TELEMÁTICA
REVISTA DIGITAL de las Tecnologías de la Información y las Telecomunicaciones
Revista Telemática. Vol. 18. No. 1, enero-abril, 2019, p.1- 10 ISSN 1729-3804

COMBINACIÓN DE MECANISMOS MPLS EN UNA ARQUITECTURA SDN

Ing. Daniel Iglesias de la Torre[1], Dr. C. Ing. Félix Álvarez Paliza[2], MSc. Arelis Ramos Fleites[3]

[1]Universidad Central Marta Abreu de Las Villas, Carretera a Camajuaní Km 5 ½, [2]Universidad Central Marta Abreu de Las Villas, Carretera a Camajuaní Km 5 ½, [3]Universidad Central Marta Abreu de Las Villas, Carretera a Camajuaní Km 5 ½
[1]e-mail: danielit@uclv.cu, [2]e-mail: fapaliza@uclv.edu.cu, [3]e-mail: arelis@uclv.edu.cu

RESUMEN

Las redes de telecomunicaciones han evolucionado rápidamente en los últimos cinco años con la aparición de lo que a menudo se denomina redes definidas por software. El movimiento de las redes definidas por software se basó en la invención de OpenFlow, un medio revolucionario de implementar dispositivos de red donde el software del dispositivo se reemplaza por el uso de un controlador centralizado en la red. OpenFlow fue parte de un movimiento más amplio para hacer las redes más abiertas y, al hacerlo, aumentar la competencia y permitir una innovación más amplia. Recientemente, los conceptos de las redes definidas por software han comenzado a transformar la red de área amplia de la empresa. En diversos sentidos, la transformación definida por software ha sido la cuestión más emocionante hasta la fecha, debido a los beneficios claros e inmediatos para el cliente. La necesidad de sistemas de redes de área extensa definidas por software debe entenderse en el contexto de cómo las tecnologías de la información están cambiando y de cómo se construyen las redes. Las capacidades de las redes definidas por software ayudan a los proveedores de servicios a agregar valor mejorando la infraestructura de conmutación de etiquetas multiprotocolo y aumentar el potencial de ingresos ofreciendo servicios adicionales. En este artículo se hace una breve descripción de las principales características de las redes definidas por software y su aplicación en redes de área extensa. Además se describe el protocolo Openflow y se selecciona el controlador Opendaylight. Como parte de la implementación se utiliza el protocolo de gestión de base de datos de conmutador virtual abierto, seleccionado para gestionar el plano de control y mecanismos de control con conmutación de etiquetas multiprotocolo para transmitir flujos en el plano de datos.

PALABRAS CLAVES: Redes de área extensa definidas por software, Openflow.

COMBINATION OF MPLS MECHANISM IN AN SDN ARCHITECTURE

ABSTRACT

Networking has rapidly evolved in the last five years with the emergence of what is often called software-defined networking. The software-defined networking subject was based on the invention of OpenFlow, a revolutionary means of implementing network devices where the software of the device is replaced by the use of a centralized controller in the network. OpenFlow was part of a broader movement to make networking more open and, by doing so, increase competition and enable broader innovation. Recently, software-defined networking concepts have begun to transform the enterprise wide-area network. In several ways, software-defined represents the most exciting transformation provided the clear and immediate benefits to the customer.

The needs for software-defined wide-area network systems must be understood in the context of how information technology is changing as well as how networks are built. Software-defined networking capabilities help service providers to add value by enhancing multiprotocol label switching infrastructure; increase revenue potential by offering additional services on the software-defined wide-area network; and, perhaps most important, being responsible for the intelligence in the controller. In this article, a brief description of main characteristics of networks defined by software and its application in wide area networks is made. In addition, OpenFlow protocol is described and Opendaylight controller is selected. As part of the implementation, the selected open virtual switch database management protocol is used to manage the control plane and control mechanisms with multiprotocol label switching to transmit flows in the data plane.

KEY WORDS: Software-defined wide-area network, OpenFlow.

1. INTRODUCCIÓN

En los últimos años la tecnología ha cambiado considerablemente con el surgimiento y consolidación de nuevas tendencias, como la aparición de una gran cantidad de dispositivos que se conectan a la red, grandes volúmenes de datos, nuevas aplicaciones y servicios. Debido a esto, las redes se ven obligadas a soportar unas exigencias para las que no fueron diseñadas originalmente, lo que hace que tengan la necesidad de modificar su enfoque [1].
Los proveedores de Internet (ISPs), debido a su interés por controlar los flujos de datos que atraviesan las redes para poder brindar servicios de calidad y en tiempo real a sus usuarios, han tratado de lidiar con este masivo crecimiento y expansión poniendo en práctica diferentes técnicas que se han ido desarrollando a través de las últimas décadas. De esta manera se crea la arquitectura de Redes Definidas por Software (SDN), que se presentan como un nuevo paradigma para cubrir las nuevas necesidades y promete transformar las arquitecturas y la gestión de las redes que se conocen en la actualidad. Se proponen nuevas arquitecturas de red que sean flexibles, escalables, programables y capaces de soportar múltiples servicios; que logren optimizar la forma en la que operamos, comprendemos, diseñamos e interactuamos con las redes [2].

La Open Networking Foundation (ONF) ha liderado el desarrollo de la arquitectura SDN y la normalización de sus elementos, entre ellos el protocolo OpenFlow. El cual fue desarrollado para mediar entre el software de control y los elementos de la red. Esto permite el acceso directo a la gestión de datos de reenvío en dispositivos de red como conmutadores y enrutadores, tanto físicos como virtuales y de un modo abierto [3].

Recientemente, los conceptos de SDN han comenzado a transformar las redes de área extendida empresariales. Los sistemas SD-WAN (Wide Area Networks) hacen al cliente menