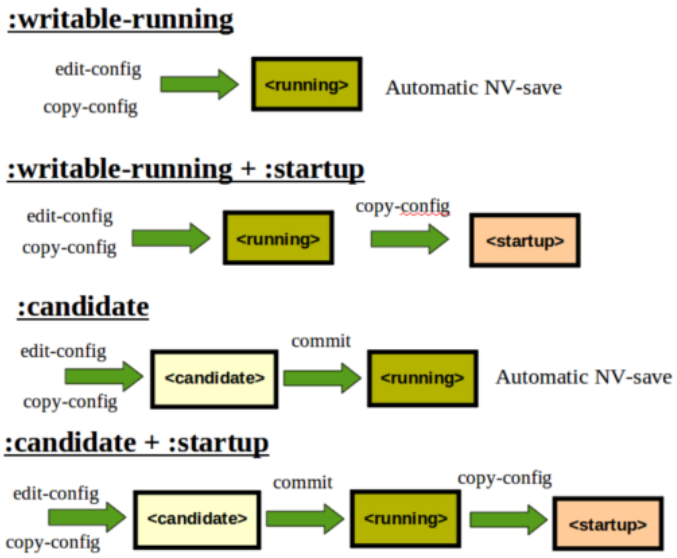


Figura 8-1: Estructura de operaciones NETCONF

Fuente: Andy, 2020

En NETCONF se definen 3 datastore estándar las cuales se describen a continuación, sin embargo también se pueden operar datastores fuera de línea mediante el elemeto <url>.

El datastore **<running/>** representa toda la configuración activa actualmente en el dispositivo, es el único datastore estándar obligatorio. A menos que el servidor soporte la capabilite **:candidate** este debe permitir que el datastore sea editado directamente, de lo contrario no lo permitirá. Si es permitido, se anuncia el capabilite **:writable-running**.

Este datastore también es utilizado para contener la información del estado conceptual actual del dispositivo, lo que permite a operaciones como <get>, que operan en la base de datos **<running/>** pueden recuperar información de estado y estadísticas, además de los parámetros de

configuración. La operación <get-config> se puede utilizar en lugar de <get> para recuperar solo los datos de configuración.

El datastore <**candidate**/> está disponible si el servidor soporta el capabilite **:candidate**, es un bloc de notas global utilizado para recolectar ediciones a partir de una o más operaciones <edit-config>. Un cliente puede construir un conjunto de cambios los cuales pueden ser o no validados por el servidor, hasta que explícitamente se comprometa a la configuración en ejecución.
El cliente puede utilizar la operación <commit> para activar los cambios inscrustados en el datastore <**candidate**/> y hacerlo parte del <**running**/>. Después de que la operación <commit> sea exitosa todos los datastore tienen el mismo contenido de configuración.

El datastore <**startup**/> está disponible si el capabilite **:startup** es soportado por el servidor. Representa la configuración que se utilizará cuando se reinicie el equipo, si está presente el servidor no guardará los cambios en el datastore <**running**/> de forma automática en un almacenamiento volátil, en su lugar se requiere de la operación <copy-config> para sobre escribir el contenido del datastore <**startup**/> con la configuración actual. (Andy, 2021)

1.2.5 YANG

YANG es el lenguaje de modelado de datos que se utiliza para modelar la configuración y el estado de los datos que proporciona el protocolo NETCONF. Este lenguaje modela la organización jerárquica de datos en forma de árbol en el cuál cada nodo tiene un nombre y también un valor o un conjunto de nodos secundarios.

YANG estructura los modelos de datos en módulos y submódulos, el cual puede importar datos desde módulos externos o incluir datos desde submódulos. Un módulo contiene tres tipos de declaraciones: modulo de cabecera, revisión y definición. La declaración de cabecera del modulo describe al mismo y brinda información sobre él. La revisión da información acerca de la historia del modulo y la definición es el cuerpo del módulo en donde se define el modelo de datos.
El módulo es la unidad básica de definición YANG, define un solo modelo completo, cohesivo, o aumentar un modelo de datos existente con nodos adicionales.
Los submódulos son módulos parciales que contribuyen en la definición de un módulo. Un módulo puede incluir cualquier cantidad de submódulos, pero cada uno debe pertenecer a un solo módulo. (Bjorklund, 2010)

Este lenguaje de modelado de datos define cuatro tipos de nodos:

- *Leaf Nodes:* contiene datos simples como un entero o una cadena, tiene un valor de un tipo en particular y no contiene nodos secundarios.
- *Leaf-List Node:* es una secuencia de *Leaf Nodes* con un solo valor de un tipo en particular por cada *Leaf.*
- *Container Node:* utilizado para agrupar nodos relacionados. Un contenedor solo tiene nodos secundarios, ningún valor.
- *List Node:* define la secuencia de entrada de listas, cada entrada es como una estructura o instancia de registro y es identificada únicamente por los valores de sus *Leafs*.

Cuando se define un nodo se puede declarar si es configurable o no, si el nodo se declara mediante "config-false" el nodo es declarado como dato de estado por lo que se reporte mediante la operación <get>. Mientras que al declararlo como "config-true" es configurable mediante la operación <get-config>. (Merelo Hernández, 2017, p.17)

```
YANG Example:

  rpc activate-software-image {
      input {
          leaf image-name {
              type string;
          }
      }
      output {
          leaf status {
              type string;
          }
      }
  }

NETCONF XML Example:

  <rpc message-id="101"
       xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <activate-software-image xmlns="http://acme.example.com/system">
      <image-name>acmefw-2.3</image-name>
   </activate-software-image>
  </rpc>

  <rpc-reply message-id="101"
             xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <status xmlns="http://acme.example.com/system">
      The image acmefw-2.3 is being installed.
    </status>
  </rpc-reply>
```

Figura 9-1: Ejemplo de modelo YANG

Fuente: Bjorklund, 2010, p.23

Los modelos de YANG son mapeados en formato XML y transmitidos por el protocolo NETCONF, de igual forma permite describir operaciones utilizando rpc, con las instrucciones input y output para definir los parámetros que contiene la operación.

1.3 RESTful

Representational State Transfer es una arquitectura de comunicación utilizado en la interfaz Northbound de un controlador SDN lo que le permite al mismo poder monitorear y administrar la infraestructura de red mediante el protocolo HTTP desde un servicio Web, como se puede ver en la figura 10-1.

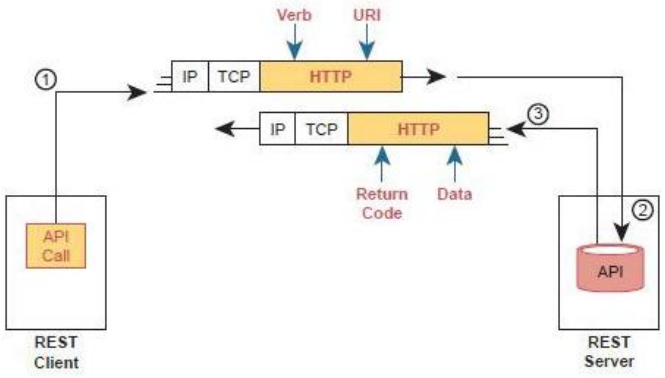


Figura 10-1: Encabezamiento de REST

Fuente: Jesus, 2021

RESTful es un servicio web basado en la arquitectura de software REST, utiliza JSON como formato de los mensajes y verbos HTTP para transportar la información entre el cliente y servidor, normalmente desde una API. Cuando el cliente realiza una solicitud, esta envía la representación del estado del recurso que se ha solicitado desde un servidor.

La arquitectura de REST utiliza los verbos HTTP para identificar las operaciones CRUD requeridas y se muestra en la tabla 3-1:

Tabla 3-1: Verbos para REST

Verbo	**Función**
Put	Crear o reemplazar datos, registra actualizaciones.
Get	Leer datos en el host.
Post	Crea datos en el servidor.
Delete	Borrar información.
Patch	Realizar actualizaciones específicas de ciertos datos, no es muy usado.
Head	Cabecera de los metadatos

Fuente: Restfulapi, 2021

Realizado por: Loza, J. 2021

1.3.1.1 Arquitectura

La arquitectura de REST está compuesta por 3 elementos como se muestra en la figura 11-1.

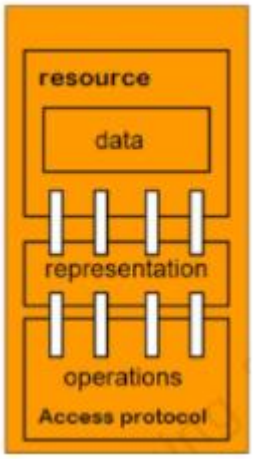


Figura 11-1: Arquitectura REST

Fuente: Marban, 2020b

Los recursos son la abstracción clave de la información, cualquier tipo de información o datos que se pueda nombrar es un recurso. El estado de un recurso se le conoce como la representación y consiste de los datos, metadatos y los enlaces hipermedia. Las operaciones que se realizan mediante los verbos HTTP, para realizar el intercambio de información. (Restfulapi, 2021)

Para identificar un recurso en particular se utiliza el URI que es un identificador uniforme de recurso, este se compone de la unión del URL y URN, como se muestra en la figura 12-1.

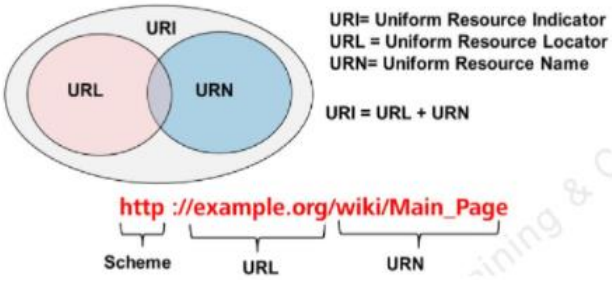


Figura 12-1: Estructura URI

Fuente: Marban, 2020b

Para que una arquitectura se considere REST debe cumplir las siguientes características:

- Interfaz Uniforme: los recursos se deben identificar mediante una URI específica para que se pueda acceder desde los métodos de HTTP.

- Sistema de capas: permite aumentar la escalabilidad o la implementación de políticas de seguridad.
- Peticiones sin estado: el servidor no guarda los datos de la consulta, por lo que siempre en cada consulta se debe indicar toda la información necesaria para entender la solicitud.
- Cacheable: se debe permitir que las respuestas sean marcadas como cachebales o no, es decir que se puedan almacenar, por lo que el cliente sabrá si se reutilizará los datos obtenidos.
- Separación cliente y servidor: separa la interfaz de usuario del servidor y el almacenamiento de datos. Esto ayuda al desarrollo frontend y backend.
- Código bajo demanda: se puede ampliar la funcionalidad del cliente mediante la descarga y ejecución del código en forma de scripts. El código descargado simplifica a los clientes al reducir la cantidad de funciones que deben implementarse previamente. (Restfulapi, 2021)(Jesus, 2021)

Uno de los componentes de RESTful son los códigos de estado HTTP, mediante el cual las personas pueden entender el estado de una solicitud HTTP, estos códigos se muestran en la figura 13-1.

Código	Descripción Humana
201	Creado OK, respuesta correcta frente a un POST.
204	Modificado OK, respuesta correcta frente a un PUT
400	Error en la petición, suele ocurrir frente a URIs mal armadas o con campos incorrectos en el cuerpo del mensaje.
401	Error en las credenciales.
404	No se encuentra el recurso. Posible problema en la URI
409	Conflicto en la creación. En un POST, significa que ya existe el recurso (se debería modificar con PUT).

Figura 13-1: Códigos de estado HTTP

Fuente: Lucas, 2020

1.4 Enrutamiento en redes de proveedores de servicio

En un proveedor de servicios se puede tener enrutamiento por OSPF, IS-IS o EIGRP, este último es un protocolo propietario de CISCO por lo cual solo están presentes para redes que tengan todos los equipos de la marca, por ello los protocolos estándares son OSPF y IS-IS.

El protocolo de IS-IS es ideal para trabajar en redes backbone ya que al trabajar solo en capa de enlace de datos, la cabecera tiene menor tamaño por lo tanto la actualización de la información de enrutamiento es más eficiente. Este protocolo también tiene la característica de ser transparente ya que puede trabajar con IPv4 e IPv6 mediante un solo proceso por lo que la migración de un protocolo a otro es más fácil de hacer que con OSPF, por lo que IS-IS es más escalable.

OSPF es el protocolo más utilizado en redes de gran escala, sin embargo ha sido implementado en redes ISP debido a que en el medio la empresa Mikrotik se ha posicionado en el mercado siendo uno de los proveedores de equipos más importante, sin embargo solo ofrecen soporte para OSPF. Una de las empresas de telecomunicaciones más importantes como es Telconet S.A. tenía en el 2005 enrutamiento mediante OSPF, en el mismo año empezaron la migración a la tecnología MPLS para mejorar la calidad de sus servicios.

Para la presente tesis se ha decidido utilizar el protocolo de enrutamiento OSPF ya que es el protocolo que está presente en la mayoría de proveedores de servicio a pesar de que IS-IS teóricamente es el más adecuado para redes de proveedores. Sin embargo debido al soporte de los equipos que están siendo tendencia en el medio y ya que IS-IS solo llega hasta capa 2, y para el presente trabajo se necesita visualizar como se encapsulan los paquetes es necesario que llegue a capa 3, se utiliza el protocolo OSPF.

1.5 Segment Routing

1.5.1 Introducción

El enrutamiento tradicional se basa en el reenvío de paquetes IP por las mejores rutas seleccionadas por el IGP a cargo (OSPF, IS-IS, EIGRP, RIP) a las direcciones IP de destino. Sin embargo, debido a que existen multitud de aplicaciones las cuales requieren de reenvío de tráfico seleccionado surge la necesidad de nuevas técnicas que faciliten el reenvío de los paquetes por medio de una red enrutada.

Segment Routing parte de la base de solventar varios problemas y sobre todo con el objetivo de simplificar la red IP, mediante la idea de otorgar la responsabilidad de codificar el camino explicito en los paquetes al router de ingreso en lugar del host que los envía, para que el enrutamiento IP sea más flexible y escalable. Para lograr esto, Segment Routing combina las mejores características de MPLS y Source Routing. La codificación del camino consiste en colocar una secuencia de instrucciones en la cabecera del paquete, a cada instrucción se le conoce con el nombre de segmento, el cual utiliza una identificación recibiendo la nomenclatura para la instrucción de *Segment ID*. Por lo tanto, en Segment Routing la ruta no depende de la señalización mediante saltos como en un IGP, un protocolo de distribución de etiquetas (LDP) o RSVP. Una característica que hace de SR una tecnología muy interesante para el futuro de las redes es su adopción a SDN, ya que puede funcionar con un plano de datos MPLS o IPv6, consiguiendo que el mecanismo de reenvío de paquetes sirva como alternativa a OpenFlow. (Santos, 2019, p.1) (P, 2018, p.1)

1.5.2 Componentes de Segment Routing

En la tecnología de segment routing se definen nuevos conceptos para describir el funcionamiento y las características:

- *SR-MPLS:* la instanciación de segment routing en un plano de datos MPLS.
- *Segment:* es la instrucción que ejecuta un nodo a un paquete entrante, puede ser el reenvío del paquete de acuerdo con la ruta más corta hacia el destino, el reenvío de paquete a través de una interfaz específica, la entrega de un paquete a una aplicación o servicio, etc. En la figura 14-1 se observan los segmentos 507 y 709 en el router E, el segmento 507 indica que el paquete debe viajar al router G.

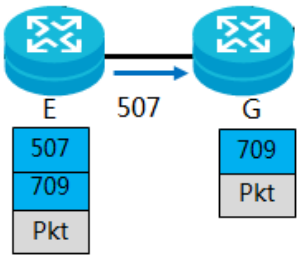


Figura 14-1: Segmento de SR

Fuente: ciscolive.com

- *SID: Segment ID,* el identificador de un segmento. En la figura 14-1 se observa el identificador 507 y 709 para cada segmento.
- *SR-MPLS SID:* una etiqueta de MPLS o un valor de índice en un espacio de etiqueta MPLS asociado a un segmento. Este rango de etiquetas va desde 16,000 hasta el 23,999.

- *Segment Routing Domain (SR domain):* se trata del conjunto de nodos que participan en un modelo de enrutamiento basado en el origen. Estos nodos pueden estar conectados a la misma infraestructura física, a la vez que pueden estar conectados remotamente entre ellos. Si muchas instancias de protocolos están desplegadas, el dominio SR (*SR domain*) incluye todas las instancias del protocolo en la red. Sin embargo en algunos despliegues de red se desea subdividir la red en varios dominios SR, la cual en cada subdivisión se incluye uno o más instancias de protocolos. En la figura 15-1 se observa la distribución de una red con segment routing.

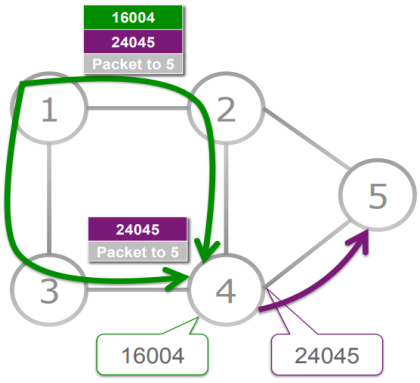


Figura 15-1: Ejemplo de una red con dominio SR

Fuente: ciscolive.com

- *Active Segment:* segmento que es usado por un router receptor que procesa el paquete. En un plano de datos MPLS, es la etiqueta principal. En la figura 15-1 el active segment corresponde al que tiene el valor 16004 en el router 1, cuando llega al router 4, sería 24025.
- *Push:* operación que consiste en la inserción de un segmento en el inicio de una lista de segmento (*segment list*). En SR-MPLS la parte superior de la lista corresponde a la *outerlabel* de la pila de etiquetas.
- *Next:* cuando el *active segment* está completado, se realiza la operación *Next* que consiste en la inspección del siguiente segmento y entonces el siguiente segmento se convierte en el *active segment*. En SR-MPLS la operación *Next* se implementa como una acción *POP* en la parte superior de la etiqueta.
- *Continue:* cuando un *segment active* no se completa, se mantiene como *segment active*. En SR-MPLS la operación *Continue* se implementa con una acción *SWAP* en la parte superior de la etiqueta.
- *SR Global Block (SRGB):* conjunto de segmentos globales en un dominio SR. Si un nodo participa en múltiples dominios SR, existe un SRGB para cada dominio SR. En SR-MPLS, SRGB es de propiedad local de un nodo e identifica el conjunto de etiquetas

locales reservadas para segmentos globales. En SR-MPLS, se usa idénticos SRGB en todos los nodos dentro de un dominio SR. En la figura 16-1 se observa cómo funciona.

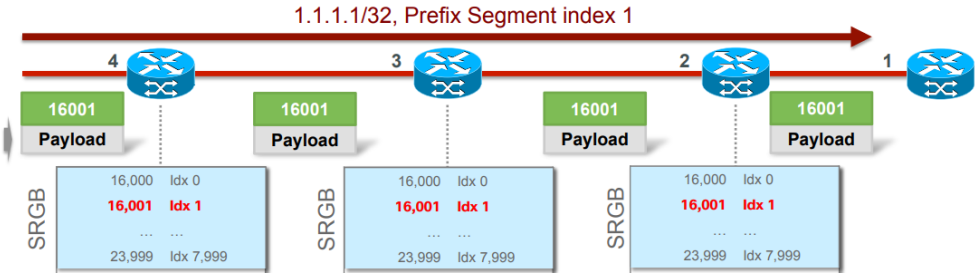


Figura 16-1: SRGB en routers dentro de un dominio SR

Fuente: ciscolive.com

- *SR Local Block (SRLB):* propiedad local de un nodo SR. Si un nodo participa en múltiples dominios SR, hay un SRLB para cada dominio. En SR-MPLS, SRLB es un conjunto de etiquetas locales reservada para segmentos locales.
- *Global Segment:* un segmento que es parte del SRGB de un dominio SR. La instrucción asociada con el segmento está definida en el nivel de dominio SR. El segmento de camino más corto topológico de un destino dado dentro de un dominio SR es un ejemplo de segmento global (*global segment*).
- *Local Segment:* en SR-MPLS corresponde a la etiqueta local *outside* del SRGB. Puede ser parte de un SRLB explícitamente anunciada. La instrucción asociada con el segmento se define a nivel de nodo.
- *IGP Segment:* el nombre genérico que recibe un segmento adjuntado a una parte de la información anunciada por un IGP de estado del enlace.
- *IGP-Prefix Segment:* un segmento IGP que representa los prefijos de un IGP. Cuando un *IGP-Prefix Segment* es de la forma global dentro de una instancia o topología IGP Segment Routing, identifica la instrucción de reenvío de paquete a lo largo de una ruta calculada utilizando el algoritmo de enrutamiento especificado en el campo algoritmo, en la topología y en la instancia IGP donde es anunciado.

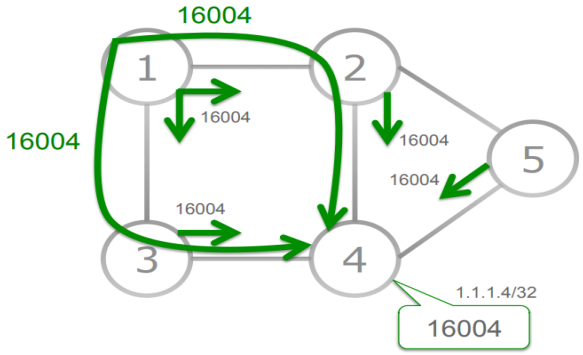


Figura 17-1: Ejemplo de *IGP-Prefix Segment* desde router 1 a router 4

Fuente: ciscolive.com

- *Prefix-SID:* el *SID* de un *IGP-Prefix Segment.* En la figura 18-1 el SID es 16004.
- *IGP-Adjacency Segment:* un segmento adjuntado a una o a un conjunto de adyacencias unidireccionales. Por defecto es un segmento local del nodo que lo anuncia. En la figura 18-1 se observa un ejemplo.

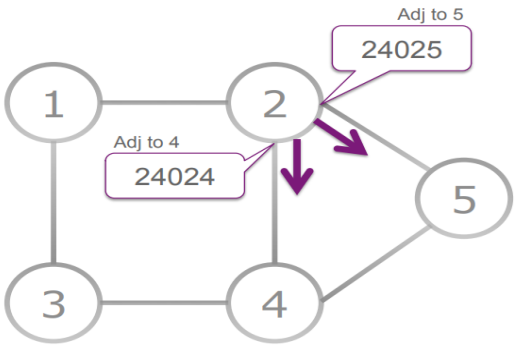


Figura 18-1: *IGP-Adjacency Segment* en router 2 hacia 5 y 4.

Fuente: ciscolive.com

1.5.3 Características de Segment Routing

1.5.3.1 Segmento

Un segmento puede estar asociado con una instrucción topológica, cuando el segmento es local puede dar la instrucción al nodo para que reenvíe el paquete por una interfaz de salida especifica. Un segmento global puede dar la instrucción a todo un dominio *segment routing* (SR) de reenviar el paquete por un camino especifico hasta el destino.

De igual manera un segmento puede estar asociado a una instrucción de servicios por ejemplo cuando un paquete debe ser procesado por un contendor o una máquina virtual, o también a una instrucción asociada a tratamiento de *QoS.*

En un escenario distribuido, los segmentos se asignan y señalizan mediante IS-IS u OSPF. Un nodo puede decidir como dirigir los paquetes con una política de SR, el nodo también es el que puede calcular la política de SR. En un escenario centralizado, los segmentos se asignan y se instancian por un controlador SR, el cual decide que nodos necesitan dirigir qué paquetes en qué políticas de enrutamiento de origen. Este controlador también se encarga de calcular las políticas de enrutamiento de origen. La arquitectura de segment routing no restringe al controlador en la forma de programar la red, tampoco restringe el número de controladores que puede haber en la red ya que varios controladores pueden controlar el mismo dominio SR. En un escenario híbrido se complementa un plano de control distribuido básico con un controlador centralizado. La arquitectura de SR no restringe cómo los nodos que son parte de un plano de control distribuido interactúan con el controlador SR. (Filsfils et al., 2018, pp.3-8)

1.5.3.2 Unión de Segmentos

Una de las características de segment routing es que puede proveer a una red; escalabilidad, red opaca y servicio independiente. Para ello SR utiliza la unión de segmentos (*Binding Segment*). El BSID (*Binding SID*) está atado a políticas de SR, instancia la cual puede involucrar una lista de *SIDs*, cada paquete receptado con un *active segment* igual a BSID son dirigidos sobre las políticas SR a las cuales está atado. Un BSID puede ser un *SID* local o global, si es local el BSID debe ser asignado desde el SRLB, mientras que si es global tiene que ser asignado desde el SRGB.
El uso de BSID permite a la instanciación de políticas ser almacenadas solo en el nodo o nodos que necesita imponer la política. La dirección del tráfico a un nodo que respalda la política solo requiere la imposición del BSID, si la política cambia, significa que solo el nodo que impone la política necesita ser actualizado, los usuarios de esta política no se ven afectados. En la figura x-x se observan dos BSID el 30410 y el 30710 que son un tipo de túnel SR en el cual se incluyen los nodos que participan en dicho binding. (Filsfils et al., 2018, p.21)

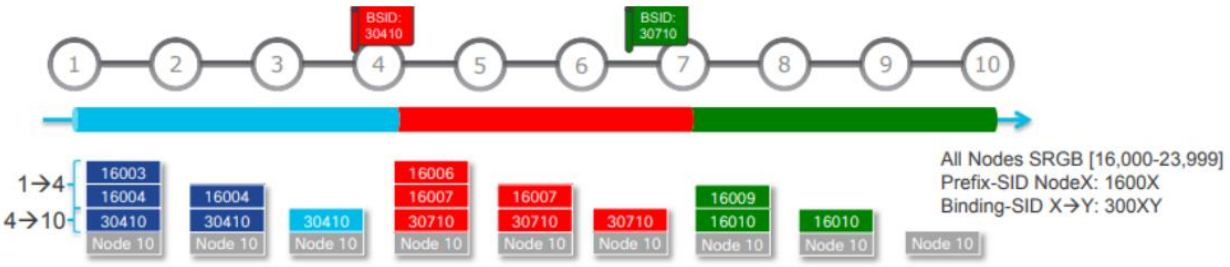


Figura 19-1: Ejemplo de BSID en un dominio SR

Fuente: www.ciscolive.com

1.5.3.3 Control de congestión

Segment routing no introduce nuevos requerimientos para el control de congestión, por defecto el tráfico que se envía se asume que se hace mediante *best effort*. El control se debe implementar en los puntos finales de la red, donde las políticas de SR están en funcionamiento, el ancho de banda asignado puede ser administrado mediante el monitoreo del tráfico entrante que se encuentra asociado a una unión de segmentos BSID el cual identifica la política SR.

1.5.3.4 Protección de red

Segment routing provee *fast reroute protection* (FRR) utilizando algoritmos avanzados de IP FRR llamado *Topology Independent Loop Free Alternate* (TI-LFA). El termino TI, hace referencia a

la habilidad de brindar una ruta de *backup* libre de bulces sin considerar la topología antes del fallo y después del fallo. Por cada destino en la red, TI-LFA prepara un plano de datos *switch-over* para ser activado cuando se detecta el fallo en un enlace sobre el cual se llega a un destino. En el modo de falla de enlace, el destino se protege asumiendo el fallo del enlace. En el modo de protección de nodo, el destino se protege asumiendo que el vecino conectado al enlace primario se ha caído. En el modo de protección de SRLG, el destino se protege asumiendo que el conjunto de enlaces configurados que comparten destino con el enlace primario ha caído. Un router detectar una falla en un enlace o nodo adyacente y rápidamente lleva a cabo una reparación local y utiliza una ruta de *backup* pre calculada, el tiempo para esta acción está dentro de los 50ms. Segment routing y el algoritmo TI-LFA calcula una ruta de *backup* óptima por designación adelantada. El algoritmo identifica cualquier segmento adicional requerido para alcanzar el nodo de tránsito seleccionado a lo largo de una ruta de *backup* para evitar bucles de tráfico durante el proceso de convergencia, el algoritmo debe calcular esta ruta por cada destino cada vez que la topología cambia.

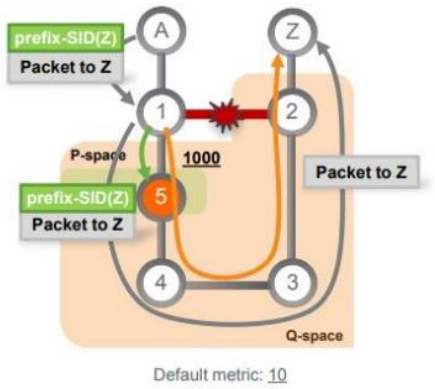


Figura 20-1: Protección del enlace mediante SR Zero-Segment

Fuente: www.ciscolive.com

Se definen los siguientes componentes en una red configurada con esta tecnología.

- P space que hace referencia al conjunto de nodos accesibles desde el nodo origen de un enlace protegido mediante el protocolo SPT (Shortest Paht Tree).
- Q space es el conjunto de nodos alcanzables desde el nodo destino de un enlace protegido mediante SPT.
- P node es un nodo que pertenece al P space.
- Q node es un nodo que pertenece al Q space.
- PQ node es un nodo que pertenece tanto al P space como al Q space y funciona como el nodo de destino del túnel que está protegido.

En la figura 20-1 se observa cómo actúa TI-LFA, refuerza la ruta de post convergencia libre de bucles mediante la codificación de esta como una lista de segmentos. Una ruta post convergencia

es aquella que será usada después de que el IGP ha establecido convergencia después de un fallo. Esta tecnología permite evitar el uso de túneles RSVP-TE en MPLS, tampoco es necesario establecer sesiones TLDP, no es necesario crear estados en la red para que se cumpla una ruta explicita para FRR. (Aviat, 2019, p.6)(Bashandy et al., 2018, pp.3-4)

1.5.3.5 Plano de control

En un dominio SR que tiene un nodo con capacidad SR-IGP anuncia segmentos para sus prefijos y adyacencias. Estos segmentos se denominan segmentos IGP o SID IGP y juegan un papel muy importante en segment routing ya que permiten la expresión de cualquier ruta en todo el dominio SR, dicha ruta se expresa como un solo segmento IGP o una lista de múltiples segmentos IGP. Los anuncios de los segmentos IGP requieren de extensiones en los protocolos de estado del enlace, los únicos protocolos que soportan estas extensiones son IS-IS y OSPF.

Para las extensiones de SR en los IGPs, se definen los *IGP-Prefix Segment* el cual es un segmento IGP unido a un prefijo IGP, este segmento es global a menos que se anuncie lo contrario, dentro de un dominio SR. El contexto de un segmento de prefijo IGP (*IGP-Prefix Segment*) incluye los prefijos, la topología y el algoritmo, múltiples SIDs pueden ser asignadas al mismo prefijo a lo largo de una tupla (prefijo, topología y algorítmo), se mantiene único. Este *IGP-Prefix Segment* identifica la ruta, relacionada al prefijo, calculada según el algoritmo asociado. (Filsfils et al., 2018, p.9)

1.5.4 Extensiones IS-IS para SR

Se definen las codificaciones IS-IS necesarias para los segmentos de prefijos IGP, los segmentos de adyacencia IGP y la unión de segmentos. Para esto al igual que OSPF, se hace uso de los Sub-TLV.

En primer lugar, se definen los identificadores de segmentos denominado *Prefix Segment Identifier (Prefix-SID) Sub-TLV*, el cual lleva los prefijos segment routing SID IGP, denominado *Prefix-SID* el cual debe ser único dentro de un dominio IGP y está asociado a un prefijo anunciado por un nodo y puede ser presentado en varios TLVs con el formato que se muestra en la figura 21-1.

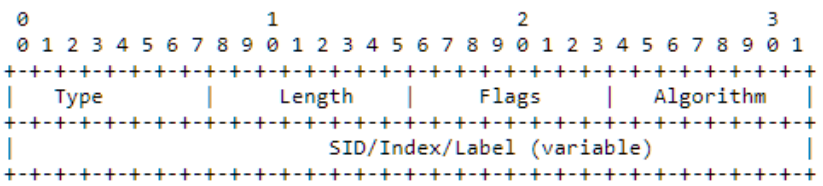


Figura 21-1: Formato del Sub-TLV para el identificador *Prefix-SID*

Fuente: S. Previdi et al., 2019, p.5

En este formato se define un nuevo campo, *Flags*, el cual puede contener las distintas banderas:

- *R-Flag (Re-advertisement Flag):* si se utiliza, entonces el prefijo al cual se adjunta este *Prefix-SID* ha sido propagado por el router desde cualquier otro nivel de IS-IS o mediante la redistribución de rutas.
- *N-Flag (Node-SID Flag):* si se utiliza, entonces el *Prefix-SID* hace referencia al router identificado por el prefijo, normalmente esta bandera se utiliza para *Prefix-SIDs* que están atados a una dirección de loopback de un router. Se hace uso de esta bandera cuando el *Prefix-SID* es un *Node-SID*.
- *P-Flag (No-PHP Flag):* si se utiliza, entonces PHP no tiene que eliminar el *Prefix-SID* antes de entregar el paquete al nodo que ha anunciado el *Prefix-SID.*
- *E-Flag (Explicit Null Flag):* si se utiliza, entonces cualquier vecino *upstream* originador del *Prefix-SID* tiene que sustituir el *Prefix-SID* con un *Prefix-SID* que tiene un valor Explicit Null antes de reenviar el paquete.
- *V-Flag (Value Flag):* si se utiliza, entonces el *Prefix-SID* lleva un valor en lugar de un índice, por defecto esta bandera no se utiliza.
- *L- Flag (Local Flag):* si se utiliza, entonces el valor/índice transportado por el *Prefix-SID* tiene un significado local.

El campo *algorithm* describe los algoritmos utilizados cuando un nodo calcula la accesibilidad a otros nodos o a prefijos relacionados a estos nodos. En este campo del *Prefix-SID* se contiene el identificador del algoritmo que utiliza el router para calcular la accesibilidad del prefijo al que está asociado el *Prefix-SID*. En el origen el campo algoritmo debe tener un valor de 0 o cualquier valor anunciado en el SR-algorithm Sub-TLV.

Otro identificador que se define es el *Adjyacency Segment Identifier (Adj-SID)* Sub-TLV, este identificador transporta las SR *IGP-Adjacency-SID* con banderas y campos que pueden ser utilizados para transportar otros tipos de SIDs. El formato definido para el *Adj-SID* se muestra en la figura 22-1.

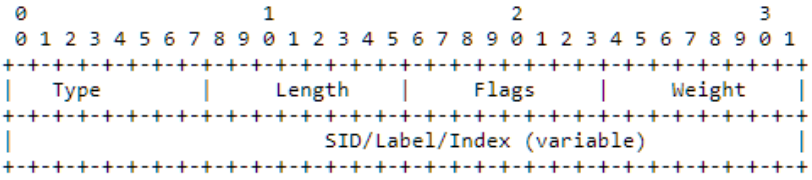


Figura 22-1: Formato del Sub-TLV para el identificador *Adj-SID*

Fuente: S. Previdi et al., 2019, p.9

En el campo *Flags* se definen las siguientes banderas:

- *F-Flag (Address-Family Flag):* si no se utiliza, entonces el identificador *Adj-SID* se está utilizando para reenviar tráfico IPv4 al vecino.
- *B-Flag (Backup Flag):* si se utiliza, el identificador *Adj-SID* es elegible para protección de la red mediante FRR.
- *V-Flag (Value Flag):* si se utiliza, el identificador *Adj-SID* transporta un valor.
- *L-Flag (Local Flag):* si se utiliza, el valor o índice transportado por el identificador tiene un significado local.
- *S-Flag (Set Flag):* cuando se utiliza, indica que el identificador *Adj-SID* hace referencia a un conjunto de adyacencias y por lo tanto pueden ser asignadas a otras adyacencias.
- *P-Flag (Persistent Flag):* cuando se utiliza, indica que el identificador *Adj-SID* se asigna de forma persistente.

El campo *Weight* representa el peso del identificador *Adj-SID* para propósitos de balanceo de carga.
Un router SR puede asignar un identificador *Adj-SID* para cada adyacencia, puede asignar más de un identificador a una adyacencia, y puede asignar el mismo identificador a diferentes adyacencias.

A parte de los identificadores, también se define un *SID/Label Sub-TLV* el cual contiene un SID o una etiqueta MPLS, con el formato que se muestra en la figura 23-1.

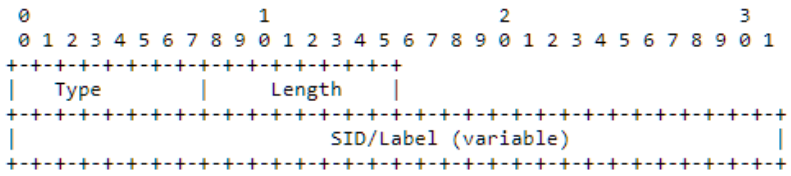


Figura 23-1: Formato del Sub-TLV para *SID/Label*

Fuente: S. Previdi et al., 2019, p.12

Se define también un *TLV SID/Label Binding* el cual puede ser originado por cualquier router en un dominio IS-IS. Este TLV es utilizado para anunciar prefijos para SR *Mapping Server* (SRMS) el cual es útil cuando funcionan internamente tanto SR como LDP, anunciar *Mirror SID* que indican la capacidad de un nodo para procesar trafico que originalmente estaba destinado a otro nodo IGP. El campo *SID/Label* contiene el primer valor del SRGB mientras que el rango contenga el número de elementos SRGB.

Al igual que en las extensiones de OSPF, en IS-IS también se definen capacidades que debe tener un router, ya que SR requiere que cada router anuncie la capacidad de su plano de datos SR y el rango de valores de etiquetas que utiliza MPLS para SR en el caso donde los SIDs globales son asignados. Las capacidades del plano de datos y el rango de etiquetas son anunciadas utilizando un nuevo Sub-TLV. Se definen los mismos Sub-TLV que en OSPF (SR-Algorithm, SR Local Block) pero para dominios IS-IS. (S. Previdi et al., 2019, pp.5-18)

1.5.5 OSPF como plano de control de SR

1.5.5.1 Introducción

Segment Routing utiliza OSPF para anunciar la información de la topología, información de prefijo, un bloque global de enrutamiento de segmento SRGB y la información de la etiqueta asignada por MPLS. Para cumplir con todas estas funciones, OSPF implementa extensiones en su TLV. OSPF define los protocolos para SR en sub-TLV que habilitan capacidades SID y NE SR.

Segment routing necesita de ciertas capacidades para anunciar la información de un router hacia otro en la misma área. Estas capacidades son advertidas entre los routers mediante el Router Information Opaque LSA de OSPF. Los TLVs aplicados son SR-Algorithm TLV (8) y SID/Label Range TLV (9), la función de estas capacidades se muestra en la tabla 4-1.

Tabla 4-1: Capacidades OSPF para SR

Tipo	**Contenido**	**Función**	**Posición**
TLV (8)	SR-Algoritmo TLV	Publica el algoritmo utilizado	Type 10 Opaque LSA
	SID/ Rango de etiquetas	Anuncia el alcance SR SID o SRGB	Type 10 Opaque LSA
	TLV de bloque local SR	Anuncia el alcance de la etiqueta que un vecino reserva para el SID local	Type 10 Opaque LSA
	TLV de preferencia SRMS	Anuncia la prioridad del servidor de mapeo SR con el que funciona un vecino local	Type 10 Opaque LSA
Sub-TLV (9)	SID/ Etiqueta sub-TLV	Anuncia etiqueta SR SID o MPLS	SID/ Rango de etiquetas TLV TLV de bloque local SR

	Prefijo SID sub-TLV	Anuncia el prefijo SR SID	Prefijo extendido OSPFv2 Opaco TLV de prefijo extendido OSPFv2 de LSA y TLV de rango de prefijo extendido OSPF
	Adj-SID Sub-TLV	Anuncia el prefijo SR SID	OSPFv2 Extended Link Opaco LSA OSPFv2 Extended Link TLV
	LAN Adj-SID Sub-TLV	Anuncia los SID de adyacencia SR en una LAN	OSPFv2 Extended Link Opaco LSA OSPFv2 Extended Link TLV

Fuente: Huawei, 2021

Realizado por: Loza, L. 2021

1.5.5.2 Tipos de LSA

OSPF utiliza la LSDB y la llena con los LSA (*Link State Advertisement*) los cuales van dentro del LSU, existen múltiples tipos de LSA, los cuales son:

- *Type 1 – Router LSA*

Los paquetes se envían entre los routers que se encuentran dentro de la misma área y que no saldrán de la misma. Se utiliza este LSA para enviar información sobre las interfaces de un router OSPF pero también para llevar la información sobre sus vecinos para tener adyacencia dentro del área, el algoritmo SPF se basa en este LSA para elegir las rutas.

- *Type 2 – Network LSA*

Este LSA es generado por el router designado (DR) para informar de todos los routers que tienen adyacencia con él, se envían dentro de la misma área a todos los vecinos.

- *Type 3 – Summary LSA*

Generados por el ABR para sumarizar el área directamente conectada, y anunciar la información intra área a los routers de otras áreas a las cuales el ABR se encuentra conectado, utilizando los prefijos sumarizados.

- *Type 4 – Summary ASBR LSA*

Encargado de anunciar la presencia de un ASBR a las otras áreas a las cuales está conectado, advierte a los routers de la red cómo llegar a él.

- *Type 5 – OSPF ASBR External LSA*

Son generados por el ASBR para anunciar las rutas externas redistribuidas en un sistema autónomo que ejecuta OSPF.

- *Type 7 – NSSA External LSA*

Creadas por el ASBR dentro de un área *Not So Stubby Area* ya que no utiliza los LSA tipo.

- *Type 8 – Link LSA*

Anuncia la dirección local de enlace de origen propio a todos los demás routers conectados a esos enlaces que son similares al Router LSA. En la tabla 5-1 se especifican los LSA para SR y lo que transportan.

Tabla 5-1: LSAs para Segment Routing

LSA Type	Función
Type 4	Transporta en el TLV el algoritmo de Segment Routing Transporta en el TLV el rango SID/Label
Type 7	Prefijo extendido en el Opaque LSA (7)
Type 8	Enlace extendido en el Opaque LSA (8) Transporta en el sub TLV el Adj-SID y el LAN Adj-SID

Fuente: Psenak et al., 2019

Realizado por: Loza, J. 2021

1.5.5.3 Formato del paquete OSPF

Los 5 paquetes mencionados en la sección 1.5.4.3 tienen una cabecera estándar de 24 bytes. Esta cabecera contiene toda la información necesaria para determinar cuando un paquete debe ser aceptado para su posterior procesamiento.

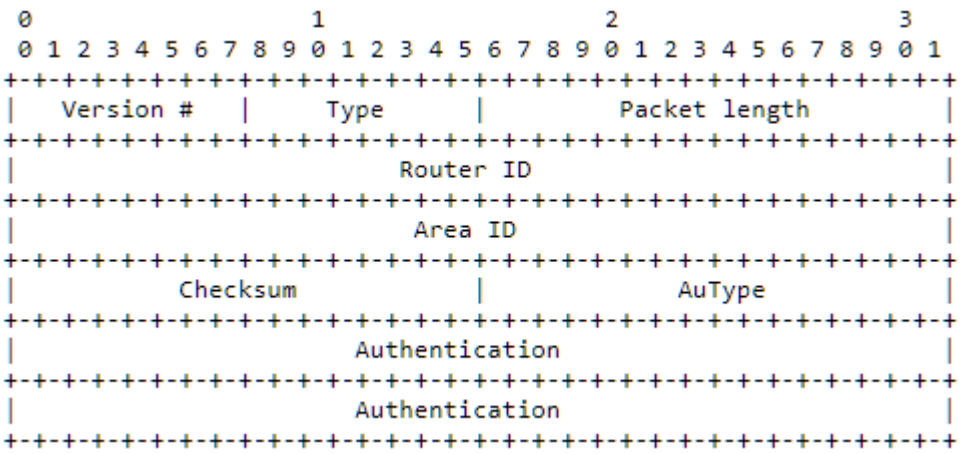


Figura 24-1: Formato de la cabecera del paquete OSPF

Fuente: Moy, 1998, p.190

El campo *Version* indica la versión de OSPF, el campo *Type* indica el tipo de paquete OSPF, el campo *Packet length* hace referencia a la longitud del mensaje en bytes, a esta longitud se le debe sumar los 24 bytes de la cabecera, el campo *Router ID* contiene la información correspondiente a la identificación del router que ha generado este mensaje, normalmente es la dirección IP de la interfaz desde la que sale el mensaje. El campo *Area ID* identifica el área a la cual pertenece este mensaje, si no se han configurado áreas este campo tiene un valor de 0. Campo *Checksum* para comprobar si existe algún error en el mensaje. El capo *AuType* indica el tipo de autenticación que utiliza el mensaje, si tiene el valor de 0 no existe autenticación, un valor de 1 indica autenticación con una contraseña simple mientras que un valor de 2 indica autenticación criptográfica. El último campo *Authentication* de 64 bits es utilizado para la autenticación del mensaje en caso de que exista.

1.5.5.4 Extensiones OSPF

Se hace uso del prefijo/enlace extendido *Opaque* LSA para anunciar los diferentes SID de segment routing. Este LSA define un nivel superior de TLVs para OSPF. Se utilizan Sub-TLV los cuales existen dentro de un TLV y se utilizan para agregar información adicional, cada Sub-TLV consta de tres campos, uno del campo *Type, Length* y *Value.* El campo *Type* indica el tipo de elementos en el campo *Value*, el campo *Length* indica la longitud del campo *Value*.
Estos Sub-TLVs denominado *SID/Label*, en el campo *Value* llevan la información de los segmentos anunciando el SID o etiqueta asociado a un prefijo o adyacencia, como se puede ver en la figura 24-1. Si la longitud de tiene un valor de 3, representa una etiqueta, mientras que el valor 4 representa a un SID.

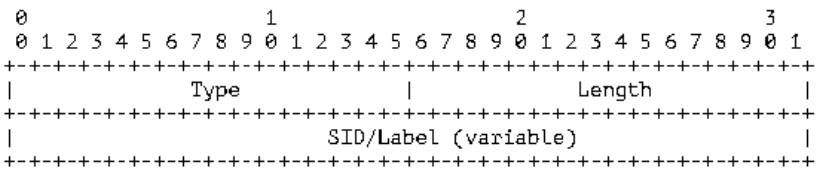


Figura 25-1: Formato del Sub-TLV para la extensión de SR en OSPF.

Fuente: Psenak et al., 2019, p.5

Segment routing requiere de capacidades adicionales en el router para anunciarse a otros routers en el área. Una de ellas es el algoritmo SR TLV, el cual es un nivel superior de TLV de un *Router Information Opaque LSA.* Este algoritmo es opcional y solo debe ser anunciado una vez en el *Router Information Opaque LSA.* Si el algoritmo no es anunciado por el nodo, este se considera que no tiene capacidades para SR. Un router SR puede utilizar varios algoritmos al calcular accesibilidad a los routers OSPF o a los prefijos en un área OSPF, por ello se utiliza el algoritmo

SR TLV para anunciar los algoritmos que se está utilizando un router hacia otro router de un área OSPF, esta información va en el campo *Value* en el Sub-TLV como se puede ver en la figura 26-1. Los algoritmos que puede ejecutar un router es *SPF* y *Strict SPF*, la diferencia recae en que el segundo algoritmo ordena a todos los nodos de una ruta que respeten la decisión de enrutamiento de *SPF* por lo que no se pueden alterar las rutas calculadas por este algoritmo.

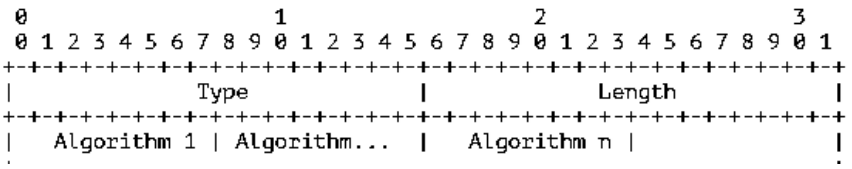


Figura 26-1: Formato del Sub-TLV para describir algoritmos.

Fuente: Psenak et al., 2019, p.6

El rango de TLV *SID/Label*, los prefijos pueden ser anunciados con la forma de un índice el cual define el desplazamiento en el espacio *SID/Label* anunciado por el router en el Sub-TLV como se observa en la figura 27-1.

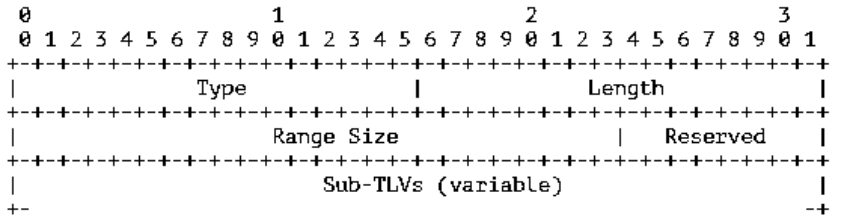


Figura 27-1: Formato del Sub-TLV para describir el rango SID/Label

Fuente: Psenak et al., 2019, p.7

En principio los Sub-TLV solo soportan los *SID/Label*, el cual tiene que ser incluido en el rango de TLV ya que es utilizado para representar el primer *SID/Label* en el rango anunciado. Solo un *SID/Label* puede ser anunciado en el rango TLV *SID/Label*, si más de un *SID/Label* TLV es presentado, el rango *SID/Label* tiene que ser ignorado. Cuando múltiples rangos *SID/Label* pueden anunciarse para anunciar múltiples rangos, en ese caso se realiza las siguientes acciones:

- El router de origen tiene que codificar cada rango en un rango TLV *SID/Label*.
- El router de origen decide cada orden en la cual el conjunto de rangos TLV *SID/Label* son anunciados dentro del *Router Information Opaque LSA*. El router de origen debe asegurarse de que el orden sea el mismo después de un reinicio exitoso para asegurar que el rango *SID/Label* y los índices correspondientes estén preservados en los reinicios.
- El router receptor tiene que adherirse al orden en el que los rangos son anunciados cuando se calcula un *SID/Label* desde un índice *SID*.

- El router de origen no tiene que anunciar rangos sobrepuestos.
- Cuando un router recibe múltiples rangos sobrepuestos, tiene que ajustarse al proceso de SR. Cuando un router de origen anuncia múltiples rangos, el router receptor debe concatenar los rangos y construir el SRGB.

Los TLV SR *Local Block* (TLV SRLB) contienen el rango de etiquetas que el nodo ha reservado para SIDs locales. Los SID del SRLB pueden ser usados para las adyacencias SID, pero también por componentes distintos del protocolo OSPF como cuando un controlador puede dar la instrucción a un router para asignar una específico SID local. Algunos controladores o aplicaciones pueden usar el plano de control para descubrir todo el conjunto de SIDs locales en un router en particular, en este caso el SRLB es anunciado en el plano de control, y la forma en la que es anunciado es mediante los TLV SRLB. (Psenak et al., 2019, pp.5-8)

1.5.6 SRV6 como plano de datos

Segment Routing puede transportarse mediante IPv6 como plano de datos, se basa en una cabecera de IPv6 para SR conocida como SRH, en la figura 28-1 se muestra el stack del protocolo.

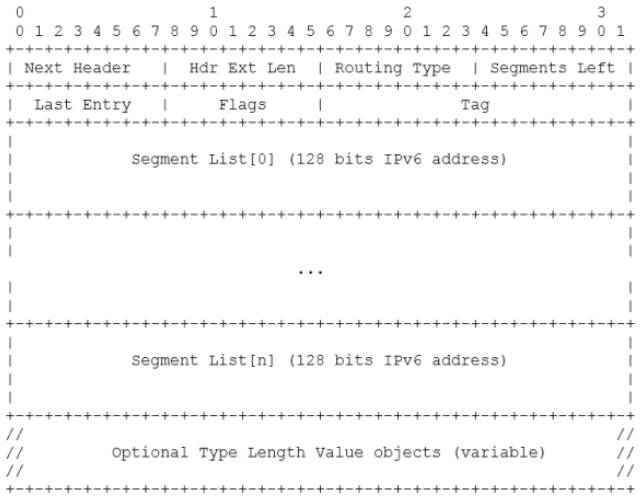


Figura 28-1: Stack del protocolo SRH IPv6

Fuente: Filsfils et al., 2020, pp.5

Tabla 5-1: Campos del SHR

Campo	Significado
Next Header	Selector de 8 bits que identifica el tipo de cabecera inicial de la parte fragmentable del paquete original.
Hdr Ext Len	Un entero de 8 bits que indica la longitud de la cabecera de enrutamiento en unidades de 8 octetos.
Routing Type	Identificador de 8 bits de una variante particular de una cabecera de enrutamiento.

Segments Left	Identificador de 8 bits utilizado para numerar los tramos de rutas restantes.
Last entry	Contiene el índice, dentro la lista de segmentos, del último elemento de la lista de segmentos.
Flags	Banderas de 8 bits, 0 cuando se transmite e ignorada cuando se recibe.
Tag	Etiqueta los paquetes como parte de una clase o grupo de paquetes, en base a propiedades en común.
Segment List	Dirección IPv6 de 128 bits que representa n segmentos en la lista de segmentos. La lista de segmentos está codificada empezando desde el último segmento de la política de SR.
TLV	Campo utilizado para transportar metadatos para el procesamiento de segmentos.

Fuente: Filsfils et al., 2020, pp.5-6

Realizado por: Loza, J. 2021

En segment routing existen varios tipos de nodos, el nodo origen que es el que origina el paquete con un segmento en la dirección de destino de la cabecera IPv6 como se observa en la figura 29-1. Nodo de tránsito que reenvían paquetes destinados a un segmento destino como se observa en la figura x-x. Nodo de destino el cual procesa un segmento local en la dirección de destino de la SRH como se observa en la figura 29-1.

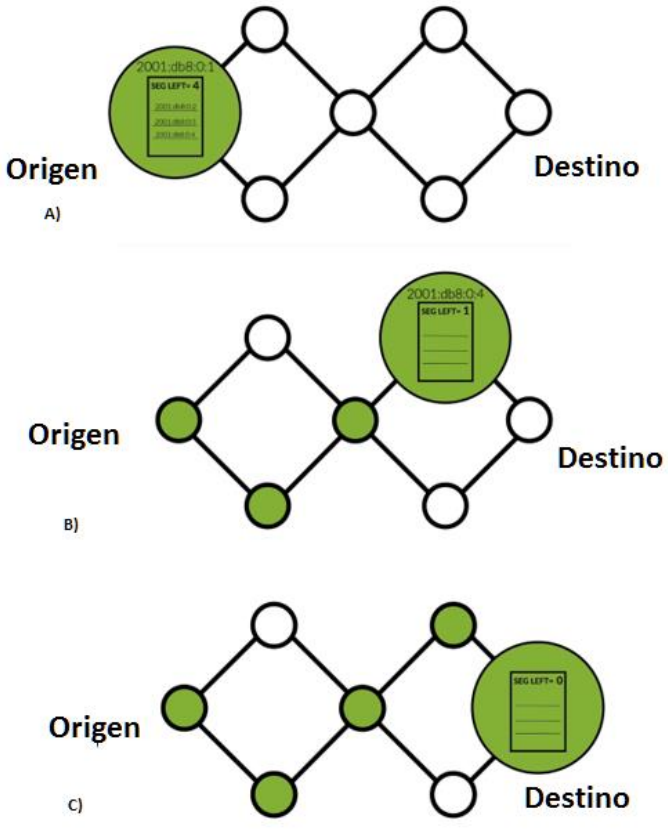


Figura 29-1: A) Nodo origen B) Nodo de tránsito C) Nodo destino

Fuente: Loza, J. 2021

Procesamiento de los segmentos en SRv6, consiste en que el nodo de origen dirige un paquete en una política SR. Si en la política se tiene una lista de segmentos con un solo segmento, y no es necesario añadir información a la bandera o TLV de SRH.

El único nodo de tránsito que tiene permitido inspeccionar el SRH es el nodo que corresponde a la dirección de destino del paquete, cualquier otro nodo de tránsito no puede inspeccionar el SHR y tiene que reenviar el paquete a través de la dirección de destino de acuerdo a su tabla de enrutamiento. Cuando un SID está en la dirección de destino de una cabecera IPv6 de un paquete, se enruta mediante una red IPv6 como una dirección IPv6.

El nodo de destino crea la FIB para entradas locales de SIDs. Una entrada en la FIB puede representar una instancia SID SRv6, puede representar una interfaz local o una ruta no local. En la figura 30-1 se puede observar como se encapsula un paquete cuando entra a un dominio SRv6. (Filsfils et al., 2020)

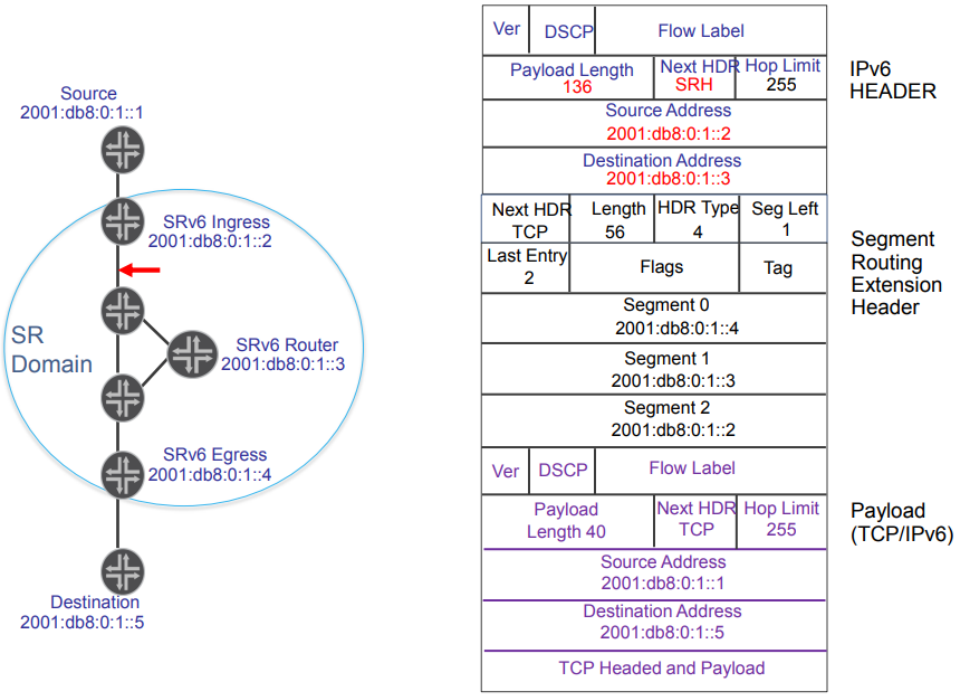


Figura 30-1: Paquete dentro de un dominio SRv6

Fuente: J.Horn, 2016, p.22

El segmento está formado por dos partes como se observa en la figura 31-1, la primera parte el *Locator* son los bits más significativos utilizados para enrutar el segmento a su nodo padre, la parte *Function* son los bits menos significativos e identifican la acción que se va a realizar sobre el nodo padre.

Figura 31-1: Formato de un segmento SRv6

Fuente: Farrel, 2017, p15.

1.5.7 MPLS como plano de datos

1.5.7.1 Reenvío de prefijos

Ya que SR permite múltiples SID para el mismo prefijo, es posible tener múltiples comportamientos en el reenvío para un mismo destino, al instaurar SR sobre el plano de datos MPLS este principio encaja sin problemas. Un operador puede asignar múltiples etiquetas MPLS o índices a un mismo prefijo y asignar diferentes comportamientos de reenvío a cada etiqueta/SID las cuales son descargadas en la red por el MCC (MPLS Control Plane Client) que es el cliente de plano de control MPLS, en la FIB para diferentes comportamientos de reenvío. El MCC en la entrada del dominio SR o cualquier punto del dominio, puede elegir aplicar un comportamiento de reenvío particular a un paquete en particular, aplicando la acción PUSH a dicho paquete utilizando el correspondiente SID.

Los múltiples comportamientos de reenvío para un mismo prefijo puede ser rutas diferentes, diferente ECMP o UCMP. En la figura 32-1 se muestran las acciones que se realizan para el reenvío de etiquetas.

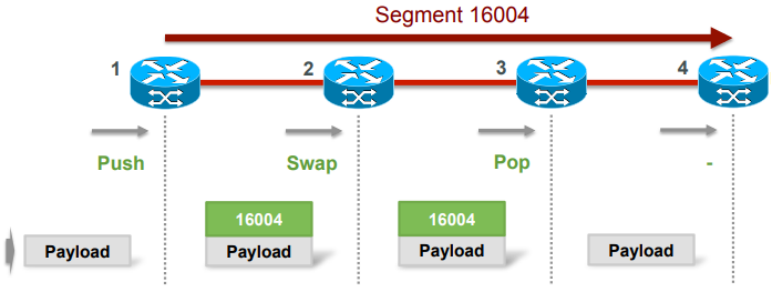


Figura 32-1: Reenvío de etiquetas SR-MPLS

Fuente: Filsfils y Michielsen, 2020, p.6

1.5.7.2 Representación del Segment ID

El SID se representa mediante una etiqueta MPLS o un índice. Un SID global es una etiqueta o un índice que puede ser asignado a una etiqueta MPLS dentro del SRGB del nodo que instala el SID global en su FIB y recibe el paquete etiquetado. El MCC tiene que asegurar que cualquier valor de etiqueta correspondiente a cualquier SID que instale en el plano de datos, para ello se debe cumplir que; el valor de etiqueta tiene que ser único dentro del router en el cual corre el

MCC, es decir, una etiqueta tiene que ser utilizada para representar un solo SID, no tiene que ser utilizada para representar mas de un SID u otra forma de reenvío en el router. Y además el valor de etiquete no tiene que pertenecer a un rango de etiquetas destinadas a propósitos especiales.

1.5.7.3 SRGB y SRLB

La representación del SRGB en una lista de rangos MPLS sigue las reglas:

- La lista de rangos que comprende el SRGB no tiene que sobreponerse.
- Cada rango en la lista que hace referencia al SRGB no tiene que cubrir o sobreponerse a valores o rangos de etiquetas reservados.
- Si el SRGB de un nodo no se ajusta a la estructura especificada o a las reglas anteriores, el SRGB tiene que ser ignorado por todos los routers del dominio y el nodo tiene que ser tratado como si no tuviese un SRGB.
- La lista de rangos de etiquetas tiene que ser utilizada solo para instanciar un SID global en el plano de datos MPLS.

Un segmento local puede ser asignado desde el SRLB o desde cualquier etiqueta que no ha sido utilizada mientras que no se use una etiqueta con un propósito especial. El SRLB consiste en el rango de etiquetas locales reservadas por un nodo para ciertos segmentos locales. En la figura 18-1 se observa el rango de etiquetas para el SRGB.

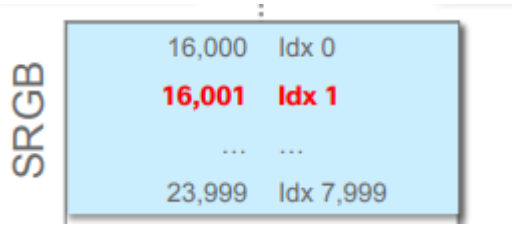


Figura 33-1: Rango de etiquetas e índices para SRGB

Fuente: Filsfils y Michielsen, 2020, p.10

1.5.7.4 Asignación de un índice SID en la etiqueta MPLS

Se debe calcular el valor de la etiqueta MPLS a partir del índice de un SID. El valor del índice es determinado por el MCC como puede ser IS-IS u OSPF, la etiqueta calculada se descarga a la FIB o se envía junto a un paquete, o ambas a la vez.

1.5.7.5 Etiquetas MPLS para SID global y local

La etiqueta que corresponde al SID global "*Si"* la cual es representada por el índice global *"I"* y es descargada a la FIB, es utilizada para hacer coincidir paquetes cuyo *active segment* es "*Si"*. Para SIDs locales el MCC es responsable de descargar el valor correcto de etiqueta a la FIB, por ejemplo, un IGP con extensiones SR descargada la etiqueta MPLS correspondiente a un *Ajd-SID*. En la figura 34-1 se observa la cabecera que tendría un paquete encapsulado en SR-MPLS

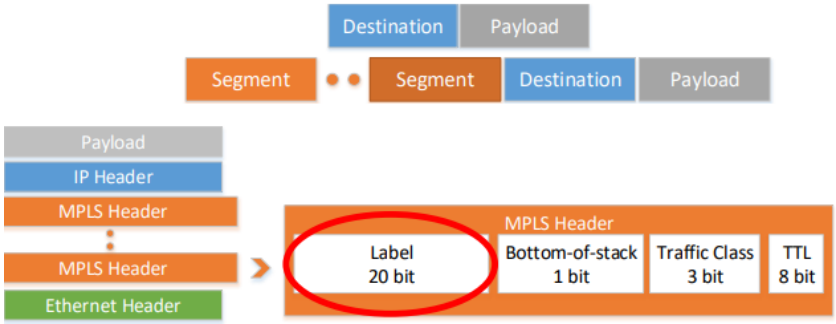


Figura 34-1: Paquete encapsulado en SR-MPLS

Fuente: At, 2020, p.8

1.5.7.6 Reenvío de SIDs globales

Para el reenvío de los SIDs se necesita implementar las operaciones PUSH, CONTINUE y NEXT de segment routing en el plano de datos MPLS.

Si un MCC determina que antes de enviar un paquete a un vecino, la operación PUSH o CONTINUE se debe aplicar a un paquete entrante relacionado con el SID global *"Si"*, el mismo que es representado por el índice global *"I"* y pertenece al router relacionado a este índice. El vecino que recibe el paquete puede estar directamente conectado al router que corre el MCC ya sea por una interfaz física o virtual. El método por el cual el MCC determina si la operación PUSH o CONTINUE debe ser aplicada utilizando el SID *"Si"* es el caso en el que IS-IS recibe un prefijo SID Sub-TLV.

La operación NEXT es utilizada para eliminar la etiqueta más alta. Siguiendo los siguientes pasos:

- Elimina la etiqueta más alta.
- Aplicar la instrucción asociada con la etiqueta entrante que ha sido eliminada.

La acción en el paquete después de eliminar la etiqueta más alta depende de la instrucción asociada con la etiqueta entrante, así como el contenido del paquete que se encuentra debajo de la etiqueta superior la cual fue eliminada.

1.5.7.7 Reenvío de SIDs locales

De la misma forma se implementan las operaciones PUSH, CONTINUE y NEXT. Cuando el MCC determina que la operación PUSH debe ser aplicada a un paquete entrante utilizando el SID local "*Si"* antes de enviar el paquete a un vecino el cual puede estar conectado por una interfaz virtual o física. Un SID local puede ser el anuncio y asignación de *Adj-SID* por IS-IS, un ejemplo puede ser la protección contra fallos utilizando TI-LFA. Un SID local se especifica mediante una etiqueta MPLS, la operación PUSH para un SID local es idéntico a la operación PUSH utilizado por MPLS, la acción de reenvío después de añadir la etiqueta MPLS correspondiente al SID local es determinada por el MCC. Por ejemplo, en el caso de que la operación PUSH fuese realizada para reenviar un paquete sobre una ruta de backup calculada por TI-LFA, la acción de reenvío puede ser enviar el paquete a un cierto vecino que continuará reenviando el paquete sobre la ruta de backup.

Un SID local en un router corresponde a una etiqueta local, en este escenario la etiqueta saliente hacia el siguiente salto se determina por el MCC que corre en este router y el comportamiento de reenvío para la operación CONTINUE es idéntica a la operación SWAP de MPLS. La operación NEXT para SIDs locales es idéntico a la operación NEXT para los SIDs globales. (Bashandy et al., 2019, pp.4-15)

1.6 Estado del arte redes con SDN y Segment Routing

1.6.1 Casos aplicativos

1.6.1.1 BELL CANADÁ SDN-SR

BELL Canadá es un operador de capa 1, su proyecto de transformación de red es conocido como la Red 3.0, y su objetivo es entregar la mejor experiencia a sus usuarios a través del impulso de nuevo software y tecnologías en la nube.

Los requerimientos de la red de siguiente generación son los siguientes:

- Debe ser un estándar de la industria, ratificado por organizaciones de estándares globales.
- Reutilizable en el núcleo/WAN como punto de unión para todas las redes.
- Programación de redes.
- Provee soluciones para la transición y el estado final.
- Interoperabilidad con brownfield y greenfield
- Manejo implícito de ECMP

Para Bell Canadá, Segment Routing cumple con los requerimientos de las redes de siguiente generación, y está siendo adoptado como parte de una fase de la transformación para la Red 3.0. El objetivo final del operador es una combinación de Segment Routing y control SDN centralizado, para habilitar ingeniería de tráfico bajo demanda de un extremo a otro en sus redes, desde un centro de datos a la WAN, y desde la red de acceso al CORE.
La ingeniería de tráfico bajo demanda proporciona dos beneficios clave:

- Aumento de la utilización de la red: solo con agregar capacidad a la red ya no es suficiente para satisfacer las demandas del tráfico, ya que los ingresos derivados por bit no pueden cubrir los costes por bit. También es necesario de mayor eficiencia para que los operadores puedan aprovechar al máximo cada enlace. Bell Canadá lo denomina como la habilidad "sleeping capex".
- Simplificar drásticamente las operaciones de red: la ingeniería de tráfico en las redes con MPLS es manual, compleja y consume mucho tiempo. Segment Routing simplifica la función de TE (Traffic Engineer) a la vez que elimina meses de planificación inicial. Los resultados es una reducción del opex y mayor agilidad de la red, lo cual está alineado con lo que la nube y la virtualización requiere.

En la figura 35-1 se puede observar la arquitectura de red Bell Canadá, en donde la red de CORE tiene implementado Segment Routing sobre ISIS y SDN mediante PCEP.

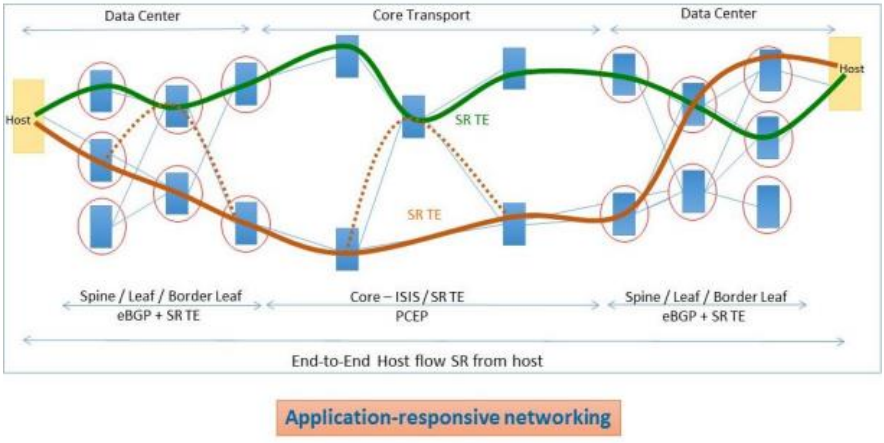


Figura 35-1: Arquitectura de red con SR y SDN de Bell Canadá

Fuente: Bell Industries, 2017

De acuerdo con los requerimientos de la red de la siguiente generación, para mantener tanto el *greenfield* como el *brownfield* de la red, Bell Canadá está adoptando un enfoque por fases para Segment Routing, con los siguientes pasos:

1. Retener la red MPLS distribuida, pero comenzar a agregar Segment Routing, por lo que la arquitectura, la ingeniería y los equipos de operaciones pueden empezar a utilizar esta tecnología.
2. Empezar a introducir un control centralizado por medio de PCEP en ciertas partes aisladas de la red, para comenzar a testear las características de prueba.
3. Implementar Segment Routing en toda la infraestructura de Bell Canadá.

Los últimos pasos son a largo plazo. Con Segment Routing establecido en la red, el operador podrá simplificar la pila de protocolo al eliminar las etiquetas de LDP y RSVP-TE. (Perrin, 2017, pp. 7-8)

1.6.1.2 Soluciones de Aptira

De la misma forma hay empresas que ofrecen soluciones con Segment Routing y SDN, como es el caso de Aptira, el cual despliega SR sobre MPLS y para integrar SDN lo hace mediante el controlador OpenDaylight, para automatizar la red utiliza un servicio de orquestación, en este caso es Cloudify. (Aptira, 2021)

Empresas como Cisco y Juniper ofrecen equipos para el despliegue de redes con SR, de la misma forma esta tecnología tiene el apoyo de toda la industria ya que colaboran los operadores lideres y el grupo IETF. Segment Routing es una arquitectura novedosa, la cual fue diseñada con SDN en mente. El objetivo principal es separar los planos de control y de datos para simplificar la red. SDN es una idea que se empieza a implementar en las redes con la idea de simplificarlas conjuntamente con SR.

1.6.1.3 Despliegue experimental en Latinoamérica SND-IP (AmLight)

AmLigth ha realizado el experimento de una red global definida por software, esta red se implementó junto con 13 NRENs y universidades de todo el mundo como se observa en la figura 18-1. El objetivo de este estudio fue proveer conectividad punto a punto de capa 3 sin utilizar routers legacy, transformar los sistemas autónomos que corren OpenFlow en una red de tránsito IP/BGP y proveer una migración factible de las redes tradicionales IP/BGP hacia redes SDN/OpenFlow.

Figura 36-1: Red desplegada SDN/IP mediante el controlador ONOS

Fuente: Freitas et al., 2016, p.7

Para el desarrollo de este experimento AmLight utiliza el controlador ONOS el cual es un sistema operativo gratuito y de código abierto. Además AmLight es uno de los colaboradores del proyecto ONOS desde 2015.

La red desplegada en Latinoamérica se observa en la figura 37-1, y tiene el objetivo de demostrar que ONOS puede trabajar en una red real y su alto desempeño, disponibilidad y escalabilidad cumple con los más altos requerimientos de los operadores de red.

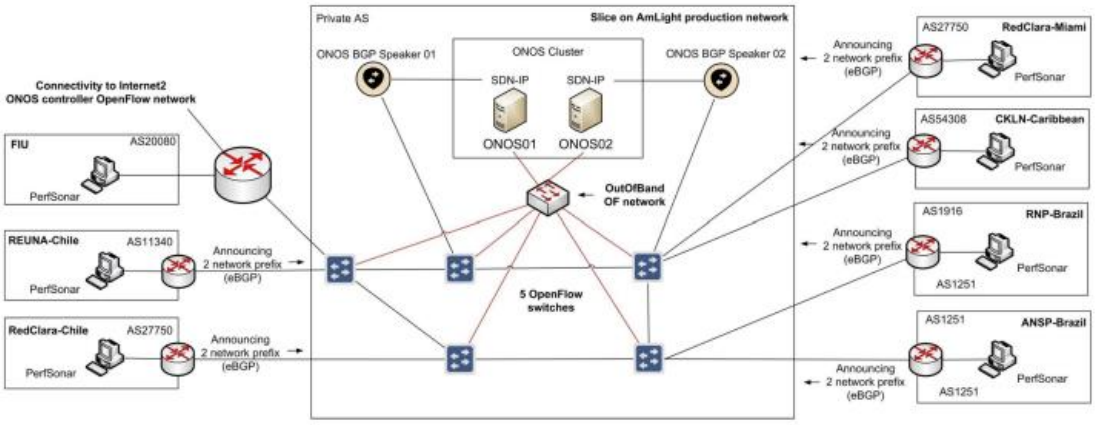


Figura 37-1: Diseño de la red con SDN/IP en Latinoamérica

Fuente: Freitas et al., 2016, p.8

En 2016 AmLight continuó el estudio con la participación de más instituciones que colaboraron, debido a esto implementaron diferentes características a la red como soporte *Pipeline Multi-table*, QoS e IPv6, de igual forma se desarrolló la implementación de un servicio LAN privado mediante ONOS, denominado VPLS. (Freitas et al., 2016)

1.6.2 Casos de estudio

1.6.2.1 Centro de Datos basado en SDN y Segment Routing

En este estudio realizado por Pang, Xu y Fu, analizan mediante simulación se la combinación del protocolo MPTCP y Segment Routing para gestionar el tráfico y limitar los requisitos de almacenamiento. Y el diseño de una red basada en SDN para un ambiente de una red centro de datos. El diseño de la red se muestra en la figura 38-1.

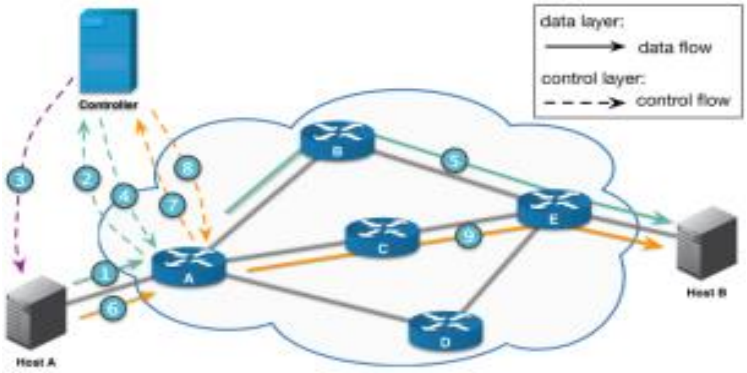


Figura 38-1: Estructura de la red SDN MTCP-SR

Fuente: Pang, Xu y Fu, 2017, p.**7**

El objetivo del estudio es comparar el rendimiento de la configuración propuesta MTPC-SR en comparación con otros protocolos como TCP y MTCP. Para observar el rendimiento de la red con cada protocolo, desde el HOST A se envían paquetes al HOST B los cuales son medidos y los resultados se muestran en la figura 39-1.

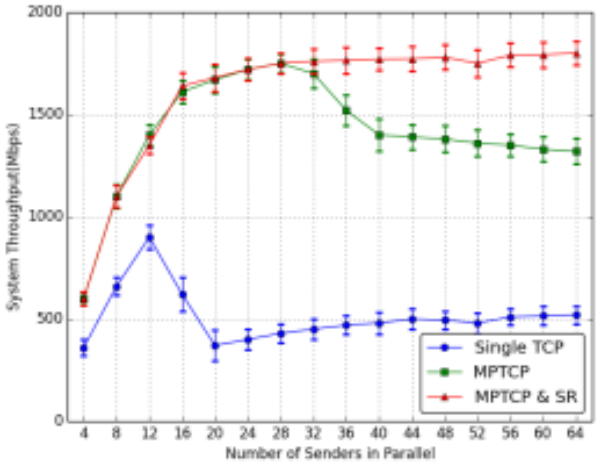


Figura 39-1: Comparación del Throughput vs Paquetes enviados

Fuente: Pang, Xu y Fu, 2017, p.8

Es apreciable que a medida que la red se congestiona más, el protocolo MTCP-SR presenta mayor rendimiento respecto a los otros protocolos, por lo que se observa que es mejor la configuración con SR.

En la red la capa de control está basada en un controlador SDN para recolectar la información, realizar el cálculo de las rutas, generar las rutas Segment Routing y la gestión de los flujos, mediante el cual se graba el estado del flujo y es el responsable de la asignación de rutas en la topología actual. (Pang, Xu y Fu, 2017)

1.6.2.2 Ingeniería de tráfico con SR

En el estudio realizado por Davoli et al, diseñan la arquitectura de una red basada en SDN y la implementación del código abierto Luca para una red con ingeniería de tráfico mediante Segment Routing. Tienen como objetivo demostrar que a raíz de las redes diseñadas por software, la ingeniería de tráfico va a ser más beneficiosa. Segment Routing (SR) es un enfoque emergente para el enrutamiento que puede simplificar la aplicación de la ruta delegando toda la configuración y el estado por flujo en el borde de la red. En este trabajo proponen una arquitectura que integra el paradigma SDN con TE basado en SR, para lo cual proporcionan una implementación de referencia de código abierto. En la figura 40-1 se muestran los resultados.

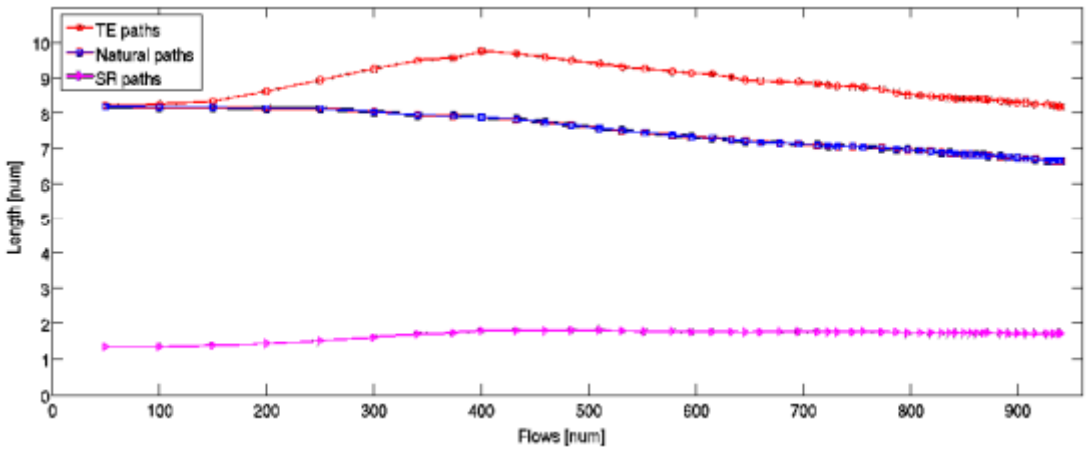


Figura 40-1: Comparación entre las rutas SR, TE y naturales.

Fuente: Davoli et al, 2015, p.7

Como se puede observar al incrementar el número de flujos (carga de tráfico) la longitud media de las rutas TE aumenta de 8,2 a 9,7. Cuando la carga aumenta aún más, la longitud media comienza a disminuir.

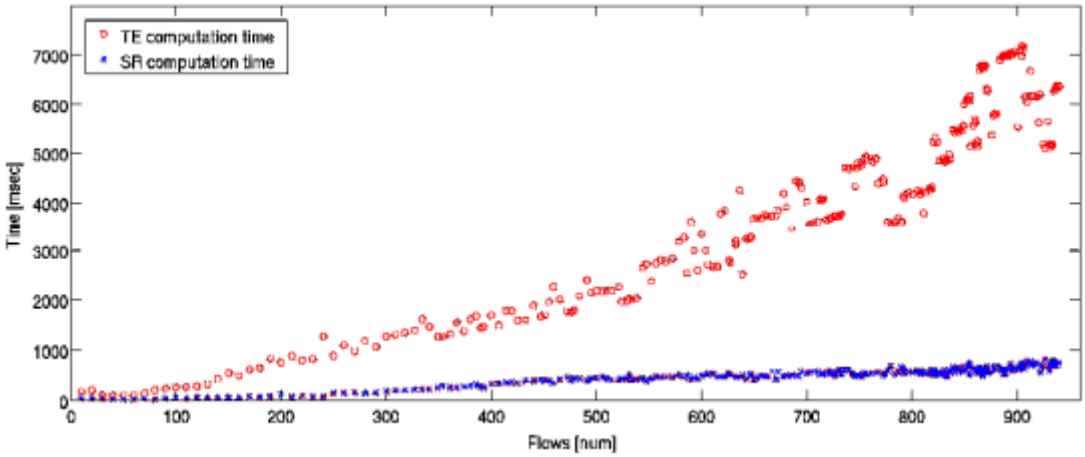


Figura 41-1: Comparación del tiempo de procesamiento vs carga en la red

Fuente: Davoli et al, 2015, p.7

El procesamiento de SR es significativamente menor, esto se evidencia cuando la carga de tráfico en la red empieza a incrementar de forma considerable.

1.6.2.3 Segment Routing en redes multicapa

En esta demostración experimental, Castoldi et al demuestran las operaciones de segment routing para redes multicapa, en particular un bypass óptico dinámico basado en SR y balanceo de carga efectivo, para ello han realizado un banco de pruebas de red multicapa que muestra capacidades mejoradas para lograr utilización eficaz de los recursos garantizando al mismo tiempo operaciones de control ligeras.

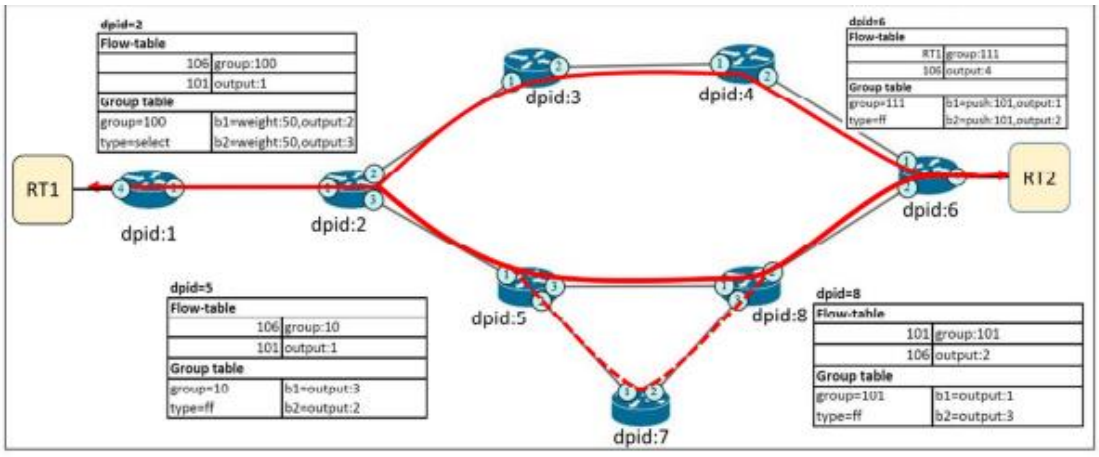


Figura 42-1: Topología de la red experimental multicapa

Fuente: Castoldi et al, 2017, p.3

En este estudio validan el potencial de la tecnología Segment Routing para simplificar y automatizar el aprovisionamiento y la recuperación de paquetes en redes multicapa. Se realizaron

dos escenarios para observar el comportamiento de la red en los cuales SR demostró ser efectivo en garantizar la utilización eficiente de los recursos con operaciones de control ligeras.

1.6.3 Integración SDN y Segment Routing

Existen múltiples formas en la que se integra una red con SDN sin embargo para la integración mediante Segment Routing existen dos formas, por medio de PCEP y utilizando el propio protocolo de enrutamiento para la comunicación.

Segment Routing como protocolo de comunicación, en principio el protocolo diseñado para comunicar un controlador con la red es OpenFlow, mediante el cual se pretende programar el estado de flujo directamente en la FIB desde el controlador, por ello SDN requiere separación entre la RIB y FIB. En lugar de un protocolo de enrutamiento que configure la RIB, con SDN y por medio de una API se puede programar la RIB desde las interfaces Northbound y por medio de un protocolo como REST.

El problema con OpenFlow es que provoca que las redes tengan estado, mientras que los routers modernos trabajan sin estado, reenvían el tráfico basado en las etiquetas entrantes o la IP de destino. Este tipo de reenvío necesita de dos búsquedas en la FIB, la primera búsqueda se realiza en la tarjeta de línea de entrada para determinar la tarjeta de línea de salida. La otra búsqueda se encarga de resolver la adyacencia en capa 2 y reescribe el encabezado. En una red con SDN y OpenFlow, la búsqueda de FIB coincide con 3-Tuple o 7-Tuple, lo que provoca una programación bidireccional de estado de flujo en la FIB a través de la red. Un nodo de red puede parecer simple como el plano de control eliminado, sin embargo, el plano de datos se vuelve exponencialmente complicado ya que necesita mantener los estados de flujo para cada aplicación en la red. Además, una SDN clásica no tiene la capacidad para realizar estos desafíos.

Cuando se despliega SDN, la filosofía de la red pasa a ser: no programar el flujo en la red, programar los paquetes. Es por ello, que se propone a Segment Routing como una alternativa genuina para OpenFlow.

Segment Routing en el plano de control utiliza las extensiones del protocolo IGP/BGP para la distribución de etiquetas salto a salto, mientras que en el plano de datos puede utilizar MPLS o IPv6, por lo que no es necesario realizar cambios en la FIB. SR con plano de datos IPv6 se llama SRv6. Una conexión a gran escala con SR puede admitir varios casos de uso de SDN que se extienden a diferentes partes de una red de proveedores de servicio. SR como tecnología de enrutamiento de origen puede coexistir con cualquier paradigma de enrutamiento de destino y cambio de etiquetas. (Abuhayat, 2020)(Paraschis, 2019, pp.68-76)

Otra forma de integrar SDN y SR es mediante el protocolo PCEP, el cual es uno de los protocolos de comunicación para redes extensas y controladores SDN. Lo primero a tener en cuenta es que se debe utilizar BGP-LS ya que permite inyectar información del IGP en BGP, y mediante este protocolo llevar la información del estado del enlace de la red al controlador SDN y después mediante PCEP calcular la ruta a través de la red. En esta arquitectura intervienen los elementos PCC el cual corresponde al cliente del cálculo de ruta que viene a ser un router dentro de la red y el PCE que es el elemento de cálculo de ruta, es decir el controlador SDN. En la figura 43-1, se puede observar la integración de SDN y el core de una red con Segment Routing mediante el protocolo de comunicación PCEP y BGP-LS, los cuales se encargan conjuntamente de informar el estado de la red, realizar el cálculo de la ruta e inyectarla en los routers correspondientes. (At, 2020, pp.1-23)(Paraschis, 2019, pp.21-31)

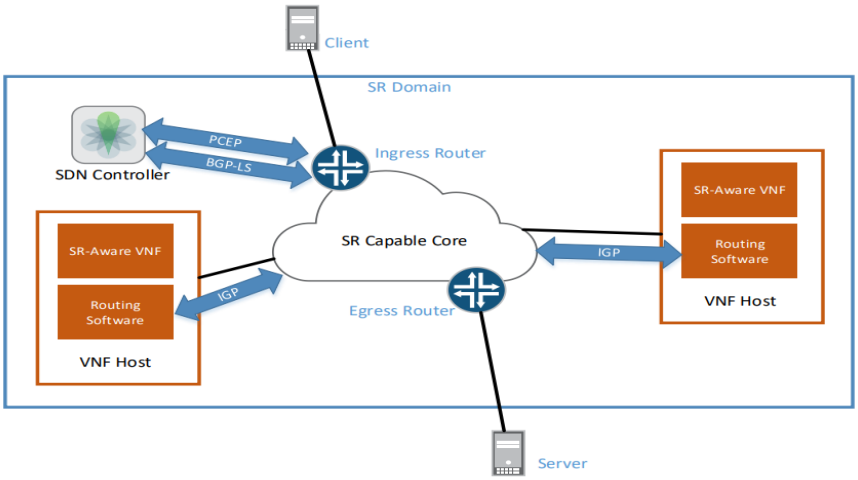


Figura 43-1: Arquitectura de red con SR y SDN mediante PCEP BGP-LS

Fuente: At, 2020, P.12

1.7 TRex

Actualmente los routers han adquirido nuevas y mejores funcionalidades y servicios, por lo que es necesario un generador de tráfico *statefull* para proporcionar escenarios de tráfico más realistas a lo que se tiene en una red real. Mediante este tipo de generadores se puede obtener métricas de desempeño precisas y descubrir cuellos de botella en escenarios de tráfico realistas.

TRex es un generador de tráfico de código libre desarrollado por cisco, cuenta con los modos stateless y statefull, el cual se puede utilizar mediante líneas de comando o mediante una interfaz gráfica sin embargo para esta versión solo se encuentra disponible el tráfico stateless mediante el cual se puede generar tráfico de capa 2 y 3. Este generador es capaz de generar y analizar tráfico

statefull de capa 4 y capa 7 del modelo OSI, de hasta 200 Gb/s, brinda estadísticas generales de la red como latencia, jitter, paquetes transmitidos y paquetes recibidos, entre otros. (Trex, 2021)

Para poner a prueba una red, se debe transmitir una cantidad de tráfico a través del dispositivo bajo prueba (DUT Device Under Test) y analizar los paquetes al final, tradicionalmente se consigue esto mediante un solo dispositivo de testeo el cuál puede ser transmisor y receptor a la vez como se ve en la figura 44-1. Este dispositivo se conoce como generador de paquetes.

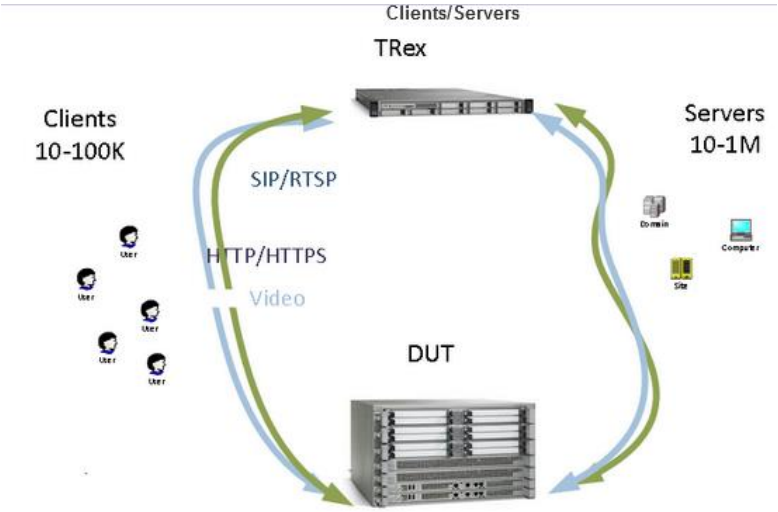


Figura 44-1: Conexión para pruebas con TRex

Fuente: TRex, 2021

1.7.1 Funcionamiento

TRex permite la creación de plantillas para múltiples tipos de tráfico y replicarlas para generar flujos de tráficos masivos por lo que permite medir la capacidad de una red. Las plantillas utilizadas son archivos pcap en formato yaml en los cuales se pueden modificar diferentes parámetros como se muestra en la figura 45-1.

```
cap_info :
   - name: cap2/udp_64B.pcap
     cps   : 1000.0
     ipg   : 10000
     rtt   : 10000
     w     : 1
     limit : 1000
```

Figura 45-1: Parámetros de tráfico para generar el tráfico

Fuente: TRex, 2021

Esta plantilla tiene un archivo cap2/udp_64B.pcap que contiene un paquete de 64 Bytes, los parámetros establecidos generan 1000 flujos es decir conexiones por segundo CPS, donde CPS=PPS, es decir esta plantilla genera 1000 paquetes por segundo. Para generar más tráfico se modifica el parámetro cps o mediante el multiplicador -m al momento de iniciar TRex. La cantidad de flujos que se genera lo determina el parámetro limit. En el ejemplo de la figura se genera 1000 paquetes en 1000 flujos, por lo que en 1000 segundos se genera 1000000 paquetes. Para saber el tamaño del tráfico generado se debe saber el número total de PPS y multiplicarlo por el tamaño de un paquete en el ejemplo el tamaño generado es 64 Mbytes.

Una vez se tiene configurado el tráfico a generar, en TRex también se debe establecer el pool de direcciones IP que corresponde a los clientes y servidores como se muestra en la figura 46-1.

```
generator :
  distribution : "seq"
  clients_start : "16.0.0.1"
  clients_end   : "16.0.0.255"
  servers_start : "48.0.0.1"
  servers_end   : "48.0.0.255"
  dual_port_mask : "1.0.0.0"
  tcp_aging      : 0
  udp_aging      : 0
```

Fuente 46-1: Parámetros de red para la generación de tráfico
Fuente: TRex, 2021

CAPÍTULO II

2. MARCO METODOLÓGICO

2.1 Diseño de la red

2.1.1 Topología de la red

La red de un proveedor de servicios tiene como objetivo ser la plataforma de transporte de información que comunique al cliente (quien solicita el acceso a un servicio) con un servidor. Esta arquitectura de red es conocida como un modelo de capas o *tiered model*, la cual permite organizar la red en tres capas; core, distribución y acceso, mediante este tipo de red se puede interconectar diferentes redes LAN/MAN de uno o varios clientes que necesiten de la infraestructura del proveedor de servicios para poder acceder hacia servidores u otras sucursales las cuales se encuentran separadas geográficamente. Los clientes deben pagar el alquiler de las líneas y servicios ofrecidos por el SP (proveedor de servicios) es por ello que la red debe cumplir con ciertos objetivos fundamentales de una red: ser escalable para tener la capacidad de crecer a medida que se agregan nuevos clientes y servicios sin afectar el funcionamiento actual que tenga la red, garantizar disponibilidad para ofrecer un rendimiento constante y confiable, la red debe ser segura y manejable para poder ser administrada por el personal de red que se encuentre disponible en ese momento, para que la red funcione de forma eficiente.

Para cumplir con dichos objetivos, se debe diseñar correctamente el corazón de la red que se encuentra en el core ya que en esta capa se encuentra presente todo el tráfico proveniente de las capas inferiores de acceso/distribución y se encarga de reenviarlo a sus destinos, es por ello que la presente tesis se ha enfocado en el diseño de esta capa para buscar un buen rendimiento de la red mediante la integración de tecnologías para reenvío de tráfico como lo es Segment Routing y manejo de la red mediante un controlador SDN.
En base al estudio de los protocolos de enlace interior, se ha seleccionado OSPF como IGP debido a su compatibilidad con la tecnología de Segment Routing aunque IS-IS también es compatible, su funcionamiento es más complejo sin embargo es muy útil cuando no se requiera que el core base el reenvío en IP, el cual no es el caso ya que para medir el comportamiento de la red, se ha capturado el tráfico IP generado por TRex en los enlaces de core.

En la figura 1-2 se muestra la topología de red en donde los routers P (Provider) son nodos de tránsito en el core, en el nodo P-2 se ha conectado al controlador SDN, los routers PE (Provider

Edge) son los encargados de comunicar la red con el exterior (clientes/servidores) y los routers CE (Customer Edge) conectan la red de clientes y la granja de servidores Web, VoIP y Streaming. La topología de red es *partial-mesh* ya que cada nodo está conectado directamente con al menos otros dos nodos de la red, creando la redundancia suficiente para ofrecer alta disponibilidad ya que el protocolo de enrutamiento tiene disponibles múltiples enlaces físicos para el reenvío de tráfico en caso de caída de alguno de ellos.

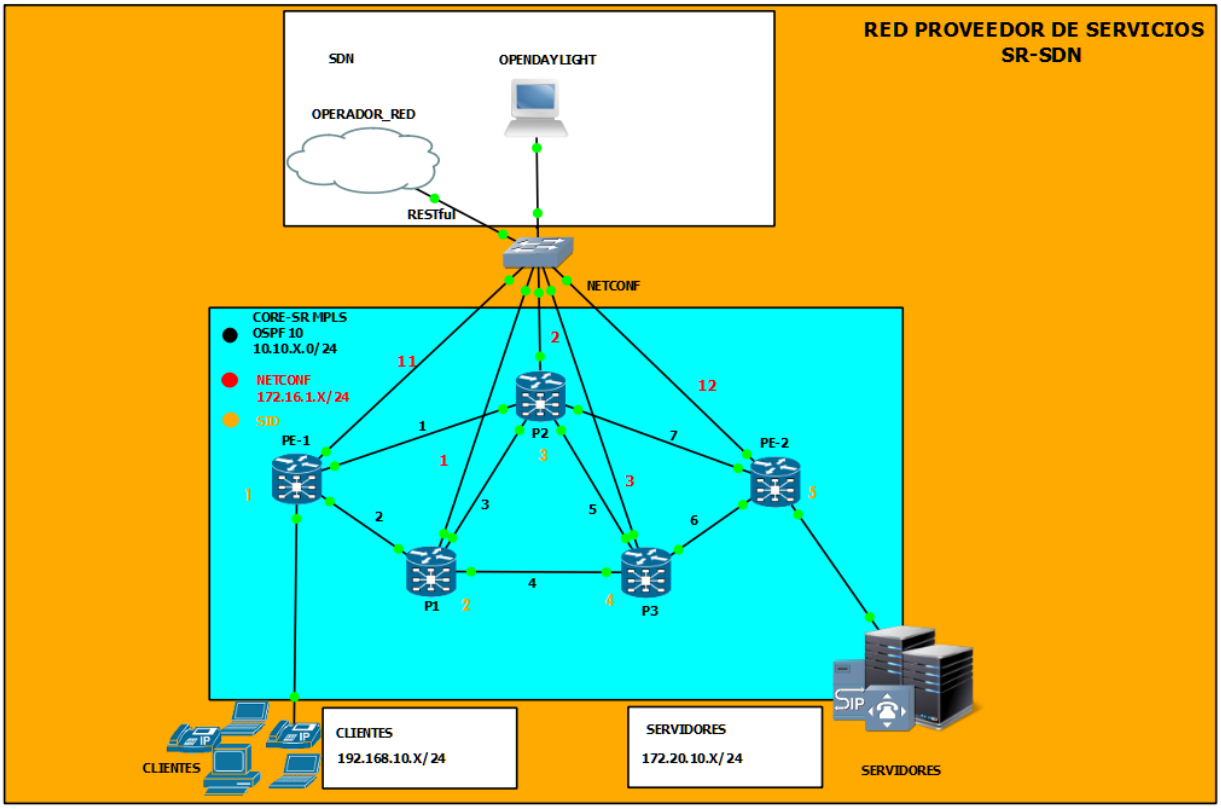


Figura 1-2: Topología de red con Segment Routing y SDN

Fuente: Loza, J. 2021

2.1.2 Equipos utilizados

En redes de proveedores de servicio a nivel mundial, la cuota de mercado pertenece un 56,4% a la empresa CISCO, un 9.5% para HUAWEI, y un 5,2% para HPE, es por ello que se ha seleccionado esta marca para el desarrollo de la presente tesis. (Fernández, 2020)

Debido a las tecnologías utilizadas y al tráfico generado, el equipo destinado para redes de proveedores de servicio por parte de CISCO son los que funcionan con el sistema operativo IOS XR, se ha utilizado la versión 6.3.1 la cual brinda soporte para OSPF, MPLS, SR y la posibilidad de integrarse con un controlador SDN, los equipos comerciales con este SO son el ASR 9K, NCS, CSR y XR 12000. En el entorno GNS3 el equipo cuenta con interfaces que tienen capacidad de 1

Gbps, cuenta con 9 puertos y un puerto especial de gestión utilizado para SDN las especificaciones completas del dispositivo se pueden ver en el ANEXO A.

2.2 Red con OSPF

2.2.1 Configuraciones

El enrutamiento por IGP OSPF es una de las tecnologías utilizadas para mejorar el tránsito de una red, debido a mecanismos integrados como ECMP para balanceo de carga, capacidades de autenticación, protección de la red en caso de que caiga un enlace poder buscar una ruta alternativa. En primer lugar, se ha realizado el direccionamiento necesario para que los routers puedan comunicarse entre sí, como se muestra en la tabla 1-2.

Tabla 1-2: Direccionamiento IP

Dispositivo	Interfaz	Dirección	Parámetros OSPF
P-1	Loopback 0	1.1.1.2/32	Instancia 10 Área 0 Router-id: 1.1.1.2
	Gi0/0/0/0	10.10.4.1/24	
	Gi0/0/0/1	10.10.3.1/24	
	Gi0/0/0/2	10.10.2.2/24	
	MgmtEth0/0/CPU0/0	172.16.1.1/24	
P-2	Loopback 0	1.1.1.3/32	Instancia 10 Área 0 Router-id: 1.1.1.3
	Gi0/0/0/0	10.10.7.1/24	
	Gi0/0/0/1	10.10.3.2/24	
	Gi0/0/0/2	10.10.5.1/24	
	Gi0/0/0/3	10.10.1.2/24	
	MgmtEth0/0/CPU0/0	172.16.1.2/24	
P-3	Loopback 0	1.1.1.4/32	Instancia 10 Área 0 Router-id: 1.1.1.4
	Gi0/0/0/0	10.10.4.2/24	
	Gi0/0/0/1	10.10.6.1/24	
	Gi0/0/0/2	10.10.5.2/24	
	MgmtEth0/0/CPU0/0	172.16.1.3/24	
PE-1	Loopback 0	1.1.1.1/32	Instancia 10 Área 0 Router-id: 1.1.1.1
	Gi0/0/0/0	10.10.1.1/24	
	Gi0/0/0/1	10.10.2.1/24	
	Gi0/0/0/2	10.10.20.1/24	
	MgmtEth0/0/CPU0/0	172.16.1.11/24	

PE-2	Loopback 0	1.1.1.5/32	Instancia 10 Área 0 Router-id: 1.1.1.5
	Gi0/0/0/0	10.10.6.2/24	
	Gi0/0/0/1	10.10.7.2/24	
	Gi0/0/0/2	10.10.40.1/24	
	MgmtEth0/0/CPU0/0	172.16.1.12/24	
Opendaylight	Eth0	172.16.1.10/24	
POSTMAN	VMNET 11	172.16.1.1/24	
TRex	Eth0	192.168.10.1/24	
	Eth1	172.20.10.1/24	

Realizado por: Loza, J. 2021

Una vez se ha estructurado la red y establecido el direccionamiento, se han configurado los distintos equipos para establecer enrutamiento, debido a la topología de la red, se ha modificado el costo de los enlaces entre PE-1 y P-2 y entre P-2 y PE-2 con un valor de 10000 como se muestra en la figura 2-2 se tienen dos rutas para llegar con el mismo costo, para mayor detalle de la configuración se puede observar el ANEXO B. Esto es para que desde PE-1 hacia PE-2 que son los puntos finales de la red y donde se conectan los clientes y servidores, exista dos rutas para el reenvío de tráfico, realizando balanceo de carga para optimizar la red.

```
RP/0/0/CPU0:P-2#sh run
Wed Jul 21 17:37:38.604 UTC
Building configuration...
!! IOS XR Configuration 6.1.3
!! Last configuration change at Wed Jul 21 17:30:43 2021 by cisco
!
hostname P-2
interface Loopback0
 ipv4 address 1.1.1.3 255.255.255.255
!
interface MgmtEth0/0/CPU0/0
 ipv4 address 172.16.1.1 255.255.255.0
!
interface GigabitEthernet0/0/0/0
 ipv4 address 10.10.7.1 255.255.255.0
!
interface GigabitEthernet0/0/0/1
 ipv4 address 10.10.3.2 255.255.255.0
!
interface GigabitEthernet0/0/0/2
 ipv4 address 10.10.5.1 255.255.255.0
!
interface GigabitEthernet0/0/0/3
 ipv4 address 10.10.1.2 255.255.255.0
!
router ospf 10
 router-id 1.1.1.3
 auto-cost reference-bandwidth 10000
 area 0
  interface Loopback0
  !
  interface GigabitEthernet0/0/0/0
   cost 20
   network point-to-point
  !
  interface GigabitEthernet0/0/0/1
   network point-to-point
  !
  interface GigabitEthernet0/0/0/2
   network point-to-point
  !
  interface GigabitEthernet0/0/0/3
   cost 20
   network point-to-point
```

Figura 2-2: Configuración de P-2

Fuente: Loza, J. 2021

2.2.2 Tablas de rutas y funcionamiento

La configuración es la misma en los distintos routers a excepción del costo, y para comprobar que todos han establecido adyacencias se verifica la tabla de rutas en uno de los equipos, en la figura 3-2 se observa que el router PE-1 tiene todas las direcciones para llegar a los destinos y en la figura 4-2 se observa la base de datos OSPF en donde contiene los ID de cada router OSPF en la instancia 10 dentro del área 0, de igual forma se muestra los vecinos del router PE-1, con lo cual se puede definir que el enrutamiento es correcto.

```
RP/0/0/CPU0:PE-1#sh route
Wed Jul 21 17:41:30.542 UTC

Codes: C - connected, S - static, R - RIP, B - BGP, (>) - Diversion path
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - ISIS, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, su - IS-IS summary null, * - candidate default
       U - per-user static route, o - ODR, L - local, G  - DAGR, l - LISP
       A - access/subscriber, a - Application route
       M - mobile route, r - RPL, (!) - FRR Backup path

Gateway of last resort is not set

L    1.1.1.1/32 is directly connected, 00:11:43, Loopback0
O    1.1.1.2/32 [110/11] via 10.10.2.2, 00:11:08, GigabitEthernet0/0/0/1
O    1.1.1.3/32 [110/11] via 10.10.1.2, 00:10:48, GigabitEthernet0/0/0/0
O    1.1.1.4/32 [110/21] via 10.10.1.2, 00:10:20, GigabitEthernet0/0/0/0
                [110/21] via 10.10.2.2, 00:10:20, GigabitEthernet0/0/0/1
O    1.1.1.5/32 [110/31] via 10.10.1.2, 00:10:08, GigabitEthernet0/0/0/0
                [110/31] via 10.10.2.2, 00:10:08, GigabitEthernet0/0/0/1
C    10.10.1.0/24 is directly connected, 00:11:43, GigabitEthernet0/0/0/0
L    10.10.1.1/32 is directly connected, 00:11:43, GigabitEthernet0/0/0/0
C    10.10.2.0/24 is directly connected, 00:11:43, GigabitEthernet0/0/0/1
L    10.10.2.1/32 is directly connected, 00:11:43, GigabitEthernet0/0/0/1
O    10.10.3.0/24 [110/20] via 10.10.1.2, 00:10:46, GigabitEthernet0/0/0/0
                  [110/20] via 10.10.2.2, 00:10:46, GigabitEthernet0/0/0/1
O    10.10.4.0/24 [110/20] via 10.10.2.2, 00:11:08, GigabitEthernet0/0/0/1
O    10.10.5.0/24 [110/20] via 10.10.1.2, 00:10:46, GigabitEthernet0/0/0/0
O    10.10.6.0/24 [110/30] via 10.10.1.2, 00:10:20, GigabitEthernet0/0/0/0
                  [110/30] via 10.10.2.2, 00:10:20, GigabitEthernet0/0/0/1
O    10.10.7.0/24 [110/30] via 10.10.1.2, 00:10:48, GigabitEthernet0/0/0/0
C    10.10.20.0/24 is directly connected, 00:11:43, GigabitEthernet0/0/0/2
L    10.10.20.1/32 is directly connected, 00:11:43, GigabitEthernet0/0/0/2
O    10.10.40.0/24 [110/40] via 10.10.1.2, 00:10:08, GigabitEthernet0/0/0/0
                   [110/40] via 10.10.2.2, 00:10:08, GigabitEthernet0/0/0/1
O    172.20.10.0/24 [110/1040] via 10.10.1.2, 00:01:24, GigabitEthernet0/0/0/0
                    [110/1040] via 10.10.2.2, 00:01:24, GigabitEthernet0/0/0/1
O    192.168.10.0/24 [110/1010] via 10.10.20.2, 00:01:30, GigabitEthernet0/0/0/2
```

Figura 3-2: Tabla de rutas OSPF en PE-1

Fuente: Loza, J. 2021

```
RP/0/0/CPU0:PE-1#sh ospf database
Wed Jul 21 17:49:30.329 UTC

            OSPF Router with ID (1.1.1.1) (Process ID 10)

                Router Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum Link count
1.1.1.1         1.1.1.1         570         0x80000005 0x00c727 6
1.1.1.2         1.1.1.2         1105        0x80000005 0x00994f 7
1.1.1.3         1.1.1.3         1090        0x80000006 0x000348 9
1.1.1.4         1.1.1.4         1091        0x80000005 0x007360 7
1.1.1.5         1.1.1.5         572         0x80000005 0x009a08 6
11.11.11.11     11.11.11.11     537         0x80000007 0x00b98d 2
22.22.22.22     22.22.22.22     540         0x80000007 0x00135c 2

                Net Link States (Area 0)

Link ID         ADV Router      Age         Seq#       Checksum
10.10.20.1      1.1.1.1         570         0x80000001 0x00e9f2
10.10.40.1      1.1.1.5         572         0x80000001 0x00454f
RP/0/0/CPU0:PE-1#
RP/0/0/CPU0:PE-1#sh ospf neighbor
Wed Jul 21 17:49:34.469 UTC

* Indicates MADJ interface
# Indicates Neighbor awaiting BFD session up

Neighbors for OSPF 10

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.1.1.3         1     FULL/  -        00:00:33    10.10.1.2       GigabitEthernet0/0/0/0
    Neighbor is up for 00:18:52
1.1.1.2         1     FULL/  -        00:00:35    10.10.2.2       GigabitEthernet0/0/0/1
    Neighbor is up for 00:19:12
11.11.11.11     1     FULL/BDR        00:00:39    10.10.20.2      GigabitEthernet0/0/0/2
    Neighbor is up for 00:09:34

Total neighbor count: 3
```

Figura 4-2: Tabla de vecinos y base de datos OSPF en PE-1

Fuente: Loza, J. 2021

2.3 Red con Segment Routing

Una vez se realizaron las pruebas con OSPF se implementó la tecnología de SR sobre el plano de control OSPF y con el plano de datos MPLS en los routers de core.

2.3.1 Configuración de SR

Para SR es necesario definir los índices de cada router y las interfaces, como se muestra en la tabla 2-2, de esta forma SR construye el Prefix-SID a partir del Global Block que por defecto está comprendido por las etiquetas 16000-23999.

Tabla 2-2: Segment Routing

Router	Interfaz	Index	Prefix-SID
PE-1	Loopback 0	1	16001
PE-2	Loopback 0	5	16005
P-1	Loopback 0	2	16002
P-2	Loopback 0	3	16003
P-3	Loopback 0	4	16004

Realizado por: Loza,2021

La configuración de SR se muestra en la figura 5-2, dentro de la instancia de OSPF se habilita SR sobre el plano de datos MPLS y en el área se indica la interfaz y el índice correspondiente para cada router. La configuración es la misma para todos los routers, la configuración completa de los dispositivos de red se encuentra en el ANEXO B.

```
router ospf 10
 router-id 1.1.1.1
 segment-routing mpls
 segment-routing forwarding mpls
 auto-cost reference-bandwidth 10000
 area 0
  interface Loopback0
   prefix-sid index 1
  !
```

Figura 5-2: Configuración de Segment Routing en PE-1

Fuente: Loza, J. 2021

2.3.2 Tabla de forwarding y funcionamiento

Una vez configurada la red, se verifican las interfaces de reenvío como se muestra en la figura 6-2 las interfaces están habilitadas para el reenvío de etiquetas sin embargo el protocolo LDP no está habilitado, esta es una de las ventajas que se observan al implementar Segment Routing ya que no hace uso de un protocolo de distribución de etiquetas por lo que es más ligero que MPLS. En la misma figura también se puede observar la tabla de forwarding de PE-1, en donde se comprueba que PE-1 conoce los Prefix-SID de todos los routers de la red y como llegar hacia ellos. Como SR está configurado sobre OSPF, para llegar desde PE-1 a PE-2 también tiene dos rutas para el reenvío de tráfico, como se había configurado en OSPF con la modificación del costo.

```
RP/0/0/CPU0:PE-1#sh mpls interfaces
Wed Jul 21 18:55:11.629 UTC
Interface                  LDP      Tunnel   Static   Enabled
-------------------------- -------- -------- -------- --------
GigabitEthernet0/0/0/0     No       No       No       Yes
GigabitEthernet0/0/0/1     No       No       No       Yes
GigabitEthernet0/0/0/2     No       No       No       Yes
RP/0/0/CPU0:PE-1#sh mpls forwarding
Wed Jul 21 18:55:13.529 UTC
Local  Outgoing    Prefix             Outgoing     Next Hop        Bytes
Label  Label       or ID              Interface                    Switched
------ ----------- ------------------ ------------ --------------- ------------
16002  Pop         SR Pfx (idx 2)     Gi0/0/0/1    10.10.2.2       0
16003  Pop         SR Pfx (idx 3)     Gi0/0/0/0    10.10.1.2       0
16004  16004       SR Pfx (idx 4)     Gi0/0/0/0    10.10.1.2       0
       16004       SR Pfx (idx 4)     Gi0/0/0/1    10.10.2.2       0
16005  16005       SR Pfx (idx 5)     Gi0/0/0/0    10.10.1.2       0
       16005       SR Pfx (idx 5)     Gi0/0/0/1    10.10.2.2       0
24000  Pop         SR Adj (idx 0)     Gi0/0/0/0    10.10.1.2       0
24001  Pop         SR Adj (idx 0)     Gi0/0/0/0    10.10.1.2       0
24002  Pop         SR Adj (idx 0)     Gi0/0/0/2    10.10.20.2      0
24003  Pop         SR Adj (idx 0)     Gi0/0/0/2    10.10.20.2      0
24004  Pop         SR Adj (idx 0)     Gi0/0/0/1    10.10.2.2       0
24005  Pop         SR Adj (idx 0)     Gi0/0/0/1    10.10.2.2       0
```

Figura 6-2: Tabla de forwarding SR en PE-1

Fuente: Loza, J. 2021

En la figura 7-2 se muestra que efectivamente SR está corriendo sobre la red, el paquete en específico tiene un destino a la etiqueta 16002 correspondiente a P1.

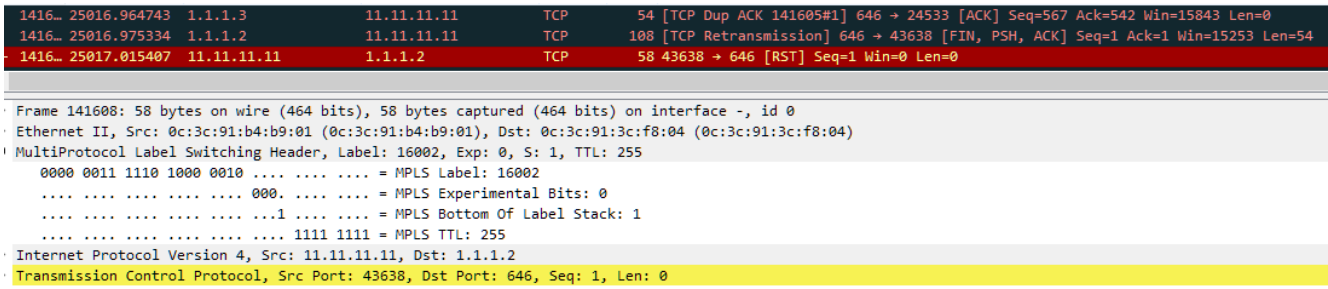


Figura 7-2: Paquete enrutado mediante SR hacia el destino 16002

Fuente: Loza, J. 2021

Después de haber verificado el correcto funcionamiento de SR sobre MPLS se debe comprobar el plano de control de OSPF para SR. Para ello en PE-1 se ejecuta el comando *sh ospf database opaque-area adv-router* que muestran los mensajes Opaque LSA que se está enviando con P-1, en la figura 8-2 se muestra el primer mensaje el cual es Opaque LSA de tipo 4 el cual contiene la información del router al que está anunciando, en este caso a la 1.1.1.2 que pertenece a P-1, también tiene información del algoritmo SR y el rango de etiquetas.

```
            OSPF Router with ID (1.1.1.1) (Process ID 10)

                Type-10 Opaque Link Area Link States (Area 0)

LS age: 599
Options: (No TOS-capability, DC)
LS Type: Opaque Area Link
Link State ID: 4.0.0.0
Opaque Type: 4
Opaque ID: 0
Advertising Router: 1.1.1.2
LS Seq Number: 80000001
Checksum: 0x2649
Length: 52

  Router Information TLV: Length: 4
  Capabilities:
    Graceful Restart Helper Capable
    Stub Router Capable
    All capability bits: 0x60000000

  Segment Routing Algorithm TLV: Length: 2
    Algorithm: 0
    Algorithm: 1

  Segment Routing Range TLV: Length: 12
    Range Size: 8000

      SID sub-TLV: Length 3
       Label: 16000
```

Figura 8-2: Mensaje type 4 de OSPF-SR

Fuente: Loza, J. 2021

En la figura 9-2 se muestra el mensaje Opaque LSA de tipo 7, utilizado para anunciar los Segment IDs, en este caso se anuncia el prefijo 1.1.1.2/32 y el índice 2, información correspondiente a P-1. Mientras que en la figura 10-2 se muestra el paquete con el Opaque LSA type 7 capturado con la información correspondiente a PE-2.

```
LS age: 599
Options: (No TOS-capability, DC)
LS Type: Opaque Area Link
Link State ID: 7.0.0.1
Opaque Type: 7
Opaque ID: 1
Advertising Router: 1.1.1.2
LS Seq Number: 80000001
Checksum: 0x88b
Length: 44

  Extended Prefix TLV: Length: 20
    Route-type: 1
    AF        : 0
    Flags     : 0x40
    Prefix    : 1.1.1.2/32

    SID sub-TLV: Length: 8
      Flags     : 0x0
      MTID      : 0
      Algo      : 0
      SID Index : 2
```

Figura 9-2: Mensaje type 7 de OSPF-SR

Fuente: Loza, J. 2021

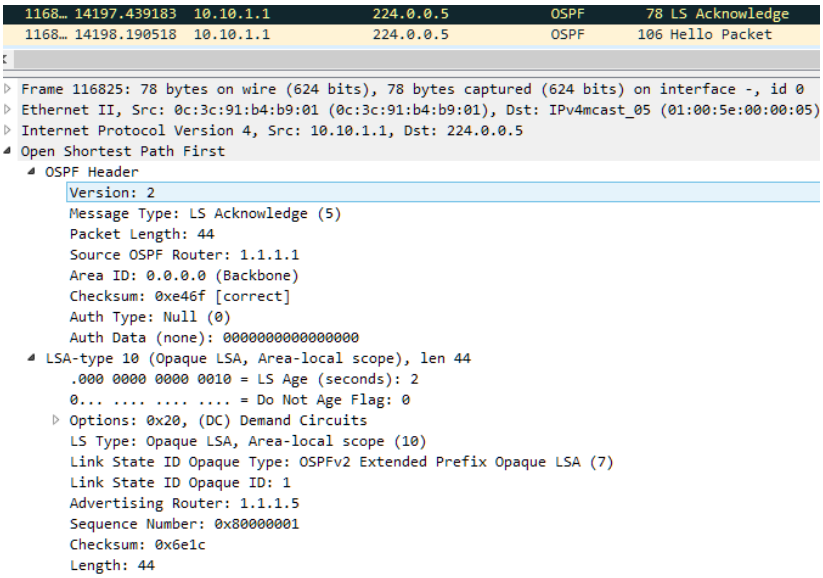


Figura 10-2: Paquete con el mensaje type 7 de OSPF-SR

Fuente: Loza, J. 2021

El mensaje Opaque LSA tipo 8 se manda por cada adyacencia que PE-1 anuncia a P-1, en la figura 11-2 se muestra uno de los mensajes, exactamente el anuncio para la adyacencia con 1.1.1.4 que pertenece a P-3.

```
LS age: 599
Options: (No TOS-capability, DC)
LS Type: Opaque Area Link
Link State ID: 8.0.0.3
Opaque Type: 8
Opaque ID: 3
Advertising Router: 1.1.1.2
LS Seq Number: 80000002
Checksum: 0x166b
Length: 68

  Extended Link TLV: Length: 44
    Link-type : 1
    Link ID   : 1.1.1.4
    Link Data : 10.10.4.1

   Adj sub-TLV: Length: 7
      Flags     : 0xe0
      MTID      : 0
      Weight    : 0
      Label     : 24000

   Adj sub-TLV: Length: 7
      Flags     : 0x60
      MTID      : 0
      Weight    : 0
      Label     : 24001

   Remote If Address sub-TLV: Length: 4
      Neighbor Address: 10.10.4.2
```

Figura 11-2: Mensaje type 8 de OSPF-SR

Fuente: Loza, J. 2021

Una vez se ha verificado que OSPF está enviando los mensajes utilizados por SR para anunciar entre los routers del core toda la información para que funcione el protocolo de enrutamiento, se puede inyectar el tráfico para observar el rendimiento de la red. En la figura 12-2 se muestra la captura del paquete con la extensión Opaque LSA Type 10.

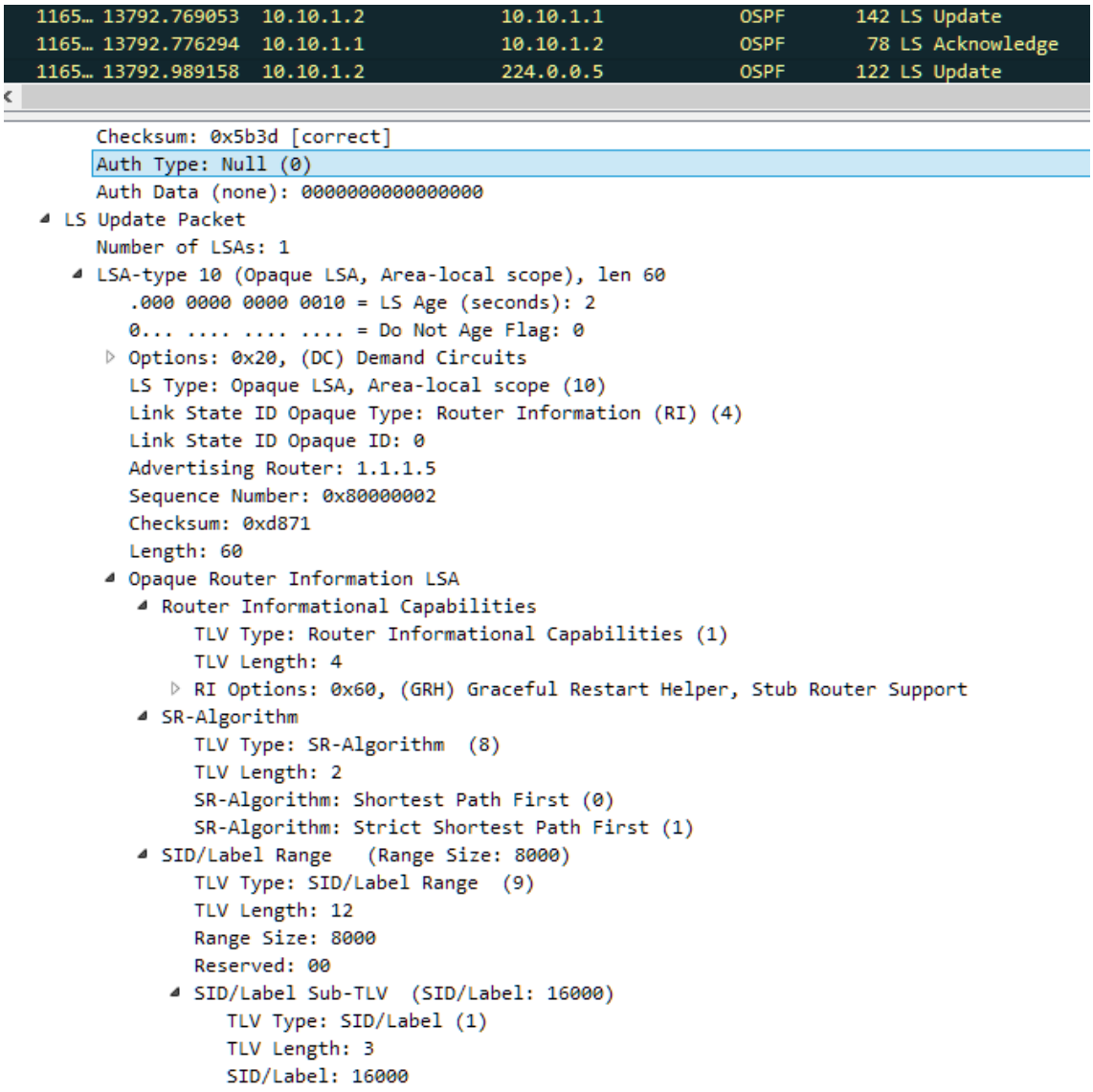


Figura 12-2: Captura del mensaje Opaque LSA Type 10

Fuente: Loza, J. 2021

2.3.3 Configuración de TI-LFA

Uno de los objetivos en el diseño de la red es que debe tener alta disponibilidad, para ello se adoptó una topología *partial-mesh* pero no es suficiente ya que OSPF debe calcular una nueva ruta cuando se falle algún enlace, como se muestra en la figura 13-2 el tiempo que requiere es muy alto y por ende se pierda hasta un 64% de los paquetes ICMP generados. Para simular la caída del enlace primero se observa la ruta principal desde CE-1 a CE-2 y mediante la herramienta *suspend* que ofrece GNS3 se ha suspendido el enlace entre PE-1 y P-1.

```
vyatta@vyatta:~$ traceroute 172.20.10.1
traceroute to 172.20.10.1 (172.20.10.1), 30 hops max, 60 byte packets
 1  10.10.20.1 (10.10.20.1)  27.261 ms  24.563 ms  24.498 ms
 2  10.10.2.2 (10.10.2.2)  76.806 ms  76.678 ms  76.542 ms
 3  10.10.4.2 (10.10.4.2)  76.464 ms  76.418 ms  76.323 ms
 4  10.10.6.2 (10.10.6.2)  227.577 ms  235.000 ms  234.996 ms
 5  172.20.10.1 (172.20.10.1)  234.972 ms  234.985 ms  234.979 ms
vyatta@vyatta:~$ ping 172.20.10.1 c 50
PING 172.20.10.1 (172.20.10.1) 56(84) bytes of data.
64 bytes from 172.20.10.1: icmp_req=1 ttl=60 time=41.7 ms
64 bytes from 172.20.10.1: icmp_req=2 ttl=60 time=45.8 ms
64 bytes from 172.20.10.1: icmp_req=3 ttl=60 time=101 ms
64 bytes from 172.20.10.1: icmp_req=4 ttl=60 time=38.6 ms
64 bytes from 172.20.10.1: icmp_req=37 ttl=61 time=38.0 ms
64 bytes from 172.20.10.1: icmp_req=38 ttl=61 time=43.4 ms
64 bytes from 172.20.10.1: icmp_req=39 ttl=61 time=31.0 ms
64 bytes from 172.20.10.1: icmp_req=40 ttl=61 time=42.8 ms
64 bytes from 172.20.10.1: icmp_req=41 ttl=61 time=43.0 ms
64 bytes from 172.20.10.1: icmp_req=42 ttl=61 time=39.4 ms
64 bytes from 172.20.10.1: icmp_req=43 ttl=61 time=96.6 ms
64 bytes from 172.20.10.1: icmp_req=44 ttl=61 time=41.8 ms
64 bytes from 172.20.10.1: icmp_req=45 ttl=61 time=67.6 ms
64 bytes from 172.20.10.1: icmp_req=46 ttl=61 time=34.9 ms
64 bytes from 172.20.10.1: icmp_req=47 ttl=61 time=51.4 ms
64 bytes from 172.20.10.1: icmp_req=48 ttl=61 time=103 ms
64 bytes from 172.20.10.1: icmp_req=49 ttl=61 time=33.1 ms
64 bytes from 172.20.10.1: icmp_req=50 ttl=61 time=32.2 ms

--- 172.20.10.1 ping statistics ---
50 packets transmitted, 18 received, 64% packet loss, time 49114ms
rtt min/avg/max/mdev = 31.058/51.525/103.804/23.371 ms
```

Figura 13-2: Protección de enlaces con OSPF

Fuente: Loza, J. 2021

Es por ello que se ha configurado una de las herramientas de Segment Routing para la protección de enlaces en la red, TI-LFA. En la figura 14-2 se muestra la configuración en PE-1, la cual es la misma para los demás routers.

```
RP/0/0/CPU0:PE-1#sh running-config router ospf
Fri Jul 23 17:21:44.059 UTC
router ospf 10
 router-id 1.1.1.1
 segment-routing mpls
 segment-routing forwarding mpls
 auto-cost reference-bandwidth 10000
 area 0
  interface Loopback0
   prefix-sid index 1
  !
  interface GigabitEthernet0/0/0/0
   network point-to-point
   fast-reroute per-prefix
   fast-reroute per-prefix ti-lfa enable
  !
  interface GigabitEthernet0/0/0/1
   network point-to-point
   fast-reroute per-prefix
   fast-reroute per-prefix ti-lfa enable
  !
  interface GigabitEthernet0/0/0/2
  !
 !
!
```

Figura 14-2: Configuración TI-LFA en PE-1

Fuente: Loza, J. 2021

2.3.3.1 Funcionamiento de TI-LFA

Para comprobar que las configuraciones están correctamente realizadas, se verifica la tabla de forwarding y como se puede ver en la figura 15-2 se han creado las rutas de backup las cuales se diferencian con el símbolo (!).

```
RP/0/0/CPU0:PE-1#sh mpls forwarding
Fri Jul 23 17:24:27.018 UTC
Local  Outgoing    Prefix            Outgoing     Next Hop        Bytes
Label  Label       or ID             Interface                    Switched
------ ----------- ----------------- ------------ --------------- ------------
16002  Pop         SR Pfx (idx 2)    Gi0/0/0/1    10.10.2.2       0
       16002       SR Pfx (idx 2)    Gi0/0/0/0    10.10.1.2       0           (!)
16003  Pop         SR Pfx (idx 3)    Gi0/0/0/0    10.10.1.2       0
       16003       SR Pfx (idx 3)    Gi0/0/0/1    10.10.2.2       0           (!)
16004  16004       SR Pfx (idx 4)    Gi0/0/0/0    10.10.1.2       0
       16004       SR Pfx (idx 4)    Gi0/0/0/1    10.10.2.2       0
16005  16005       SR Pfx (idx 5)    Gi0/0/0/0    10.10.1.2       0
       16005       SR Pfx (idx 5)    Gi0/0/0/1    10.10.2.2       0
24000  Pop         SR Adj (idx 0)    Gi0/0/0/1    10.10.2.2       0
       16002       SR Adj (idx 0)    Gi0/0/0/0    10.10.1.2       0           (!)
24001  Pop         SR Adj (idx 0)    Gi0/0/0/1    10.10.2.2       0
24002  Pop         SR Adj (idx 0)    Gi0/0/0/0    10.10.1.2       0
       16003       SR Adj (idx 0)    Gi0/0/0/1    10.10.2.2       0           (!)
24003  Pop         SR Adj (idx 0)    Gi0/0/0/0    10.10.1.2       0
24004  Pop         SR Adj (idx 0)    Gi0/0/0/2    10.10.20.2      0
24005  Pop         SR Adj (idx 0)    Gi0/0/0/2    10.10.20.2      0
```

Figura 15-2: Tabla de forwarding SR con TI-LFA en PE-1

Fuente: Loza, J. 2021

En la figura 16-2 se puede verificar que desde PE-1 a PE-2 existe protección para el enrutamiento por segmentos mediante TI-LFA – Zero-Segment, ya que como se observa en la descripción de la ruta de backup, no existe ningún segmento P, Q adicional por el cual deba reenviarse el paquete.

```
RP/0/0/CPU0:PE-1#sh ospf 10 routes 1.1.1.5/32 backup-path
Fri Jul 23 17:25:07.565 UTC

Topology Table for ospf 10 with ID 1.1.1.1

Codes: O - Intra area, O IA - Inter area
       O E1 - External type 1, O E2 - External type 2
       O N1 - NSSA external type 1, O N2 - NSSA external type 2

O    1.1.1.5/32, metric 31
       10.10.2.2, from 1.1.1.5, via GigabitEthernet0/0/0/1, path-id 1
           Backup path:
              10.10.1.2, from 1.1.1.5, via GigabitEthernet0/0/0/0, protected bitmap 0000000000000001
              Attributes: Metric: 31, Primary, Downstream, Node Protect, SRLG Disjoint
       10.10.1.2, from 1.1.1.5, via GigabitEthernet0/0/0/0, path-id 2
           Backup path:
              10.10.2.2, from 1.1.1.5, via GigabitEthernet0/0/0/1, protected bitmap 0000000000000002
              Attributes: Metric: 31, Primary, Downstream, Node Protect, SRLG Disjoint
```

Figura 16-2: Protección mediante TI-LFA Zero-Segment

Fuente: Loza, J. 2021

En la red, existen nodos que tienen protección con double-segment, se observa en la figura 17-2 como en P-1 existe este tipo de TI-LFA para la protección del reenvío de tráfico hacia PE-1, el camino en este caso está conformado por el nodo P que tiene destino hacia P-2 y el nodo Q en el que está el destino final PE-1.

```
RP/0/0/CPU0:P-1#sh ospf routes backup-path
Fri Jul 23 18:03:56.195 UTC

Topology Table for ospf 10 with ID 1.1.1.2

Codes: O - Intra area, O IA - Inter area
       O E1 - External type 1, O E2 - External type 2
       O N1 - NSSA external type 1, O N2 - NSSA external type 2

O    1.1.1.1/32, metric 11
       10.10.2.1, from 1.1.1.1, via GigabitEthernet0/0/0/2, path-id 1
           Backup path: TI-LFA, Repair-List: P node: 1.1.1.3         Label: 3
                                             Q node: 1.1.1.1         Label: 24005
              10.10.3.2, from 1.1.1.1, via GigabitEthernet0/0/0/1, protected bitmap 0000000000000001
              Attributes: Metric: 31, SRLG Disjoint
```

Figura 17-2: Protección mediante TI-LFA Double-Segment

Fuente: Loza, J. 2021

2.4 Rendimiento de la red

Para medir el rendimiento de la red tanto con OSPF como de Segment Routing, se inyecta tráfico Web, VoIP y Streaming en distintas pruebas de estrés como se muestra en la tabla 3-2, aumentando la cantidad de tráfico que ingresa para observar en qué punto el rendimiento entre las dos tecnologías empieza a marcar una tendencia favorable hacia SR. Esta tendencia se mide en base a latencia, pérdida de paquetes, jitter y calidad de las conexiones por streaming.

Tabla 3-2: Tráfico para pruebas de estrés

Prueba	Usuarios	Tráfico (Mbyes)
1	50	106,40
2	150	319,2
3	250	638,4
4	350	744,8

Realizado por: Loza, J. 2021

Cabe recalcar que los usuarios son una variable significativa ya que las redes no se despliegan en función de un número de usuarios, si no del tráfico que estos generan y las capacidades que deben incluir para poder transportar una determinada cantidad de tráfico sin que el usuario final note problemas al momento de trabajar con los mismos.

Para realizar un análisis estadístico confiable es recomendable utilizar pruebas paramétricas las cuales tienen menor probabilidad de error y son mucho más confiables, para realizar pruebas paramétricas es necesario tener mínimo 30 muestras por cada prueba. Debido a que se tienen 7 enlaces y en cada de uno de ellos se genera una muestra, se deben realizar 5 repeticiones de la misma prueba para obtener 35 muestras en cada prueba. Estas muestras se promedian para obtener un comportamiento fiable de la red por cada prueba y se comparan estadísticamente entre los dos grupos (OSPF y SR) para definir el rendimiento de la red.

2.4.1.1 Configuración del generador TRex

Para integrar TRex se aprovecha la característica de gns3 para alojar dockers, se puede descargar la imagen desde el repositorio web de docker *trexcisco/trex* e instalarlo en una instancia de gsn3 como se muestra en el ANEXO D. Una vez iniciado se accede a una interfaz de línea de comandos mediante la cual se debe configurar los parámetros para generar el tráfico.

Desde el terminal y accediendo en el directorio *trex-core/scripts*, mediante el comando *sudo ./dpdk_setup_ports.py -i* se realizó la configuración de los puertos que actúan como cliente y

servidor, tendiendo en cuenta que el mismo equipo virtual actúa como cliente y servidor, utilizando dos interfaces diferentes para ello. La configuración de los puertos se muestra en la figura 18-2.

```
[root@c-s etc]# cat tesis.yaml
### Config file generated by dpdk_setup_ports.py ###

- port_limit: 2
  version: 2
  interfaces: ['eth0', 'eth1']
  low_end: true
  port_info:
      - ip: 192.168.10.1
        default_gw: 192.168.10.2
      - ip: 172.20.10.1
        default_gw: 172.20.10.2

  platform:
      master_thread_id: 0
      latency_thread_id: 1
      dual_if:
        - socket: 0
          threads: [2,3]
```

Figura 18-2: Configuración de los puertos cliente/servidor en TRex

Fuente: Loza, J. 2021

El siguiente paso fue configurar el archivo para la generación de tráfico, en el cual se especifica el rango de direcciones IPv4 de clientes y servidores. Para especificar el tráfico, TRex cuenta con archivos pcap en donde se incluye el archivo cap del tráfico deseado. Cada servicio simulado tiene un número de conexiones por minuto (cps), un espacio de tiempo entre paquetes (igp) el cual es el mismo que el rtt. W es un sintonizador de los clientes/servidores IP generados, un valor de 1 indica un servidor un cliente, mientras que un valor de 4 configura una ráfaga de dos asignaciones de un mismo cliente a un servidor. La figura 19-2 muestra la configuración del archivo de generación de tráfico en la cual se consideran los pcap necesarios para simular los servicios Web, VoIP y Streaming, debido a que en una red que ofrece el transporte de estos servicios, el Web es el más generado, se han configurado una cantidad mayor de conexiones por segundo. Esta configuración cuenta con 255 clientes y 255 servidores, a los cuales TRex les asigna una dirección en base al pool configurado.

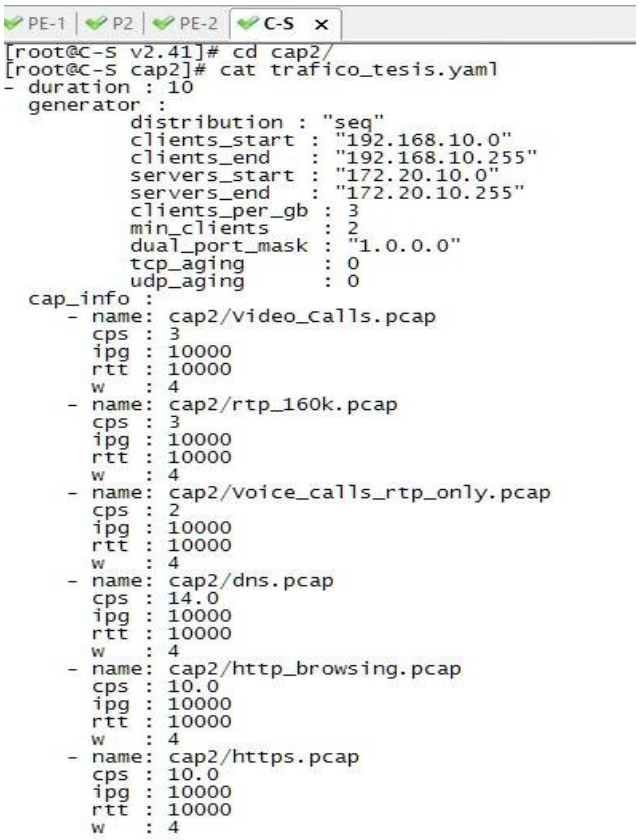


Figura 19-2: Archivo de generación de tráfico Web, VoIP y Streaming

Fuente: Loza, J. 2021

2.4.1.2 Generación de tráfico

La ejecución del programa se realizó mediante el comando *sudo ./t-rex-64 -f cap2/trafico_tesis.yaml -c 4 -m 1 -d 10 --nc*.

La opción -c indica el número de subprocesos de hardware que se utilizaron por cada par de interfaces, la opción -m es un multiplicador de la tasa de transferencia, es decir al archivo original. Esta opción es la que se ha variado para generar las distintas tasas de transferencia para observar el comportamiento de la red. La opción -d indica el tiempo en el que se está enviando el archivo de generación de tráfico, por lo tanto, se mantuvo el mismo tiempo de 60 segundos para que no se genere más tráfico del considerado con el multiplicador. En la figura 20-2 se observa el tráfico resultante de la plantilla a utilizar, en este caso el tráfico inicial es de 126,68 Mbytes.

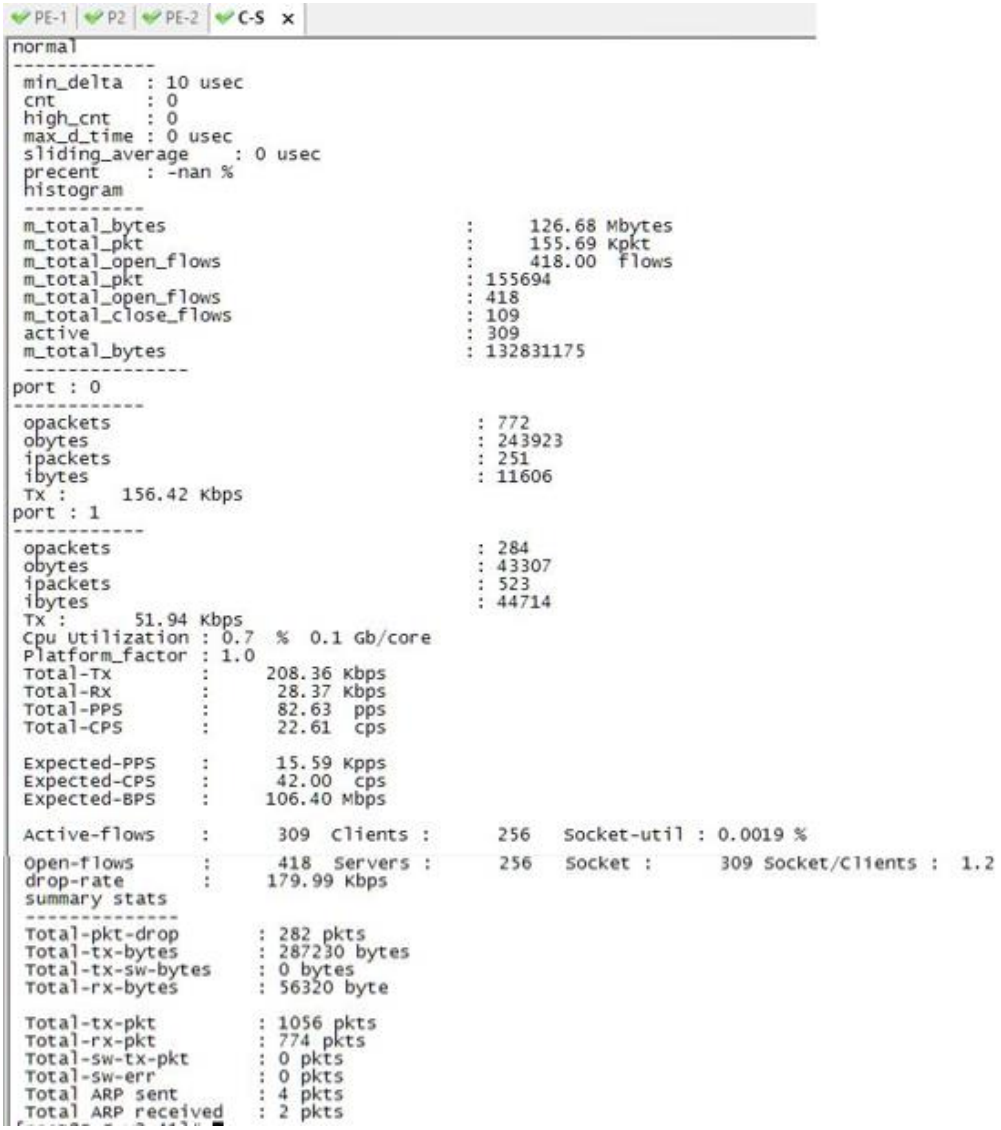
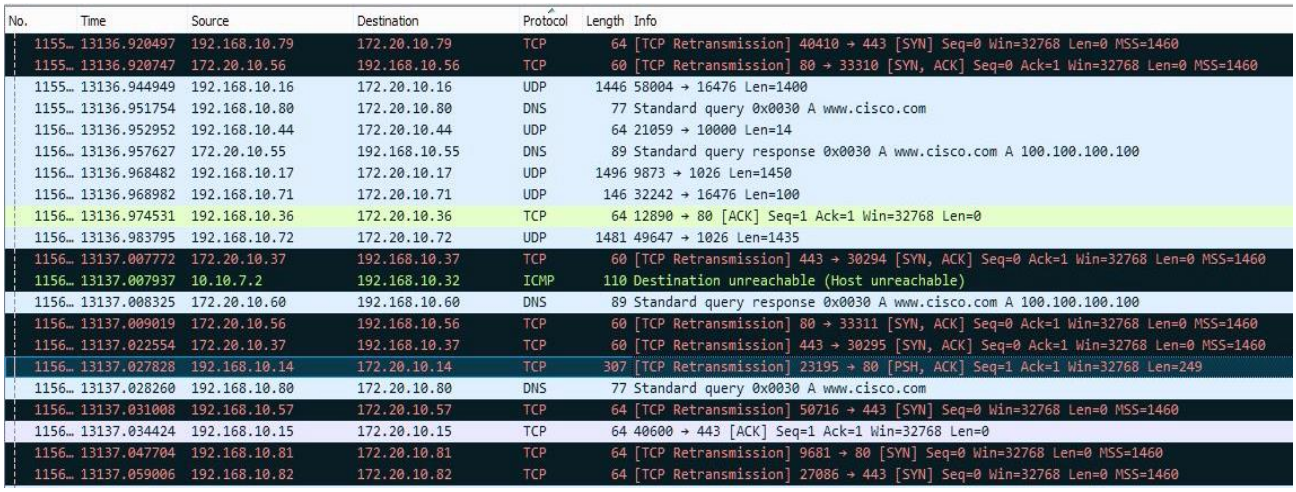


Figura 20-2: Tráfico generado por TRex

Fuente: Loza, J. 2021

En la figura 20-2 también se observan distintas estadísticas del tráfico las cuales serán utilizadas para analizar el rendimiento de la red, sin embargo para algunos parámetros se realizó la captura de paquetes mediante wireshark y realizar el análisis del comportamiento de la red para el tráfico global y por aplicaciones.



No.	Time	Source	Destination	Protocol	Length	Info
1155…	13136.920497	192.168.10.79	172.20.10.79	TCP	64	[TCP Retransmission] 40410 → 443 [SYN] Seq=0 Win=32768 Len=0 MSS=1460
1155…	13136.920747	172.20.10.56	192.168.10.56	TCP	60	[TCP Retransmission] 80 → 33310 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460
1155…	13136.944949	192.168.10.16	172.20.10.16	UDP	1446	58004 → 16476 Len=1400
1155…	13136.951754	192.168.10.80	172.20.10.80	DNS	77	Standard query 0x0030 A www.cisco.com
1156…	13136.952952	192.168.10.44	172.20.10.44	UDP	64	21059 → 10000 Len=14
1156…	13136.957627	172.20.10.55	192.168.10.55	DNS	89	Standard query response 0x0030 A www.cisco.com A 100.100.100.100
1156…	13136.968482	192.168.10.17	172.20.10.17	UDP	1496	9873 → 1026 Len=1450
1156…	13136.968982	192.168.10.71	172.20.10.71	UDP	146	32242 → 16476 Len=100
1156…	13136.974531	192.168.10.36	172.20.10.36	TCP	64	12890 → 80 [ACK] Seq=1 Ack=1 Win=32768 Len=0
1156…	13136.983795	192.168.10.72	172.20.10.72	UDP	1481	49647 → 1026 Len=1435
1156…	13137.007772	172.20.10.37	192.168.10.37	TCP	60	[TCP Retransmission] 443 → 30294 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460
1156…	13137.007937	10.10.7.2	192.168.10.32	ICMP	110	Destination unreachable (Host unreachable)
1156…	13137.008325	172.20.10.60	192.168.10.60	DNS	89	Standard query response 0x0030 A www.cisco.com A 100.100.100.100
1156…	13137.009019	172.20.10.56	192.168.10.56	TCP	60	[TCP Retransmission] 80 → 33311 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460
1156…	13137.022554	172.20.10.37	192.168.10.37	TCP	60	[TCP Retransmission] 443 → 30295 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0 MSS=1460
1156…	13137.027828	192.168.10.14	172.20.10.14	TCP	307	[TCP Retransmission] 23195 → 80 [PSH, ACK] Seq=1 Ack=1 Win=32768 Len=249
1156…	13137.028260	192.168.10.80	172.20.10.80	DNS	77	Standard query 0x0030 A www.cisco.com
1156…	13137.031008	192.168.10.57	172.20.10.57	TCP	64	[TCP Retransmission] 50716 → 443 [SYN] Seq=0 Win=32768 Len=0 MSS=1460
1156…	13137.034424	192.168.10.15	172.20.10.15	TCP	64	40600 → 443 [ACK] Seq=1 Ack=1 Win=32768 Len=0
1156…	13137.047704	192.168.10.81	172.20.10.81	TCP	64	[TCP Retransmission] 9681 → 80 [SYN] Seq=0 Win=32768 Len=0 MSS=1460
1156…	13137.059006	192.168.10.82	172.20.10.82	TCP	64	[TCP Retransmission] 27086 → 443 [SYN] Seq=0 Win=32768 Len=0 MSS=1460

Figura 21-2: Captura de paquetes generados por TRex

Fuente: Loza, J. 2021

En la figura 21-2 se puede observar una captura de wireshark en la cual se puede ver el flujo de los distintos servicios entre el pool de clientes y servidores configurados.

2.5 Integración del controlador SDN

2.5.1 Configuraciones

Se ha empleado el controlador Opendaylight para integrarlo a la red, este se puede instalar de diversas formas, en este caso se ha implementado un Docker disponible de forma oficial en el repositorio *https://hub.docker.com/r/opendaylight/odl* en la web de docker. De la misma forma que TRex, se ha instalado una instancia en GNS3 con la imagen de Opendaylight para poder utilizarlo, una vez se ha instalado se muestra una interfaz para utilizar el software como se muestra en la figura 22-2.

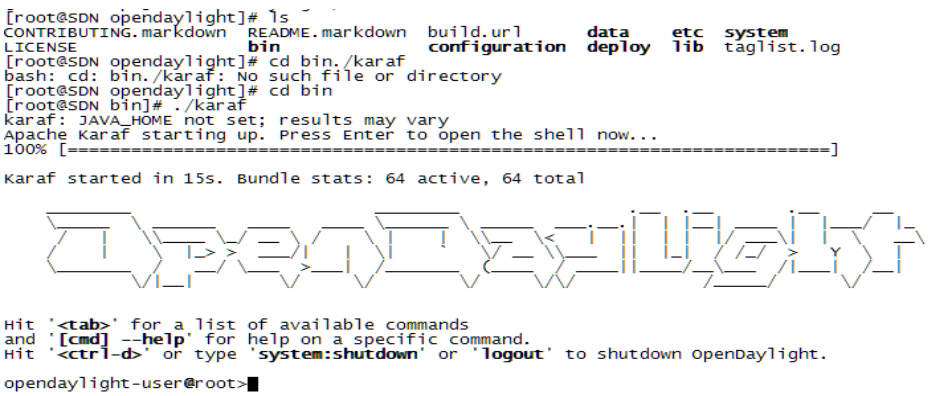


Figura 22-2: Interfaz Docker Opendaylight

Fuente: Loza, J. 2021

Opendaylight es un controlador que ofrece distintas herramientas, para ello es necesario activarlas, para el presente trabajo se han instalado las siguientes herramientas para realizar los objetivos planteados desde el terminal mostrado en la figura 22-2.

feature:install odl-aaa-authn
feature:install odl-restconf-all
feature:install odl-dlux-core
feature:install odl-dluxapps-yangman
feature:install odl-dluxapps-yangui
feature:install odl-l2switch-all
feature:install webconsole
feature:install odl-mdsal-apidocs
feature:install odl-netconf-connector-all

Estas herramientas habilitan las interfaces Northbound con Restful y Southbound con NETCONF, propiedades para que el controlador pueda comunicar las interfaces entre sí y el servicio web mediante el cual se puede tener una visión gráfica del controlador Opendaylight como se muestra en la figura 23-2.

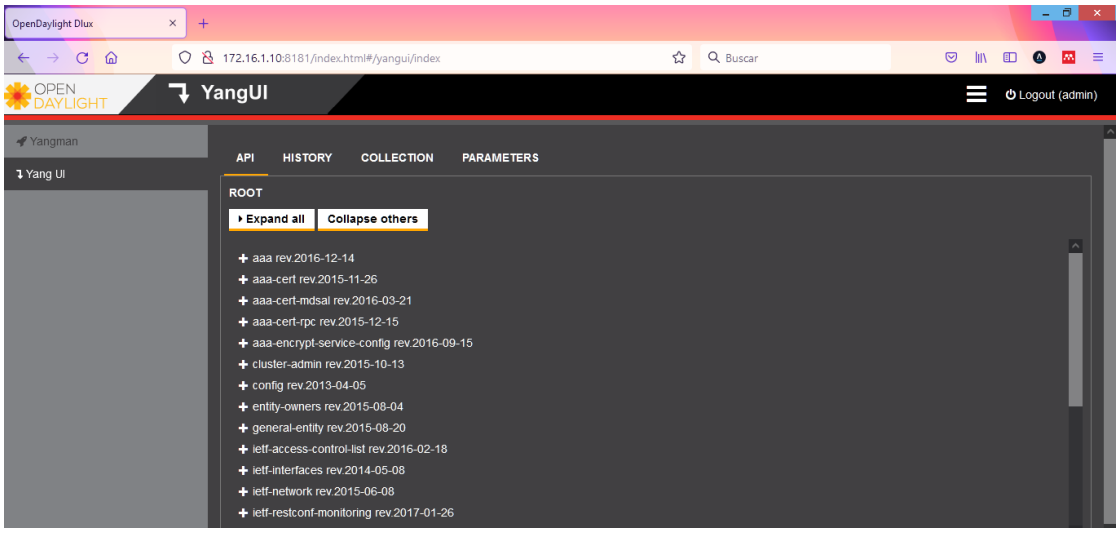


Figura 23-2: Interfaz Gráfica Web de Opendaylight

Fuente: Loza, J. 2021

En primer lugar se debe configurar los routers para poder iniciar la sesión Netconf, esta configuración se puede observar en el ANEXO B. Una vez están listos para comunicarse con el controlador, se inicia la conexión desde POSTMAN el cual es una aplicación para realizar

solicitudes mediante el protocolo RESTful, en la figura 24-2 se muestra la configuración de la solicitud para iniciar la sesión NETCONF entre PE1 y Opendaylight.

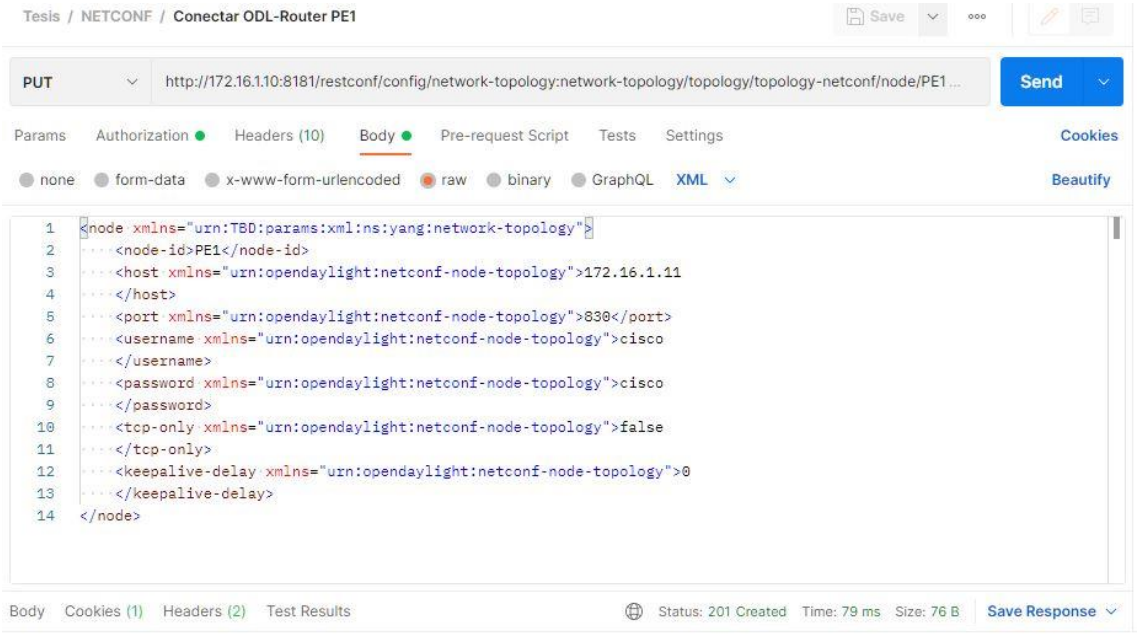


Figura 24-2: Solicitud PUT para iniciar la sesión NETCONF

Fuente: Loza, J. 2021

Se puede observar en la parte izquierda baja de la figura 25-2 que se genera un código de estado 201, esto quiere decir que la solicitud se ha realizado correctamente, sin embargo es necesario comprobarlo en el router, esta comprobación se puede ver en la figura.

```
RP/0/0/CPU0:PE-1#sh netconf-yang clients
Tue Oct 19 03:08:22.514 UTC
Netconf clients
client session ID|     NC version|    client connect time|        last OP time|        last OP type|    <lock>|
       2742090136|            1.1|          0d  1h  3m 56s|            08:25:16|              unlock|        No|
RP/0/0/CPU0:PE-1#sh netconf-yang statistics
Tue Oct 19 03:08:26.354 UTC
Summary statistics
                        # requests|            total time|   min time per request|   max time per request|   avg time per request|
other                            0|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
close-session                    0|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
kill-session                     0|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
get-schema                     545|       0h  0m 44s 886ms|       0h  0m  0s    9ms|       0h  0m  0s 759ms|       0h  0m  0s   82ms|
get                              1|       0h  0m  0s   59ms|       0h  0m  0s   59ms|       0h  0m  0s   59ms|       0h  0m  0s   59ms|
get-config                       6|       0h  0m  0s    9ms|       0h  0m  0s    0ms|       0h  0m  0s    9ms|       0h  0m  0s    1ms|
edit-config                      2|       0h  0m  0s    9ms|       0h  0m  0s    0ms|       0h  0m  0s    9ms|       0h  0m  0s    4ms|
commit                           1|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
cancel-commit                    0|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
lock                             6|       0h  0m  0s   29ms|       0h  0m  0s    0ms|       0h  0m  0s   19ms|       0h  0m  0s    4ms|
unlock                           6|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
discard-changes                  5|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
validate                         0|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
xml parse                      572|       0h  0m 14s 928ms|       0h  0m  0s    0ms|       0h  0m  0s 149ms|       0h  0m  0s   26ms|
netconf processor              572|       0h  0m 44s 996ms|       0h  0m  0s    0ms|       0h  0m  0s 759ms|       0h  0m  0s   78ms|
YFW                             26|       0h  0m  3s 199ms|       0h  0m  0s    0ms|       0h  0m  2s 129ms|       0h  0m  0s  123ms|
pending requests                 0|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
invoke rpc                       0|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
copy-config                      0|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|       0h  0m  0s    0ms|
Statistics for session with ID: 2742090136
```

Figura 25-2: Comprobación de la sesión NETCONF en PE1

Fuente: Loza, J. 2021

En esta tabla de estadísticas se puede observar las diferentes operaciones que se han realizado mediante NETCONF.

Una de las herramientas de YANG UI es poder visualizar la topología de la red, por lo tanto cuando un router se conecta al controlador ODL, se debe visualizar al mismo como está en la figura 26-2 al añadir a PE1.

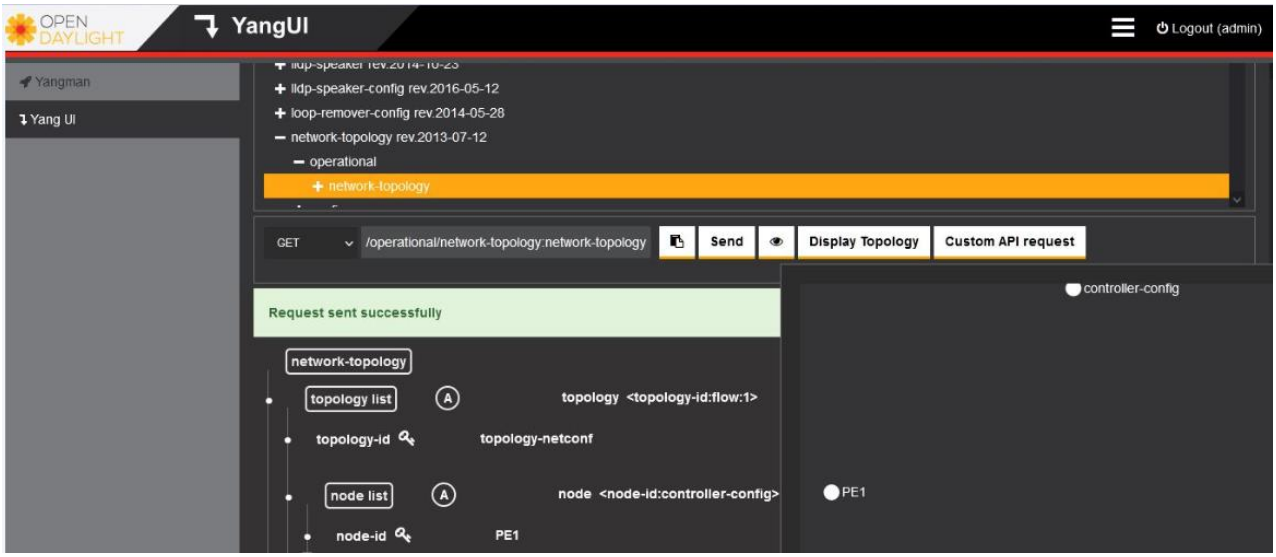


Figura 26-2: Muestra de PE1 en la topología mediante la GUI de ODL

Fuente: Loza, J. 2021

CAPÍTULO III

3. ANÁLISIS DE RESULTADOS

3.1 Rendimiento de la red

3.1.1 Prueba 1

3.1.1.1 Resultados por enlace

Los enlaces en los que se ha capturado el tráfico para el posterior análisis corresponden a los que el protocolo de enrutamiento utiliza para el envío de tráfico. En esta primera prueba se ha inyectado 106,4 Mbytes de tráfico, la media de las 5 pruebas ofrece los resultados que se muestran en la tabla 1-3.

Tabla 1-3: Resultados prueba 1

Enlace	Pérdida de paquetes (%)	Latencia(ms)	Jitter (ms)
PE1-P1	4,95	20,77	224,47
PE1-P2	6,71	29,03	245,88
P1-P2	2,70	16,41	116,11
P1-P3	4,42	22,41	230,34
P2-P3	3,04	16,67	137,49
PE2-P3	5,44	26,56	189,86
PE2-P2	6,37	20,18	206,10

Realizado por: Loza, J. 2021

3.1.1.2 Resultado de la red

El rendimiento total de la red refleja los resultados de la tabla 2-3.

Tabla 2-3: Rendimiento total de la red en la prueba 1

Pérdida de paquetes (%)	Latencia (ms)	Jitter (ms)
4,80	21,72	192,89

Realizado por: Loza, J. 2021

En esta prueba, el tráfico en toda la red está distribuido como se muestra en la figura 1-3. Como se puede observar el tráfico de servicios Web es el que genera mayor cantidad de datos mientras que el tráfico de Streaming en el que se encuentra también VoIP es el segundo.

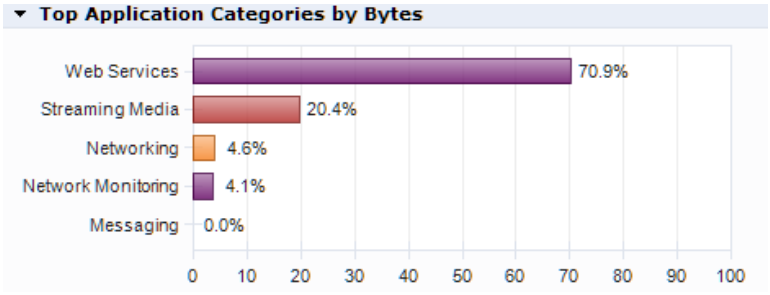


Figura 1-3: Tráfico por servicios en la red

Fuente: Loza, J. 2021

En la figura 2-3 se muestra el tráfico por protocolo, en el cual se puede apreciar la diferencia entre el tráfico de Streaming (RTP, RTSP) y el tráfico de VoIP (SIP, RTCP)

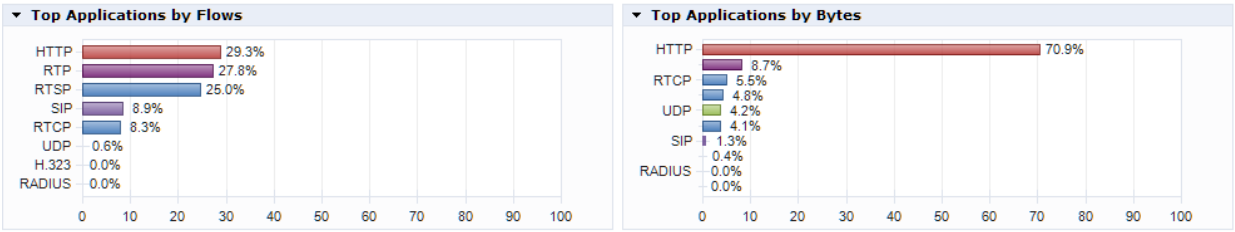


Figura 2-3: Tráfico por protocolos en la red

Fuente: Loza, J. 2021

En la figura 3-3 se observa la calidad de las llamadas en la red, este parámetro es importante ya que contiene información de todas las conexiones de *media*, incluyendo streaming y VoIP.

Figura 3-3: Calidad de las llamadas (VoIP y Streaming)

Fuente: Loza, J. 2021

3.1.2 Prueba 2

3.1.2.1 Resultados por enlace

En esta segunda prueba se ha inyectado 319,2 Mbytes de tráfico, los resultados se muestran en la tabla 2-3.

Tabla 3-3: Resultados prueba 2

Enlace	Pérdida de paquetes (%)	Latencia(ms)	Jitter (ms)
PE1-P1	10,04	59,65	316,47
PE1-P2	13,25	66,30	264,19
P1-P2	8,512	40,518	282,14
P1-P3	10,21	51,99	321,54
P2-P3	8,81	38,84	307,51
PE2-P3	10,802	56,73	301,90
PE2-P2	12,13	63,74	311,86

Realizado por: Loza, J. 2021

3.1.2.2 Resultado de la red

El rendimiento total de la red refleja los resultados de la tabla 4-3.

Tabla 4-3: Rendimiento total de la red en la prueba 2

Pérdida de paquetes (%)	Latenica (ms)	Jitter (ms)
10.54	53.97	300,80

Realizado por: Loza, J. 2021

En esta prueba, el tráfico en toda la red está distribuido como se muestra en la figura 4-3. Cómo se puede observar el tráfico de servicios Web es el que genera mayor cantidad de datos mientras que el tráfico de Streaming en el que se encuentra también VoIP es el segundo.

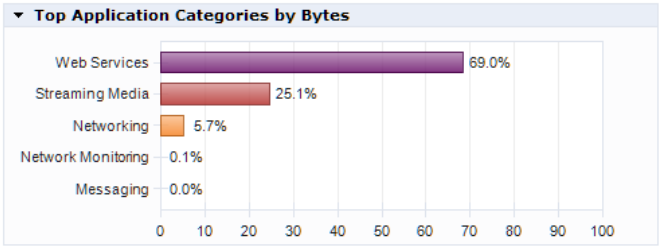


Figura 4-3: Tráfico por servicios en la red

Fuente: Loza, J. 2021

En la figura 5-3 se muestra el tráfico por protocolo, en el cual se puede apreciar la diferencia entre el tráfico de Streaming (RTP, RTSP) y el tráfico de VoIP (SIP, RTCP)

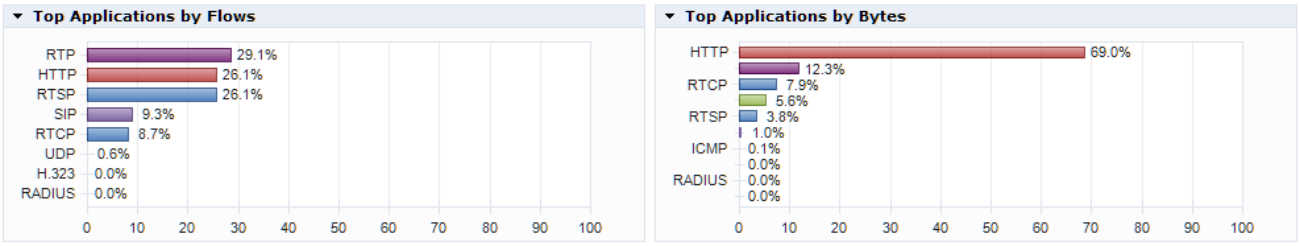


Figura 5-3: Tráfico por protocolos en la red

Fuente: Loza, J. 2021

En la figura 6-3 se observa la calidad de las llamadas cuando el tráfico aumenta en esta segunda prueba.

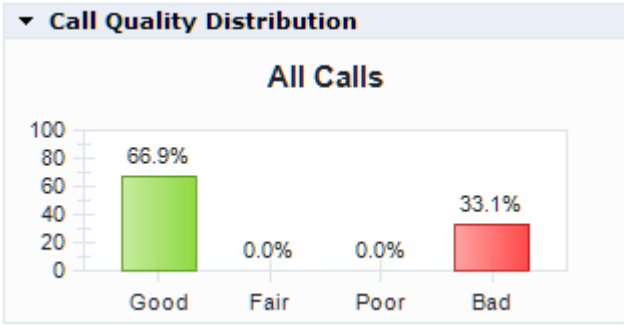


Figura 6-3: Calidad de las llamadas (VoIP y Streaming)

Fuente: Loza, J. 2021

3.1.3 Prueba 3

3.1.3.1 Resultados por enlace

En esta tercera prueba se ha inyectado 638,4 Mbytes de tráfico, los resultados se muestran en la tabla 5-3.

Tabla 5-3: Resultados prueba 3

Enlace	Pérdida de paquetes (%)	Latencia(ms)	Jitter (ms)
PE1-P1	29,20	285,69	743,30
PE1-P2	29,61	309,08	767,52
P1-P2	10,26	76,774	351,95
P1-P3	25,88	264,12	688,03
P2-P3	15,85	214,87	611,95
PE2-P3	26,32	313,24	758,92
PE2-P2	30,50	290,87	724,87

Realizado por: Loza, J. 2021

3.1.3.2 Resultado de la red

El rendimiento total de la red refleja los resultados de la tabla 6-3.

Tabla 6-3: Rendimiento total de la red en la prueba 3

Pérdida de paquetes (%)	Latencia (ms)	Jitter (ms)
23,95	250,66	663,79

Realizado por: Loza, J. 2021

En esta prueba, el tráfico en toda la red está distribuido como se muestra en la figura 7-3. Cómo se puede observar el tráfico de servicios Web es el que genera mayor cantidad de datos mientras que el tráfico de Streaming en el que se encuentra también VoIP es el segundo.

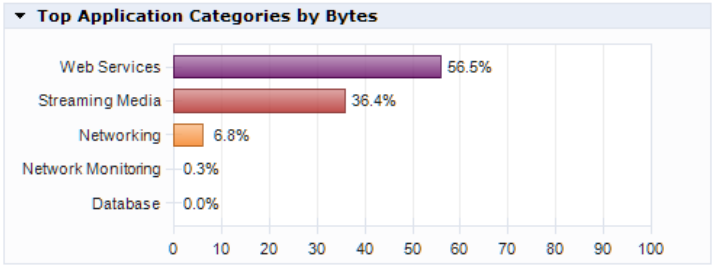


Figura 7-3: Tráfico por servicios en la red

Fuente: Loza, J. 2021

En la figura 8-3 se muestra el tráfico por protocolo, en el cual se puede apreciar la diferencia entre el tráfico de Streaming (RTP, RTSP) y el tráfico de VoIP (SIP, RTCP)

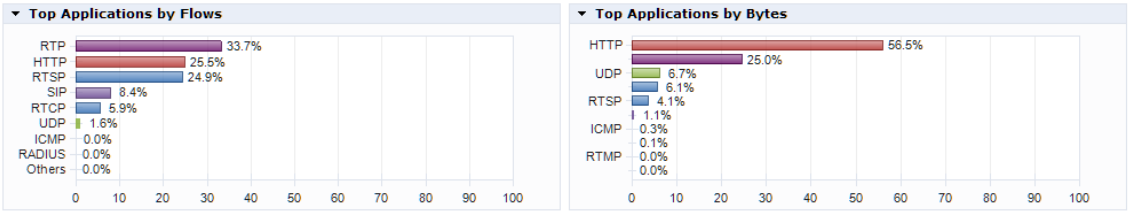


Figura 8-3: Tráfico por protocolos en la red

Fuente: Loza, J. 2021

En la figura 9-3 se observa la calidad de las llamadas que se tiene en la tercera prueba.

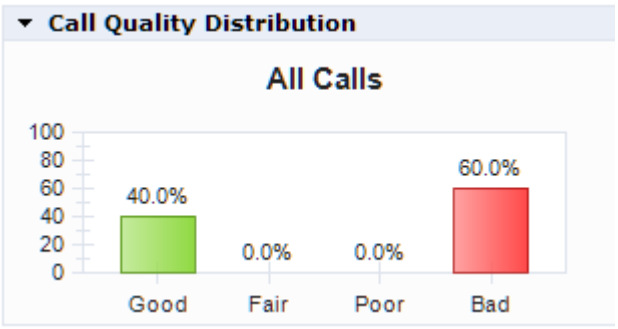


Figura 9-3: Calidad de las llamadas (VoIP y Streaming)

Fuente: Loza, J. 2021

3.1.4 Prueba 4

3.1.4.1 Resultados por enlace

En esta prueba se ha inyectado 744,8 Mbytes de tráfico en la red, y como se muestra en la figura 10-3, se puede considerar que la red ha caído debido a que uno de los routers se ha quedado sin memoria, se ha realizado todas las muestras y en cada una de ellas al menos un router presenta este problema.

```
RP/0/0/CPU0:Oct 21 06:14:00.909 : ifmgr[228]: %PKT_INFRA-LINK-3-UPDOWN : #ALMLOC#: Interface GigabitEthernet0/0/0/0, changed state to Up
RP/0/0/CPU0:Oct 21 06:14:00.909 : ifmgr[228]: %PKT_INFRA-LINK-3-UPDOWN : #ALMLOC#: Interface GigabitEthernet0/0/0/1, changed state to Up
RP/0/0/CPU0:Oct 21 06:14:00.909 : ifmgr[228]: %PKT_INFRA-LINEPROTO-5-UPDOWN : #ALMLOC#: Line protocol on Interface GigabitEthernet0/0/0/0, changed state to Up
RP/0/0/CPU0:Oct 21 06:14:00.909 : ifmgr[228]: %PKT_INFRA-LINK-3-UPDOWN : #ALMLOC#: Interface GigabitEthernet0/0/0/2, changed state to Up
RP/0/0/CPU0:Oct 21 06:14:00.919 : ifmgr[228]: %PKT_INFRA-LINEPROTO-5-UPDOWN : #ALMLOC#: Line protocol on Interface GigabitEthernet0/0/0/1, changed state to Up
RP/0/0/CPU0:Oct 21 06:14:00.919 : ifmgr[228]: %PKT_INFRA-LINEPROTO-5-UPDOWN : #ALMLOC#: Line protocol on Interface GigabitEthernet0/0/0/2, changed state to Up
RP/0/0/CPU0:Oct 21 06:14:11.538 : ospf[1018]: %ROUTING-OSPF-5-ADJCHG : Process 10, Nbr 1.1.1.1 on GigabitEthernet0/0/0/2 in area 0 from LOADING to FULL, Loading Done,
vrf default vrfid 0x60000000
RP/0/0/CPU0:Oct 21 06:22:04.955 : mediasvr[57]: %MEDIA-MEDIASVR-3-CORRUPT_FILE : File '/disk0/iosxr-infra-6.3.1/sbin/wdsysmon' is corrupted . Error - No such file or d
irectory
RP/0/0/CPU0:Oct 21 06:22:05.025 : sysmgr[75]: %OS-SYSMGR-3-ERROR : wdsysmon(1) (jid 412) can not be started, restart scheduled ('Subsystem(1786)' detected the 'warning
' condition 'Code(9)': No s
RP/0/0/CPU0:Oct 21 06:22:05.145 : sysmgr[75]: %OS-SYSMGR-3-ERROR : wdsysmon(1) (jid 412) exited, will be respawned with a delay (slow-restart)
RP/0/0/CPU0:Oct 21 06:22:05.165 : sysmgr[75]: %OS-SYSMGR-3-ERROR : wdsysmon(412) (fail count 1) will be respawned in 10 seconds
RP/0/0/CPU0:Oct 21 06:22:03.276 : sysmgr[75]: qnet(1) (jid 67) (pid 245794) (fail_count 0) abnormally terminated, restart scheduled

RP/0/0/CPU0:Oct 21 06:22:04.695 : sysmgr[75]: wdsysmon(1) (jid 412) (pid 307245) (fail_count 0) abnormally terminated, restart scheduled

RP/0/0/CPU0:Oct 21 06:22:05.015 : sysmgr[75]: Cannot spawn wdsysmon(1) (jid 412) exec=wdsysmon (No such file or directory)

RP/0/0/CPU0:Oct 21 06:22:11.635 : netio[314]: %INSTALL-PKG_PLAT-7-OPEN_FAIL : The packaging infrastructure was unable to open a file (/nvram:/rommon-vars/IOSXRV_FABRIC
_MTU), error: No such file or directory.
RP/0/0/CPU0:Oct 21 06:22:15.185 : cfgmgr-rp[155]: %HA-HA_WD_LIB-7-SVR_REG : wd_disk_client_reconnect: failed to register with WDSysMon server: 'Subsystem(753)' detecte
d the 'fatal' condition 'Code(2)'  : pkg/bin/cfgmgr-rp : (PID=315469) :  -Traceback= c654593 c0ba96a c0b832a c0b9c6b c0b7c94 420c130 c109070
RP/0/0/CPU0:Oct 21 06:22:15.185 : mediasvr[57]: %MEDIA-MEDIASVR-3-CORRUPT_FILE : File '/disk0/iosxr-infra-6.3.1/sbin/wdsysmon' is corrupted . Error - No such file or d
irectory
RP/0/0/CPU0:Oct 21 06:22:15.225 : sysmgr[75]: %OS-SYSMGR-3-ERROR : wdsysmon(1) (jid 412) can not be started, restart scheduled ('Subsystem(1786)' detected the 'warning
' condition 'Code(9)': No s
RP/0/0/CPU0:Oct 21 06:22:15.245 : sysmgr[75]: %OS-SYSMGR-3-ERROR : wdsysmon(1) (jid 412) exited, will be respawned with a delay (slow-restart)
RP/0/0/CPU0:Oct 21 06:22:15.245 : sysmgr[75]: %OS-SYSMGR-3-ERROR : wdsysmon(412) (fail count 2) will be respawned in 10 seconds
RP/0/0/CPU0:Oct 21 06:22:15.225 : sysmgr[75]: Cannot spawn wdsysmon(1) (jid 412) exec=wdsysmon (No such file or directory)

RP/0/0/CPU0:Oct 21 06:22:24.334 : netio[314]: %INSTALL-PKG_PLAT-7-OPEN_FAIL : The packaging infrastructure was unable to open a file (/nvram:/rommon-vars/IOSXRV_FABRIC
_MTU), error: No such file or directory.
RP/0/0/CPU0:Oct 21 06:22:25.264 : mediasvr[57]: %MEDIA-MEDIASVR-3-CORRUPT_FILE : File '/disk0/iosxr-infra-6.3.1/sbin/wdsysmon' is corrupted . Error - No such file or d
irectory
RP/0/0/CPU0:Oct 21 06:22:25.264 : sysmgr[75]: %OS-SYSMGR-3-ERROR : wdsysmon(1) (jid 412) can not be started, restart scheduled ('Subsystem(1786)' detected the 'warning
' condition 'Code(9)': No s
RP/0/0/CPU0:Oct 21 06:22:25.264 : sysmgr[75]: %OS-SYSMGR-3-ERROR : wdsysmon(1) (jid 412) exited, will be respawned with a delay (slow-restart)
RP/0/0/CPU0:Oct 21 06:22:25.264 : sysmgr[75]: %OS-SYSMGR-3-ERROR : wdsysmon(412) (fail count 3) will be respawned in 10 seconds
RP/0/0/CPU0:Oct 21 06:22:25.264 : sysmgr[75]: Cannot spawn wdsysmon(1) (jid 412) exec=wdsysmon (No such file or directory)
```

Figura 10-3: Fallo en P1

Fuente: Loza, J. 2021

3.2 Rendimiento de SR

3.2.1 Prueba 1

3.2.1.1 Resultados por enlace

En esta primera prueba se ha inyectado 106,4 Mbytes de tráfico, los resultados se muestran en la tabla 7-3.

Tabla 7-3: Resultados prueba 1

Enlace	Pérdida de paquetes (%)	Latencia(ms)	Jitter (ms)
PE1-P1	2,04	5,05	6.87
PE1-P2	3.47	5.76	6.53
P1-P2	0,95	1.99	4.35
P1-P3	1,6	3.96	5.93
P2-P3	1.12	1.95	5.65
PE2-P3	2.21	5.2	8.11
PE2-P2	2.26	5.59	7,83

Realizado por: Loza, J. 2021

3.2.1.2 Resultados de la red

El rendimiento total de la red refleja los resultados de la tabla 8-3.

Tabla 8-3: Rendimiento total de la red en la prueba 1

Pérdida de paquetes (%)	Latencia (ms)	Jitter (ms)
1,53	5,05	7,83

Realizado por: Loza, J. 2021

En esta prueba, el tráfico en toda la red está distribuido como se muestra en la figura 11-3. Cómo se puede observar el tráfico de servicios Web es el que genera mayor cantidad de datos mientras que el tráfico de Streaming en el que se encuentra también VoIP es el segundo.

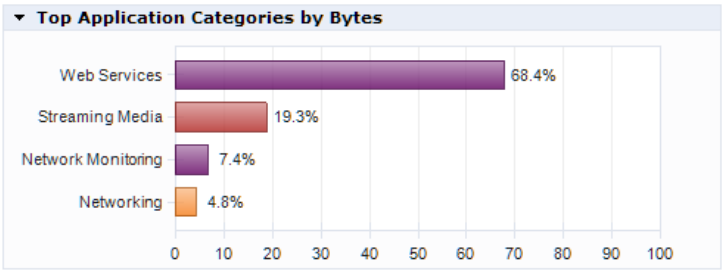


Figura 11-3: Tráfico por servicios en la red

Fuente: Loza, J. 2021

En la figura 12-3 se muestra el tráfico por protocolo, en el cual se puede apreciar la diferencia entre el tráfico de Streaming (RTP, RTSP) y el tráfico de VoIP (SIP, RTCP)

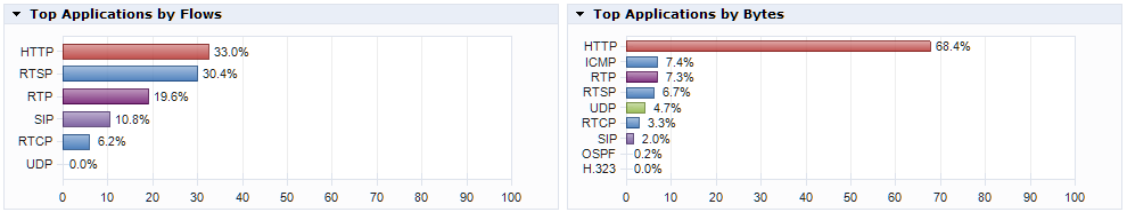


Figura 12-3: Tráfico por protocolos en la red

Fuente: Loza, J. 2021

En la figura 13-3 se observa la calidad de las llamadas en la primera prueba con Segment Routing.

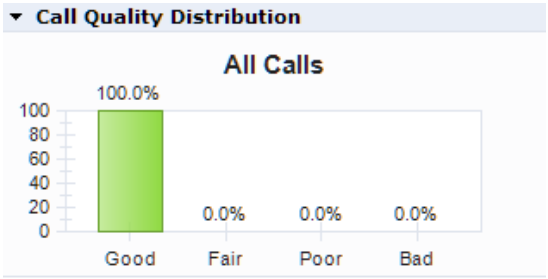


Figura 13-3: Calidad de las llamadas (VoIP y Streaming)

Fuente: Loza, J. 2021

3.2.2 Prueba 2

3.2.2.1 Resultados por enlace

En esta segunda prueba se ha inyectado 303,69 Mbytes de tráfico, los resultados se muestran en la tabla 9-3.

Tabla 9-3: Resultados prueba 2

Enlace	Pérdida de paquetes (%)	Latencia(ms)	Jitter (ms)
PE1-P1	6,90	27,82	33,62
PE1-P2	9,12	37,12	39,51
P1-P2	4,04	19,59	22,55
P1-P3	8,33	30,41	29,9
P2-P3	6,04	20,32	21,19
PE2-P3	7,47	29,30	38,82
PE2-P2	7,31	31,29	46,21

Realizado por: Loza, J. 2021

3.2.2.2 Resultados de la red

El rendimiento total de la red refleja los resultados de la tabla 10-3.

Tabla 10-3: Rendimiento total de la red en la prueba 2

Pérdida de paquetes (%)	Latencia (ms)	Jitter (ms)
7,03	27,98	33,11

Realizado por: Loza, J. 2021

En esta prueba, el tráfico en toda la red está distribuido como se muestra en la figura 14-3. Cómo se puede observar el tráfico de servicios Web es el que genera mayor cantidad de datos mientras que el tráfico de Streaming en el que se encuentra también VoIP es el segundo.

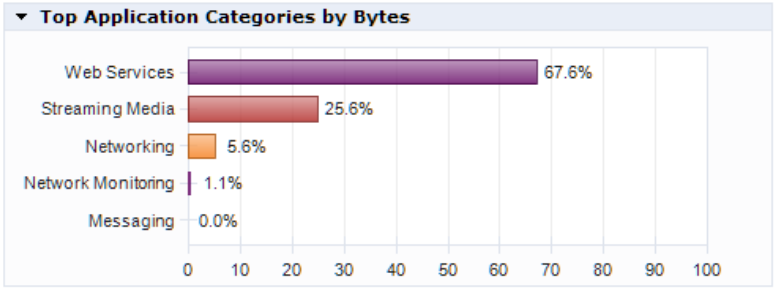


Figura 14-3: Tráfico por servicios en la red

Fuente: Loza, J. 2021

En la figura 15-3 se muestra el tráfico por protocolo, en el cual se puede apreciar la diferencia entre el tráfico de Streaming (RTP, RTSP) y el tráfico de VoIP (SIP, RTCP)

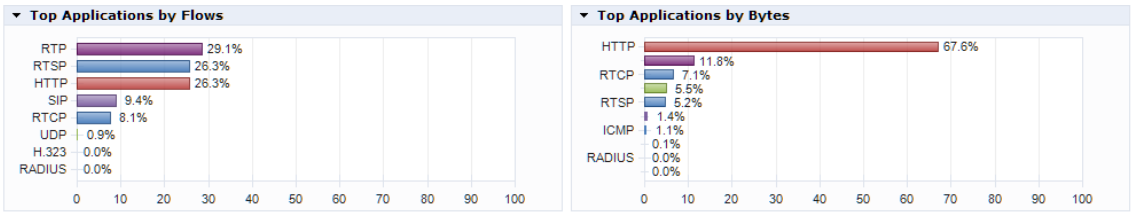


Figura 15-3: Tráfico por protocolos en la red

Fuente: Loza, J. 2021

En la figura 16-3 se observa el estado de las llamadas para la segunda prueba con segment routing.

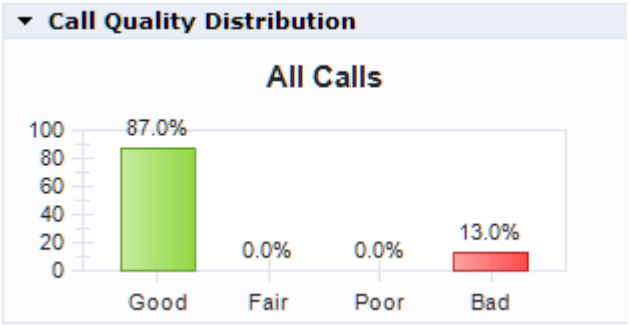


Figura 16-3: Calidad de las llamadas (VoIP y Streaming)

Fuente: Loza, J. 2021

3.2.3 Prueba 3

3.2.3.1 Resultados por enlace

En esta tercera prueba se ha inyectado 638,40 Mbytes de tráfico, los resultados se muestran en la tabla 11-3.

Tabla 11-3: Resultados prueba 3

Enlace	Pérdida de paquetes (%)	Latencia(ms)	Jitter (ms)
PE1-P1	11,99	52,62	197,96
PE1-P2	14,92	56,07	225,44
P1-P2	8,32	38,87	124,61
P1-P3	12,98	50,45	179,29
P2-P3	10,08	42,48	123,68
PE2-P3	10.02	67,76	204,08
PE2-P2	12,59	46,02	208,62

Realizado por: Loza, J. 2021

3.2.3.2 Resultados de la red

El rendimiento total de la red refleja los resultados de la tabla 12-3.

Tabla 12-3: Rendimiento total de la red en la prueba 3

Pérdida de paquetes (%)	Latencia (ms)	Jitter (ms)
11,81	50,61	180,53

Realizado por: Loza, J. 2021

En esta prueba, el tráfico en la red está distribuido como se muestra en la figura 17-3. Cómo se puede observar el tráfico de servicios Web es el que genera mayor cantidad de datos mientras que el tráfico de Streaming en el que se encuentra también VoIP es el segundo.

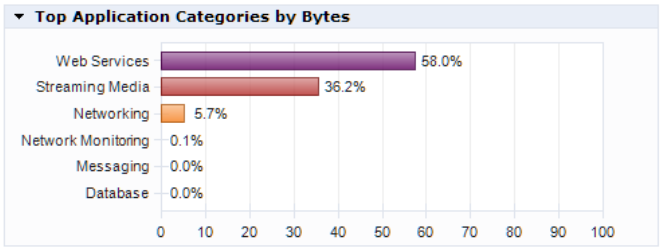


Figura 17-3: Tráfico por servicios en la red

Fuente: Loza, J. 2021

En la figura 18-3 se muestra el tráfico por protocolo, en el cual se puede apreciar la diferencia entre el tráfico de Streaming (RTP, RTSP) y el tráfico de VoIP (SIP, RTCP)

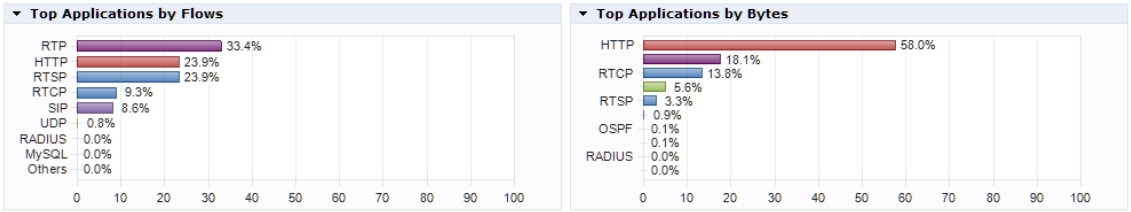


Figura 18-3: Tráfico por protocolos en la red

Fuente: Loza, J. 2021

En la figura 19-3 se puede observar la calidad de las llamadas que fluyen por la red.

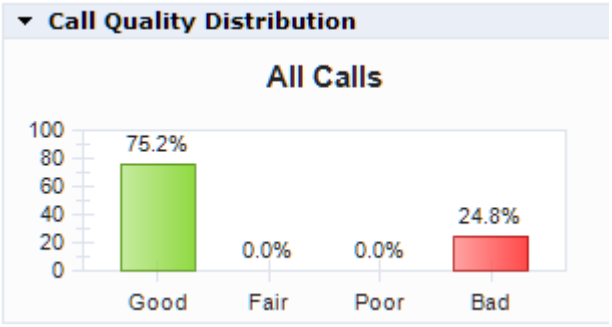


Figura 19-3: Calidad de las llamadas (VoIP y Streaming)

Fuente: Loza, J. 2021

3.2.4 Prueba 4

3.2.4.1 Resultados por enlace

En esta cuarta prueba se ha inyectado 744,8 Mbytes de tráfico, los resultados se muestran en la tabla 13-3.

Tabla 13-3: Resultados prueba 4

Enlace	Pérdida de paquetes (%)	Latencia(ms)	Jitter (ms)
PE1-P1	20,25	72,37	332,41
PE1-P2	20,14	75,35	359,12
P1-P2	11,80	62,98	263,58
P1-P3	17,07	68,94	303,86
P2-P3	17,19	57,39	236,63
PE2-P3	18,54	78,15	350,79
PE2-P2	16,39	67,14	301,278

Realizado por: Loza, J. 2021

3.2.4.2 Resultados de la red

El rendimiento total de la red refleja los resultados de la tabla 14-3.

Tabla 14-3: Rendimiento total de la red en la prueba 4

Pérdida de paquetes (%)	Latencia (ms)	Jitter (ms)
17,34	68,90	306,81

Realizado por: Loza, J. 2021

En esta prueba, el tráfico en toda la red está distribuido como se muestra en la figura 20-3. Cómo se puede observar el tráfico de servicios Web es el que genera mayor cantidad de datos mientras que el tráfico de Streaming en el que se encuentra también VoIP es el segundo.

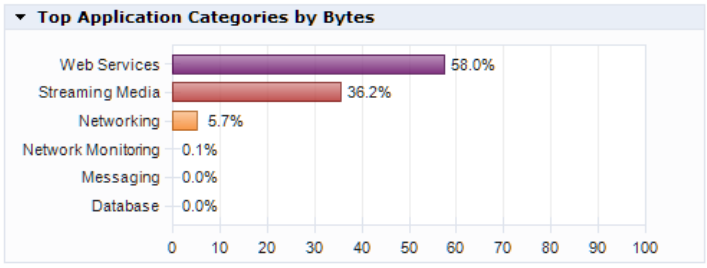


Figura 20-3: Tráfico por servicios en la red

Fuente: Loza, J. 2021

En la figura 21-3 se muestra el tráfico por protocolo, en el cual se puede apreciar la diferencia entre el tráfico de Streaming (RTP, RTSP) y el tráfico de VoIP (SIP, RTCP)

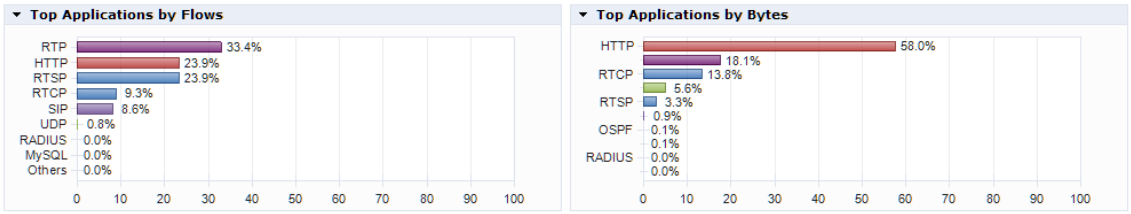


Figura 21-3: Tráfico por protocolos en la red

Fuente: Loza, J. 2021

En la figura 22-3 se observa la calidad de las llamadas para la cuarta prueba con Segment Routing.

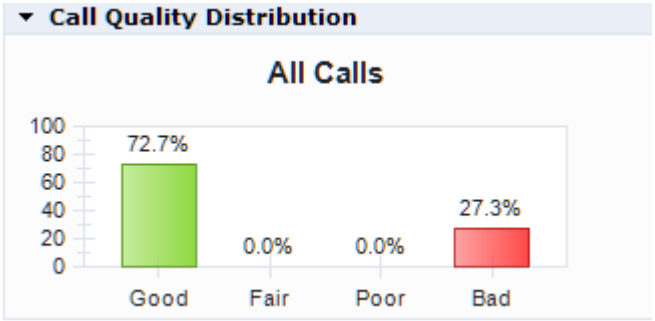


Figura 22-3: Calidad de las llamadas (VoIP y Streaming)

Fuente: Loza, J. 2021

3.3 Comparación de resultados

En primer lugar se realiza un promedio de todas las pruebas en OSPF como se muestra en la tabla 14-3, mientras que en la tabla 15-3 se muestran los de Segment Routing.

Tabla 14-3: Parámetros obtenidos al realizar las pruebas

Prueba	Pérdida de paquetes (%)	Latencia (ms)	Jitter (ms)
106,4 Mb	4,80	21,72	192,89
319,2 Mb	10,54	53,97	300,80
638,4 Mb	23,95	250,66	663,79

Realizado por: Loza, J. 2021

Tabla 15-3: Parámetros obtenidos al realizar las pruebas

Prueba	Pérdida de paquetes (%)	Latencia (ms)	Jitter (ms)
106,4 Mb	1,53	5,05	7,83
319,2 Mb	7,03	27,98	33,11
638,4 Mb	11,81	50,61	180,53
744,8 Mb	17,34	68,90	306,81

Realizado por: Loza, J. 2021

Para realizar un análisis de los resultados y obtener conclusiones adecuadas se ha utilizado una prueba estadística, debido a la naturaleza de los datos se puede realizar la prueba Wilcoxon o Mann-Whitney, las cuales son utilizadas para la comparación de muestras que tienen una distribución de datos libre, sin embargo Mann-Whitney realiza la comparación de dos muestras independientes el cual no es el caso, es por ello que se utiliza la prueba Wilcoxon mediante el software de análisis estadístico SPSS. Esta prueba se utiliza para comparar dos mediciones de rangos relacionados y determinar que la diferencia no se deba al azar. Para la regla de decisión se establece que si el nivel de significancia es menor o igual a 0.05 se rechaza la hipótesis Ho.

Para el presente trabajo se establece la hipótesis Ho como la no existencia de diferencia para los valores de latencia entre OSPF y SR.

Debido a que se realizan diferentes pruebas de estrés en la red, se ha realizado el análisis Wilconxon para cada una de ellas y para cada parámetro de red medido.
Una vez se tiene el archivo con los datos, se procede a realizar la prueba Wilconxon para cada parámetro, como se observa en la figura 23-3.

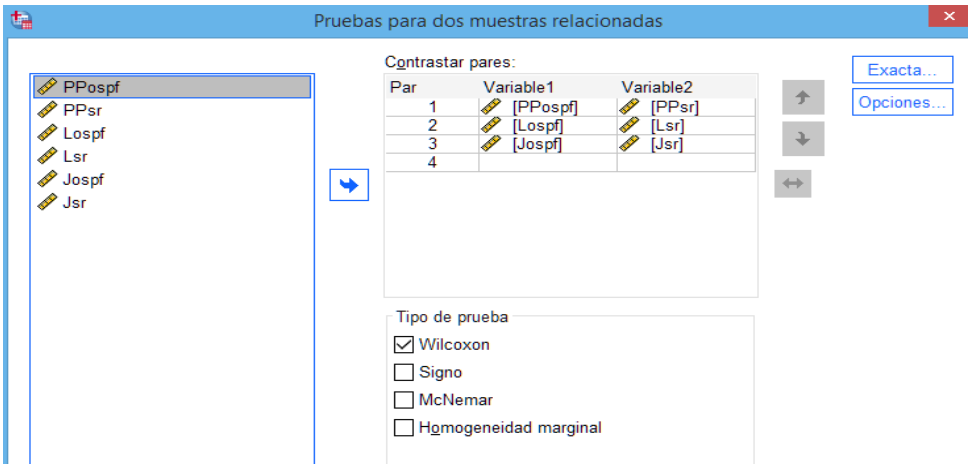


Figura 23-3: Prueba Wilconxon para los parámetros de medición de la prueba 1

Fuente: Loza, J. 2021

3.3.1 Wilconxon Prueba 1

Una vez realizado el test, se muestran los resultados en la figura 24-3 para cada parámetro de medición. Como se puede observar para los tres parámetros el resultado indica que $P \leq 0,05$ no cumple por lo tanto se rechaza la hipótesis nula, es decir para la prueba 1 existen diferencias entre el enrutamiento con Segment Routing y OSPF para los tres parámetros de medición de la red.

Estadísticos de prueba[a]

	PPsr - PPospf	Lsr - Lospf	Jsr - Jospf
Z	-2,023[b]	-2,023[b]	-2,023[b]
Sig. asin. (bilateral)	,04311	,04311	,04311

a. Prueba de rangos con signo de Wilcoxon

b. Se basa en rangos positivos.

Figura 24-3: Resultados del test Wilconxon para la prueba 1

Fuente: Loza, J. 2021

3.3.2 Wilconxon Prueba 2

Una vez realizado el test con los datos para la segunda prueba, se muestran los resultados en la figura 25-3 para cada parámetro de medición. Como se puede observar para los tres parámetros el resultado indica que $P \leq 0,05$ no cumple por lo tanto se rechaza la hipótesis nula, es decir para la prueba 2 al igual que sucede en la prueba 1 existen diferencias entre el enrutamiento con Segment Routing y OSPF para los tres parámetros de medición de la red.

Estadísticos de prueba[a]

	PPsr - PPospf	Lsr - Lospf	Jsr - Jospf
Z	-2,023[b]	-2,032[b]	-2,023[b]
Sig. asin. (bilateral)	,04311	,04217	,04311

a. Prueba de rangos con signo de Wilcoxon

b. Se basa en rangos positivos.

Figura 25-3: Resultados del test Wilconxon para la prueba 2

Fuente: Loza, J. 2021

3.3.3 Wilconxon Prueba 3

Una vez realizado el test con los datos de la tercera prueba, se muestran los resultados en la figura 26-3 para cada parámetro de medición. Como se puede observar para los tres parámetros el resultado indica que $P \leq 0,05$ no cumple por lo tanto se rechaza la hipótesis nula, es decir para la prueba 3 al igual que sucede en la prueba 1 y 2, existen diferencias entre el enrutamiento con Segment Routing y OSPF para los tres parámetros de medición de la red.

Estadísticos de prueba[a]

	PPsr - PPospf	Lsr - Lospf	Jsr - Jospf
Z	-2,023[b]	-2,023[b]	-2,023[b]
Sig. asin. (bilateral)	,04311	,04311	,04311

a. Prueba de rangos con signo de Wilcoxon

b. Se basa en rangos positivos.

Figura 26-3: Resultados del test Wilconxon para la prueba 3

Fuente: Loza, J. 2021

Debido a que en la prueba 4 para OSPF la red se cayó no se pudo obtener los parámetros de medición, sin embargo, debido a que Segment Routing no tuvo problemas en transportar el tráfico inyectado para la prueba 4, se puede establecer que en esta prueba al igual que ocurre en las anteriores, Segment Routing tiene mejor rendimiento que OSPF.

3.4 Gestión de la red con SDN

Una vez se ha establecido comunicación entre los routers y ODL/POSTMAN, se puede manipular las configuraciones remotamente de los routers, se utiliza POSTMAN ya que ofrece un servicio en la nube para almacenar los archivos de solicitudes además de una interfaz gráfica para poder realizar el manejo de los dispositivos de mejor manera, sin embargo esto se puede realizar desde ODL mediante la herramienta YANG UI y los diferentes modelos que se encuentran.

3.4.1 Creación de interfaces

En la figura 27-3 se puede observar la creación de una interfaz virtual para PE1 mediante el URN http://172.16.1.10:8181/restconf/config/network-topology:network-topology/topology/topology-netconf/node/PE1/yang-ext:mount/Cisco-IOS-XR-ifmgr-cfg:interface-configurations y con una operación POST, ya que se debe crear la configuración en el router PE1.

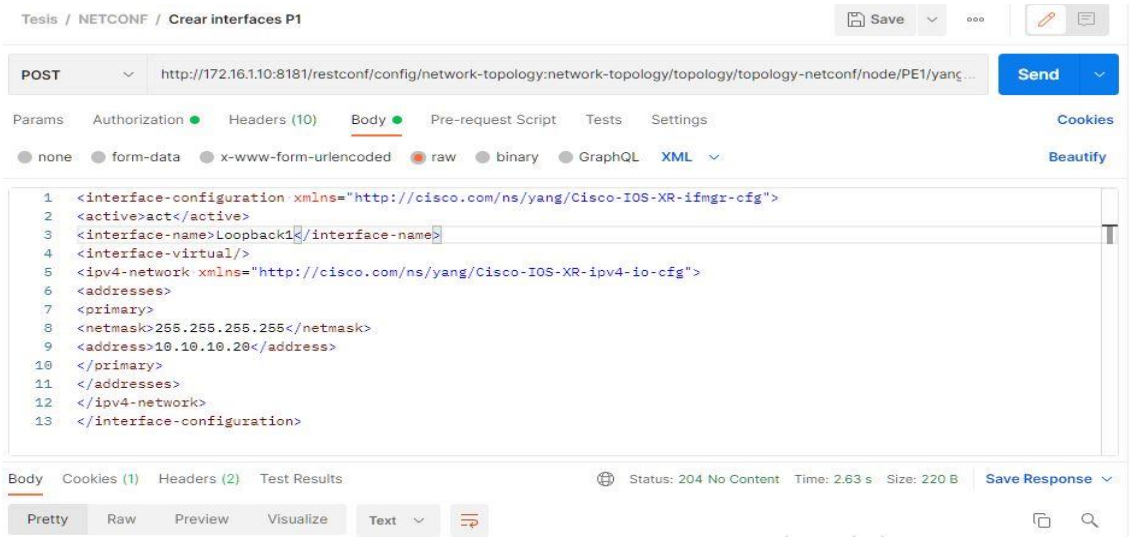


Figura 27-3: Solicitud POST para crear interfaces en PE1

Fuente: Loza, J. 2021

De igual forma se comprueba en el router PE1 que se ha creado la interfaz como se ve en la figura 28-3.

```
RP/0/0/CPU0:PE-1#sh ip int br
Tue Oct 19 03:03:53.533 UTC

Interface                  IP-Address      Status          Protocol Vrf-Name
Loopback0                  1.1.1.1         Up              Up       default
Loopback1                  10.10.10.20     Up              Up       default
MgmtEth0/0/CPU0/0          172.16.1.11     Up              Up       default
GigabitEthernet0/0/0/0     10.10.1.1       Up              Up       default
GigabitEthernet0/0/0/1     10.10.2.1       Up              Up       default
GigabitEthernet0/0/0/2     192.168.10.2    Up              Up       default
GigabitEthernet0/0/0/3     unassigned      Shutdown        Down     default
GigabitEthernet0/0/0/4     unassigned      Shutdown        Down     default
GigabitEthernet0/0/0/5     unassigned      Shutdown        Down     default
GigabitEthernet0/0/0/6     unassigned      Shutdown        Down     default
GigabitEthernet0/0/0/7     unassigned      Shutdown        Down     default
RP/0/0/CPU0:PE-1#
```

Figura 28-3: Tabla de direccionamiento IP en PE1

Fuente: Loza, J. 2021

3.4.2 Configuración de interfaces

Se utiliza la misma URN que en la creación de interfaces sin embargo se utiliza la operación PUT ya que se debe actualizar una configuración ya existente en el router. Para ello se realiza con la solicitud que se indica en el ANEXO C. Para comprobar que la solicitud se ha generado con éxito se puede capturar el tráfico mediante Wireshark como se observa en la figura 29-3.

Figura 29-3: Captura del paquete en wireshark con la solicitud PUT

Fuente: Loza, J. 2021

En este paquete se observa la comunicación entre POSTMAN y ODL en el cual se envía mediante REST sobre el protocolo HTPP un documento XML con la configuración que se desea aplicar en PE1. Entonces ODL envía la configuración que se debe aplicar al router mediante NETCONF sobre TCP y una vez aplicada la configuración, se devuelve un código de estado de confirmación a POSTMAN como se observa en la figura 30-3.

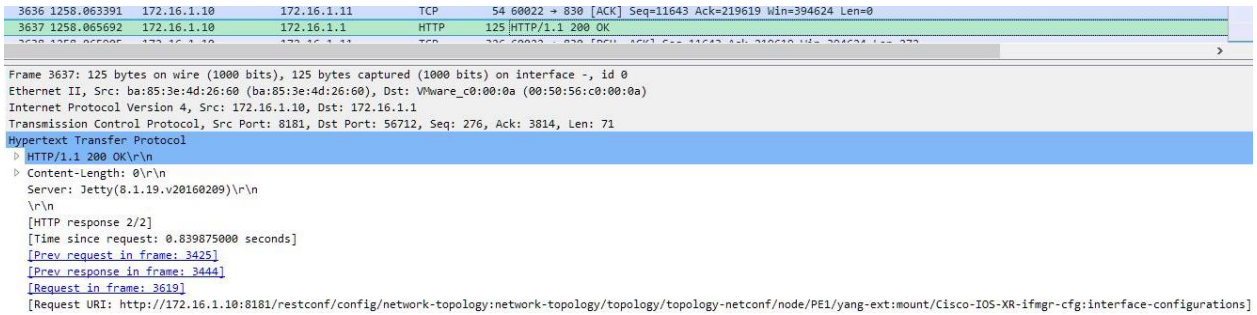


Figura 30-3: Captura del paquete con el código de estado OK

Fuente: Loza, J. 2021

3.4.3 Consulta de una configuración

De igual forma se utiliza el mismo URN que en la creación de interfaces, sin embargo se utiliza la operación GET ya que se van a leer datos de una configuración almacenada en el router (datastore). Para ello se realiza la consulta y en la interfaz de POSTMAN se observan los datos solicitados como se puede ver en la figura 31-3, cabe recalcar que debido a las características de REST esta información puede ser mostrada en diferentes formatos, en este caso se muestra en JSON.

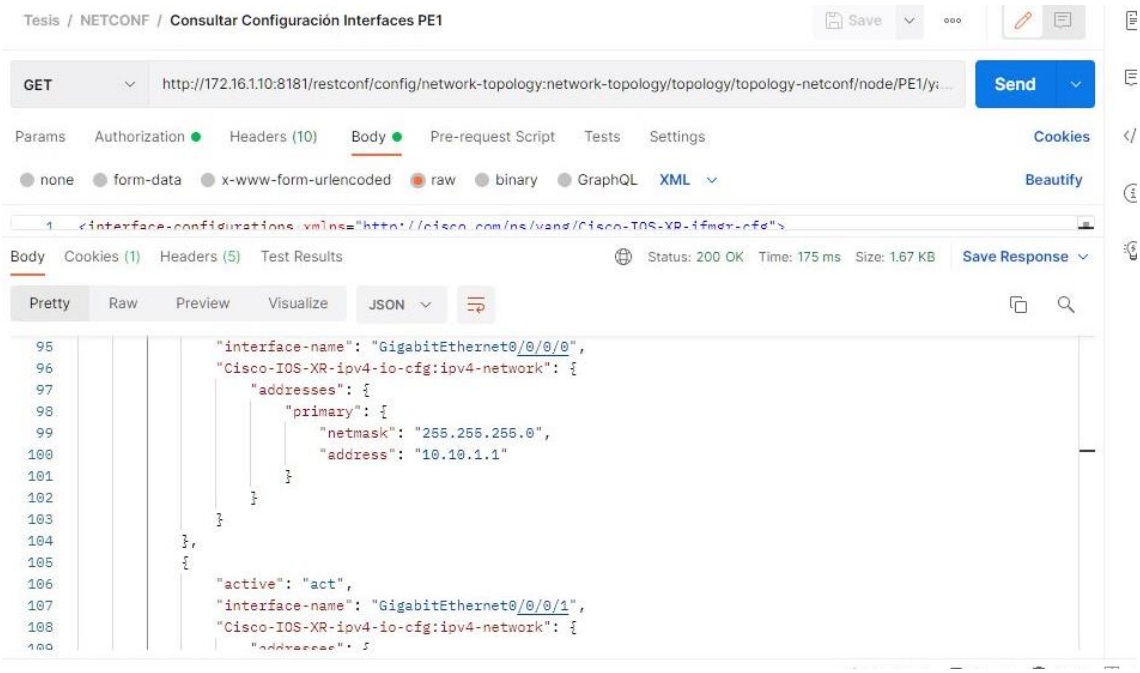


Figura 31-3: Solicitud para la consulta de la configuración de interfaces PE1

Fuente: Loza, J. 2021

CONCLUSIONES

- Se ha diseñado la red de proveedores de servicio para Web, Streaming y VoIP con enrutamiento mediante OSPF y SR, que cuenta con características de diseño para ofrecer una red escalable, con alta disponibilidad, confiable y con facilidades de gestión.
- En base al estudio de los diferentes protocolos de enrutamiento y la compatibilidad con Segment Routing se ha decidido por OSPF ya que IS-IS al ser un protocolo de capa 2, al momento de analizar el tráfico no se muestra los datos necesarios para establecer la comparación deseada.
- Al haber realizado el análisis del rendimiento de una red con OSPF y SR se pudo observar que este último tiene un mejor desempeño desde la primera prueba de estrés frente a OSPF, se pudo observar que una de las grandes ventajas es al momento de transportar tráfico en tiempo real, tanto Streaming como VoIP ya que el Jitter es considerablemente bajo, y gracias al análisis estadístico se pudo comprobar que esta hipótesis es correcta.
- Después de haber realizado el estudio de las arquitecturas de las redes SDN, se ha seleccionado la mejor manera de integrarlo a la red mediante Opendaylight, el controlador SDN que mejor se adapta debido a las herramientas básicas que ofrece y también las avanzadas las cuales no fueron estudiadas en el presente trabajo de titulación, sin embargo, cabe mencionarlas debido a que el desarrollo de las mismas ofrecerá la posibilidad de automatizar la red.
- Una vez se ha integrado el controlador SDN en la red se puedo observar los grandes beneficios que éste tiene ya que brinda la capacidad de integrar una red con servicios RESTful, en este trabajo se ha mostrado la posibilidad de configurar los dispositivos de red por este método, siendo un pilar básico para el objetivo final de una red SDN que es la automatización y se puede complementar con futuros trabajos.

RECOMENDACIONES

- Debido a las características de los programas utilizados para el desarrollo del trabajo, es necesario contar con los recursos de hardware y software necesarios para el correcto desempeño de los mismos, ya que TRex (el cual necesita mínimo 8GB de RAM para funcionar correctamente) no se pudo utilizar con todo su potencial, para realizar pruebas con un tráfico superior a 1GB con el cual se pretendía saber cual es el tráfico máximo que soporta SR una vez ya se conoce que tiene un mejor rendimiento que OSPF. Sin embargo, fue suficiente la capacidad ofrecida por el generador para observar las diferencias de rendimiento entre los dos protocolos de enrutamiento.
- Estudiar las herramientas avanzadas del controlador Opendaylight ya que tiene un potencial muy interesante para automatizar las redes mediante la conexión Northbound a una API, para poder conectarla con diferentes servicios con kubernets, openstack, entre otros. Además de la capacidad de integrar modelos YANG.
- Configurar correctamente los dispositivos de red para el correcto desempeño del protocolo de enrutamiento, comprobar que esté activo y funcionando. Realizar pruebas pequeñas de tráfico para verificar que el archivo de configuración en TRex genere el tráfico deseado. Ya que TRex no tiene una opción para configurar la cantidad de tráfico, es necesario configurar el número de conexiones por segundo que realiza cada servicio, y de esta forma se aumenta la cantidad de tráfico, hasta tener el tráfico deseado.

BIBLIOGRAFÍA

ABUHAYAT. *SR is SDN done right! OpenFlow VS Segment Routing.* [blog]. 2020 [Consulta: 23 julio 2021]. Disponible en: https://abuhayat.me/sr/.

ANDY, B. *Netconf Central NETCONF Documentation.* [en línea]. 2020. [Consulta: 25 Septiembre 2021]. Disponible en: http://www.netconfcentral.org/netconf_docs.

APTIRA. *Segment Routing in Software Defined Networking Wide Area Networks (SDN-WAN).* [en línea]. 2021. [Consulta: 1 julio 2021]. Disponible en: https://aptira.com/segment- routing-in-software-defined-networking-wide-area-networks-sdn-wan/.

MARIJKEK. *Experience with implementing VNF chains with Segment Routing and PCEP.* [en línea]. 2020. [Consulta: 4 julio 2021]. Disponible en: http://dl.ifip.org/db/conf/im/im2021exp/211086.pdf

AVIAT. *SEGMENT ROUTING 101 And the future of MPLS*. [en línea]. 2019. [Consulta: 4 junio 2021]. Disponible en: https://aviatnetworks.com/blog/segment-routing-101-and-the-future- of-mpls/

BARRANCOS MARTÍNEZ, I. *XML para todos*. [en línea]. 2003. pp. 1-26. [Consulta: 2 octubre 2021]. Disponible en: http://www.gnu.org/copyleft/fdl.html.

BASHANDY, A. et al. *Segment Routing with the MPLS Data Plane.* [en línea]. 2019. [Consulta: 4 junio 2021]. Disponible en: https://datatracker.ietf.org/doc/html/rfc8660

BJORKLUND, M. *Yang NETCONF*. [en línea]. 2010. [Consulta: 2 octubre 2021]. Disponible en: https://datatracker.ietf.org/doc/html/rfc6020

ENNS, R., et al. *Network Configuration Protocol (NETCONF).* [en línea]. 2011. [Consulta 1 octubre 2021]. Disponible en: https://datatracker.ietf.org/doc/html/rfc6241

FERNÁNDEZ, R. *Networks para empresas: cuota de mercado de proveedores 2019 | Statista.* [en línea]. 2020 [Consulta: 22 julio 2021]. Disponible en: https://es.statista.com/estadisticas/580535/cuota-de-mercado-mundial-de-los-proveedores- de-redes-informaticas-empresariales/.

FILSFILS, C., et al. *Segment Routing Architecture.* [en línea]. 2018. [Consulta: 6 abril 2021]. Disponible en: https://datatracker.ietf.org/doc/html/rfc8402

FREITAS, H.S. et al. *Deploying SDN experiments in Latin America: the ONOS and SDN-IP application use case at AmLight*. [enlínea]. 2016. [Consulta: 10 octubre 2021]. Disponible en: https://documentas.redclara.net/bitstream/10786/1070/1/Deploying%20SDN%20experiments%20in%20Latin%20America%20the%20ONOS%20and%20SDNIP%20application%20use%20case%20at%20AmLight.pdf

HALEPLIDIS, E. et al. *SDN Layers and Architecture Terminology.* [en línea]. 2015. [Consulta 15 abril 2021]. Disponible en: http://tools.ietf.org/html/draft-haleplidis-sdnrg-layer-terminology-04.

HAYCANAL. *El tráfico global de datos móviles crecerá casi 10 veces entre 2014 y 2019 | Estudios e Informes.* [en línea]. 2021. [Consulta: 7 septiembre 2021]. Disponible en: https://haycanal.com/noticias/7184/el-trafico-global-de-datos-moviles-crecera-casi-10-veces entre-2014-y-2019.

JESUS, E. *Introducción a REST API — Serie SDN №5.* [en línea]. 2021. [Consulta: 10 octubre 2021]. Disponible en: https://jesuseduardoespinoza.medium.com/introducción-a-rest-api- serie-sdn-5-c0ae9142ff1.

JIMÉNEZ, D.C. *SIMULACIÓN DE UNA RED DE VIDEOVIGILANCIA IP BASADA EN GSN3*. [en línea] (Trabajo de titulación). Universitat Politecnica de Valencia, Valencia, España. 2020. [Consulta: 16 marzo 2021]. Disponible en: https://riunet.upv.es/bitstream/handle/10251/152379/Casado%20%20Simulaci%c3%b3n%20de

%20una%20red%20SDN%20de%20videovigilancia%20IP%20basada%20en%20GNS3.pdf?sequence=1&isAllowed=y

LUCAS, S. *RESTCONF: Una guía práctica de uso - Algo de redes.* [en línea]. 2020. [Consulta: 13 octubre 2021]. Disponible en: https://algoderedes.com/restconf-guia-practica/.

MARBAN. *Red SDN HUAWEI Conceptos y Valores Proyecto ODL - Comunidad Huawei Enterprise.* [en línea]. 2020a. [Consulta: 20 octubre 2021]. Disponible en: https://forum.huawei.com/enterprise/es/red-sdn-huawei-conceptos-y-valores-proyecto-odl/thread/616448-100235.

MARBAN. *Red SDN HUAWEI Protocolos y Principios de la Interfaz Servicio RESTful-Comunidad Huawei Enterprise.* [en línea]. 2020b. [Consulta: 20 octubre 2021]. Disponibleen: https://forum.huawei.com/enterprise/es/red-sdn-huawei-protocolos-y-principios-de-la- interfaz-servicio-restful/thread/617762-100235.

MERELO HERNÁNDEZ, M. *Estudio de la gestión de configuración a través de NETCONF.* [en línea]. (Trabajo de titulación). Universidad de Sevilla, Sevilla, España. 2017. [Consulta: 7 octubre 2021]. Disponible en: http://bibing.us.es/proyectos/abreproy/91200/fichero/Estudio+de+la+gestion+de+configuracion+a+traves+de+NETCONF.pdf

MOY, J. *OSPF V2.* [en línea]. 1998. [Consulta: 10 julio 2021]. Disponible en: https://datatracker.ietf.org/doc/html/rfc2328

OPEN NETWORKING FOUNDATION. *Software-Defined Networking: The New Norm for Networks.* [en línea]. 2012. [Consulta: 5 junio 2021]. Disponible en: http://opennetworking.wpengine.com/wp-content/uploads/2011/09/wp-sdn-newnorm.pdf

OPENDAYLIGHT. *Platform Overview - OpenDaylight.* [en línea]. 2021. [Consulta: 3 octubre 2021]. Disponible en: https://www.opendaylight.org/about/platform-overview.

PAUL, D. *Introducción a Segment Routing.* [en línea]. 2018 [Consulta: 20 junio 2021]. Disponible en: https://learningnetwork.cisco.com/s/blogs/a0D3i000002SKD4EAO/introducción-a-segment-routing.

PANG, J., et al. *SDN-Based Data Center Networking with Collaboration of Multipath TCP and Segment Routing.* [en línea]. 2017. [Consulta: 18 octubre 2021]. Disponible en: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7918573

PARASCHIS, L. *SDN Innovations and Control Plane Evolution in the new Wireline Transport Architectures*. [en línea]. 2019. [Comsulta en: 17 abril 2021]. Disponible en: https://www.ciscolive.com/c/dam/r/ciscolive/emea/docs/2016/pdf/BRKOPT-2102.pdf

PARRA, G.C.B. *Diseño de un servicio L3VPN en GNS3 con tecnología de enrutamiento de segmentos en un enfoque distribuido para la comunicación de dos clientes finales.* [en línea]. (Trabajo de titulación). (Maestría). Universidad Católica de Santiago de Guayaquil, Guayaquil, Ecuador. 2020. [Consulta: 4 marzo 2021]. Disponible en: http://repositorio.ucsg.edu.ec/handle/3317/3273.

PERRIN, S. *Making Networks SDN-Ready With Segment Routing.* [en línea]. 2017. [Consulta: 10 marzo 2021]. Disponible en: https://www.segmentrouting.net/images/lightreading_report.pdf

PSENAK, P., et al. *Extensions for Segment Routing.* [en línea]. 2019. [Consulta: 10 junio2021]. Disponible en: https://tools.ietf.org/html/rfc8667.

QUIMBAYO RODRÍGUEZ, C.J. *PROPUESTA METODOLÓGICA PARA LA SELECCIÓNDE CONTROLADORES DE REDES SDN A NIVEL EMPRESARIAL*. [en línea]. (Trabajo de titulación). (Maestría). Universidad Santo Tomas, Bogotá, Colombia. 2020.[Consulta: 29 julio 2021]

RESTFULAPI. *What is REST.* [en línea]. 2021 [Consulta: 17 octubre 2021]. Disponible en: https://restfulapi.net/.

SANTOS, V. *Simplifique su red IP con Segment Routing, una parte de Adaptive IP de Ciena - Ciena.* [en línea]. 2019. [Consulta: 20 junio 2021]. Disponible en: https://www.ciena.com.mx/insights/articles/Simplify-your-IP-network-with-centralized-Segment-Routing-part-of-Cienas-Adaptive-IP_es_LA.html.

TANENBAUM, A.S., & WETHERALL, D.J. *Redes de computadoras*. [en línea]. 2012. [Consulta: 2 mayo 2021]. Disponible en: https://bibliotecavirtualapure.files.wordpress.com/2015/06/redes_de_computadoras- freelibros-org.pdf

TELEFÓNICA. *Telefónica registra durante la crisis del Covid-19 un crecimiento en su tráfico de internet equivalente al de todo el año pasado*. [en línea]. 2020. [Consulta: 7 septiembre2021]. Disponible en: https://www.telefonica.com/es/web/sala-de-prensa/-/telefonica- registra-durante-la-crisis-del-covid-19-un-crecimiento-en-su-trafico-de-internet- equivalente-al-de-todo-el-ano-pasado.

TREX. *TRex*. [en línea]. 2021 [Consulta: 10 agosto 2021]. Disponible en: https://trex-tgn.cisco.com/trex/doc/trex_manual.html.

TREX. *TRex Realistic Traffic Generator*. [en línea]. 2021 [Consulta: 20 julio 2021]. Disponible en: https://trex-tgn.cisco.com/.

ANEXOS

Anexo A: Especificaciones del router Cisco IOS XR

Specifications

Tables 1 through 7 list primary specifications for the Cisco NCS 540 fixed chassis.

Table 1. Cisco NCS 540 chassis specification

Feature	Specification
Chassis PIDs	N540-24Z8Q2C-SYS N540X-24Z8Q2C-SYS (conformal coated)
Port configuration	24 ports of 1GE/10GE 8 ports of 1GE/10GE/25GE 2 ports of 40GE/100GE
Integrated route processor	Cisco IOS XR 64-bit software running on a 4-core 1.5Ghz x86 CPU
Management ports	2x RJ45, 1 for console and 1 for management LAN port
Flexible forwarding ports	24 x 10GE ports with SFP+ or SFP optics for 10GE or 1GE options 8 x 25GE with SFP28 or SFP+ optics for 25GE, 10GE or 1GE options 2 x 100GE with QSFP28 optics (supports 4 x 25GE breakout), or as 2 x 40GE with QSFP+ optics (supports 4 x 10GE breakout)
Performance	300 Gbps and 300 Mpps of system throughput
Route scale	Up to 128K IPv4 and 32K IPv6 FIB entries
Timing	Internal GNSS module with GPS, GLONASS, Galileo, Beidou Interfaces: 1 PPS in/out, 10Mhz in/out, ToD in/out, antenna for internal GNSS

Feature	Specification
Power and cooling	2 hot-swappable power supplies provide 1+1 redundancy 4 fans provide 3+1 redundant system cooling Front-to-back airflow
Power consumption	Typical: 200W at 25°C Maximum: 240W at 40/45°C, 270W at 50/55/70°C
Physical specification	Height: 1RU 1.72 in. (4.37cm) Width: 17.3 in. (43.94cm) Depth: 10.02 in. (25.45cm) Weight: 14 lb. (6.35kg)

Description	Specification
Quality of Service (QoS)	Hierarchical QoS Ingress classification based on class of service (L2) IP differentiated service code point (L3) IP precedence (type of service) (L3) Policing, Shaping 4096 number of queues for user traffic Support for priority queuing 3GB packet buffer
Timing	T-GM, T-BC, T-TSC Integrated GNSS module with GPS, GLONASS, Galileo, Beidou SyncE, G.8265.1, G.8275.1, G.8275.2 G.8273.2 Class B
Automation	Zero-Touch Provisioning (ZTP) with iPXE Configuration management Network Configuration Protocol (NETCONF/YANG model)
Security	Control-plane and management plane protection Authentication, Authorization, and Accounting (AAA) Terminal Access Controller Access-Control System Plus (TACACS+) Secure Shell (SSH) Protocol Layer 2 ingress ACLs Layer 3 ingress ACLs
Management	MIB, XML, JSON, GPB, and SNMP MPLS OAM Ethernet OAM

Table 2. Software feature support on NCS 540 in Cisco IOS XR 6.3.2 release or beyond

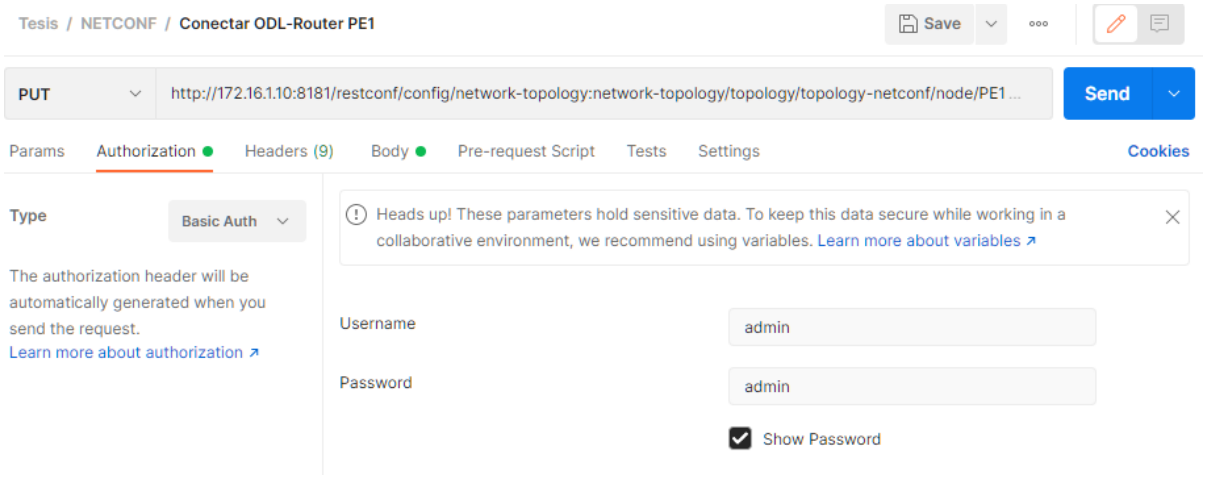
Description	Specification
Layer 2	Layer 2 switch ports IEEE 802.1Q VLAN encapsulation / Q-in-Q encapsulation IEEE 802.1ad Ethernet Link Aggregation Group (LAG) (up to 32 ports) Link Aggregation Control Protocol (LACP): IEEE 802.3ad Jumbo frames on all ports (up to 9216 bytes) L2 ingress Access Control List (ACL) Ethernet Flow Point (EFP) and VLAN trunks L2 Bridge Domains
Layer 3	IPv4 and IPv6 unicast Layer 3 interfaces: physical interfaces and subinterfaces Routing protocols: static, Open Shortest Path First (OSPFv2), OSPFv3, Intermediate System to Intermediate System (ISIS), ISISv6, and Border Gateway Protocol (BGP) 32-way equal-cost multipath (ECMP) L3 ingress and egress IPv4 ACL and IPv6 ACL Bidirectional Forwarding Detection (BFD) Cisco bundle Ethernet technology (up to 32 ports per Ethernet bundle) Jumbo frame support (up to 9216 bytes) Virtual Router Redundancy Protocol (VRRP)Integrated Routing Bridging (IRB) with Bridge Virtual Interface (BVI)
MPLS	Label switching LDP SR-MPLS RSVP-TE and SR-TE MPLS Traffic Engineering Point-to-point L2VPN – T-LDP, BGP, EVPN-VPWS Multipoint L2VPN – VPLS, EVPN L2/L3 VPN EVPN with Anycast IRB
Segment Routing (SR)	Segment Routing with MPLS data plane ISIS extensions to segment routing OSPF extensions to segment routing BGP extensions to segment routing BGP egress peering engineering (BGP-EPE) Segment Routing Traffic Engineering (SR-TE) Segment routing Topology Independent Loop-Free Alternatives (TI-LFA) On-Demand Next-hop (ODN)
Multicast	IPv4, IPv6 PIM-SM, PIM-SSM IGMPv3, MLDv2 PIM-ECMP mLDP P2MP-TE

Anexo B: Configuración del router

```
CONFIGURACIÓN PE-1
conf t
hostname PE-1
commit
int lo0
ip add 1.1.1.1/32
exit
int Gi0/0/0/0
ip add 10.10.1.1/24
no shutdown
exit
int Gi0/0/0/1
ip add 10.10.2.1/24
no shutdown
exit
int Gi0/0/0/2
ip add 192.168.10.2/24
no shutdown
exit
commit
router ospf 10
router-id 1.1.1.1
area 0
int lo0
int Gi0/0/0/0
network point-to-point
int Gi0/0/0/1
network point-to-point
int Gi0/0/0/2
commit
end
configure
router ospf 10
auto-cost reference-bandwidth 10000
commit
end
```

```
configure
router ospf 10
segment-routing mpls
segment-routing forwarding mpls
segment-routing sr-prefer
area 0
interface loopback0
prefix-sid index 1
commit
end
//////////
int g0/0/0/0
fast-reroute per-prefix
fast-reroute per-prefix ti-lfa
exit
int g0/0/0/1
fast-reroute per-prefix
fast-reroute per-prefix ti-lfa
exit
////////////
interface MgmtEth0/0/CPU0/0
ipv4 address 172.16.1.11 255.255.255.0
no shut
commit
exit
ssh server v2
ssh server netconf port 830
ssh server netconf vrf default
netconf-yang agent
ssh
netconf-yang agent session absolute-timeout 180
commit
```

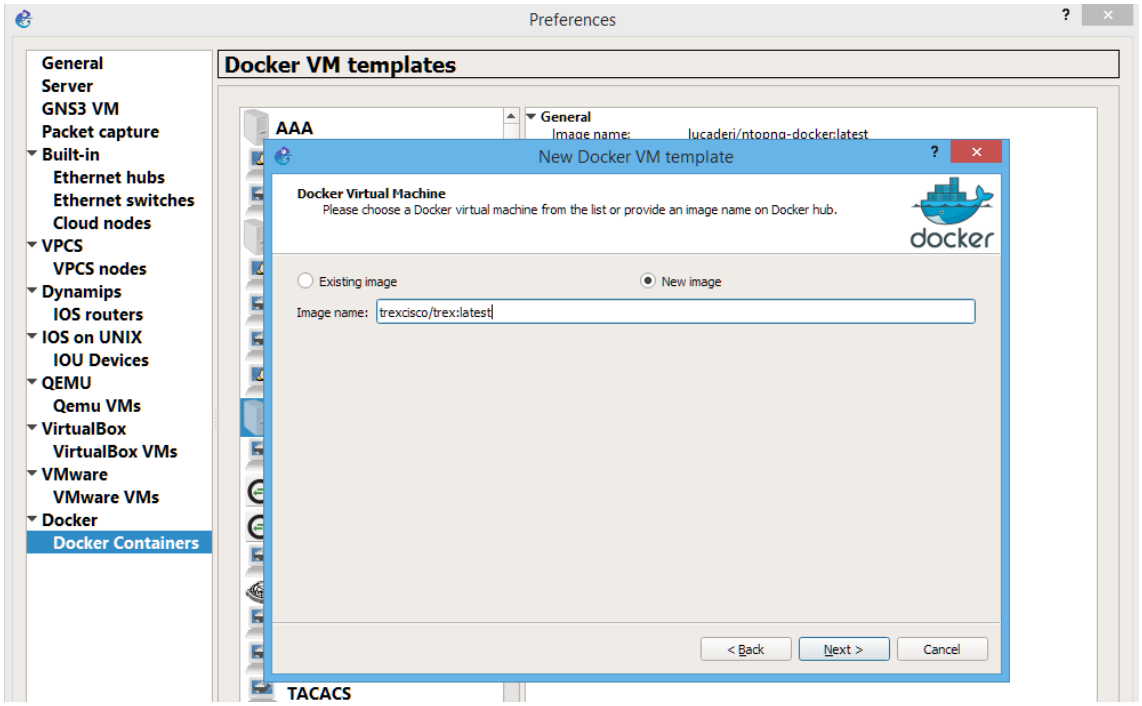
Anexo C: Archivos de configuración en POSTMAN

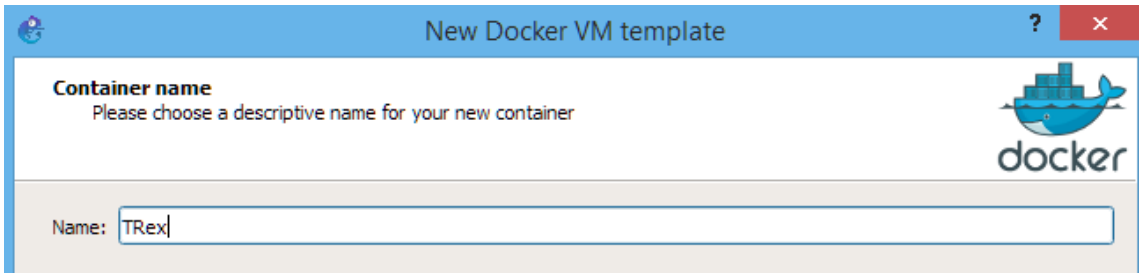


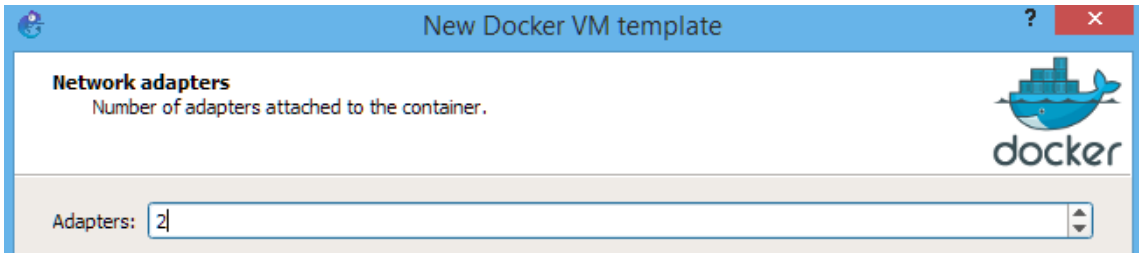
DOCUMENTO XML PARA CONFIGURAR PE1

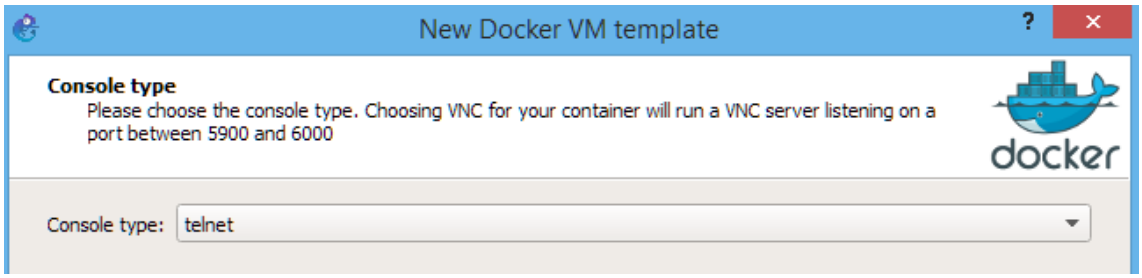
```
<interface-configurations xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-
ifmgr-cfg">
<interface-configuration>
<active>act</active>
<interface-name>MgmtEth0/0/CPU0/0</interface-name>
<ipv4-network xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ipv4-io-cfg">
<addresses>
<primary>
<netmask>255.255.255.0</netmask>
<address>172.16.1.11</address>
</primary>
</addresses>
</ipv4-network>
</interface-configuration>
<interface-configuration>
<active>act</active>
<interface-name>GigabitEthernet0/0/0/0</interface-name>
<ipv4-network xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ipv4-io-cfg">
<addresses>
<primary>
<netmask>255.255.255.0</netmask>
<address>10.10.1.1</address>
</primary>
</addresses>
</ipv4-network>
</interface-configuration>
<interface-configuration>
<active>act</active>
<interface-name>GigabitEthernet0/0/0/1</interface-name>
<ipv4-network xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ipv4-io-cfg">
<addresses>
<primary>
<netmask>255.255.255.0</netmask>
<address>10.10.2.1</address>
</primary>
</addresses>
</ipv4-network>
</interface-configuration>
<interface-configuration>
<active>act</active>
<interface-name>GigabitEthernet0/0/0/2</interface-name>
<ipv4-network xmlns="http://cisco.com/ns/yang/Cisco-IOS-XR-ipv4-io-cfg">
<addresses>
<primary>
<netmask>255.255.255.0</netmask>
<address>192.168.10.2</address>
</primary>
</addresses>
</ipv4-network>
</interface-configuration>
</interface-configurations>
```

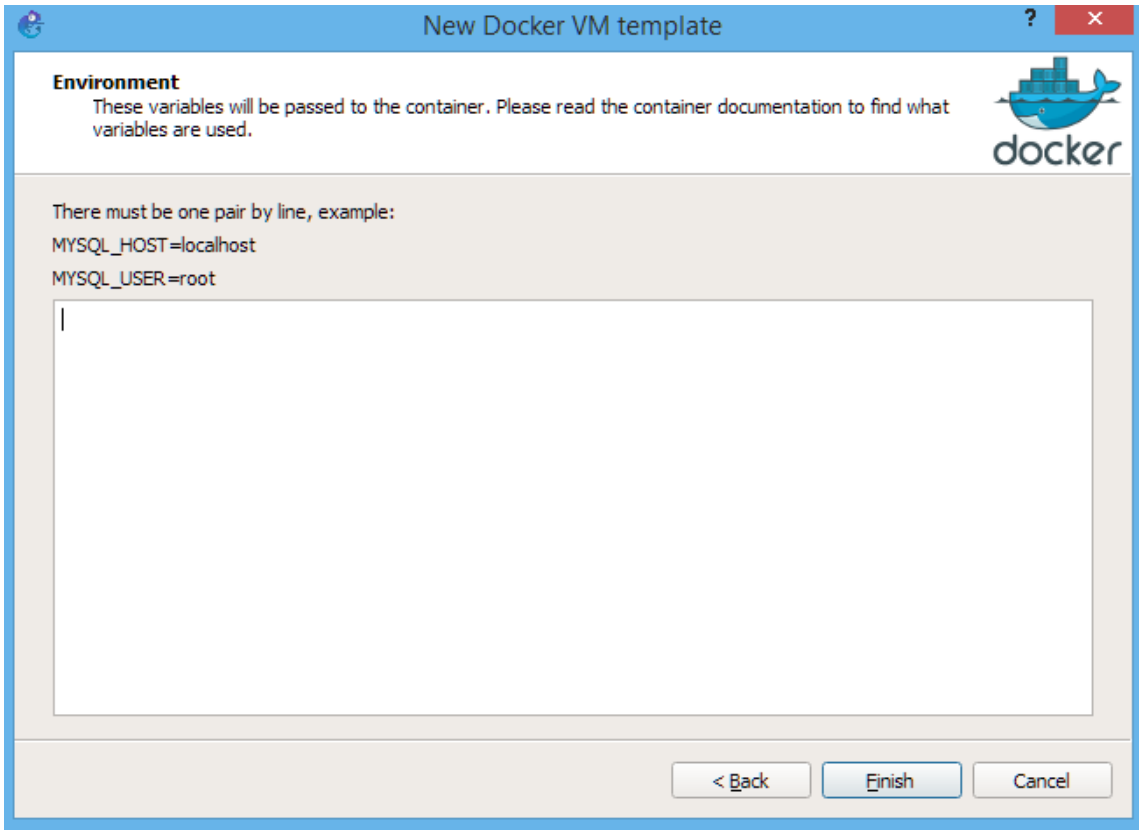
Anexo D: Instalación de TRex y Opendaylight

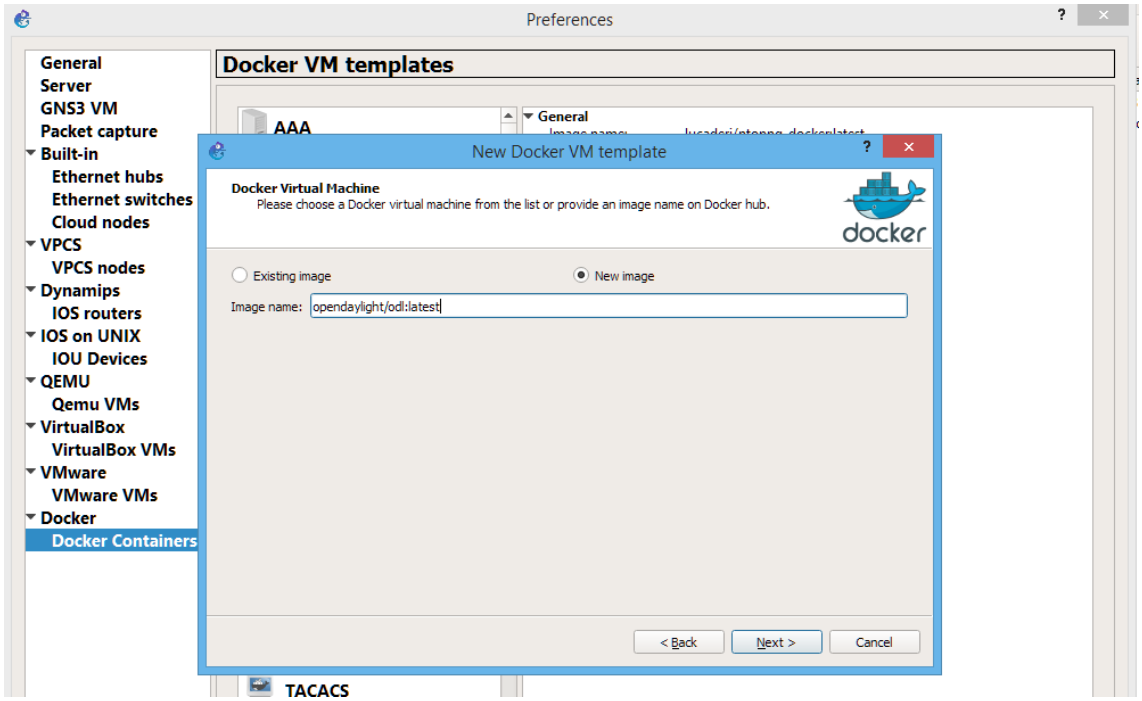









New Docker VM template
Environment
These variables will be passed to the container. Please read the container documentation to find what variables are used.
docker
There must be one pair by line, example:
MYSQL_HOST=localhost
MYSQL_USER=root
< Back
Finish
Cancel


Preferences
General
Server
GNS3 VM
Packet capture
Built-in
Ethernet hubs
Ethernet switches
Cloud nodes
VPCS
VPCS nodes
Dynamips
IOS routers
IOS on UNIX
IOU Devices
QEMU
Qemu VMs
VirtualBox
VirtualBox VMs
VMware
VMware VMs
Docker
Docker Containers
Docker VM templates
AAA
General
TACACS
New Docker VM template
Docker Virtual Machine
Please choose a Docker virtual machine from the list or provide an image name on Docker hub.
docker
Existing image
New image
Image name:
opendaylight/odl:latest
< Back
Next >
Cancel

espoch | Dirección de Bibliotecas y Recursos del Aprendizaje

UNIDAD DE PROCESOS TÉCNICOS Y ANÁLISIS BIBLIOGRÁFICO Y DOCUMENTAL

REVISIÓN DE NORMAS TÉCNICAS, RESUMEN Y BIBLIOGRAFÍA

Fecha de entrega: 11 / 03 / 2022

INFORMACIÓN DEL AUTOR/A (S)
Nombres – Apellidos: JHON JAVIER LOZA LLUCO
INFORMACIÓN INSTITUCIONAL
Facultad: INFORMÁTICA Y ELECTRÓNICA
Carrera: TELECOMUNICACIONES
Título a optar: INGENIERO EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES
f. Analista de Biblioteca responsable: Lcdo. Holger Ramos, MSc.

UNIDAD DE PROCESOS TÉCNICOS DBRA DIRECCIÓN DE BIBLIOTECAS Y RECURSOS DEL APRENDIZAJE ESPOCH

Firmado electrónicamente por:
HOLGER GERMAN RAMOS UVIDIA

0353-DBRA-UPT-2022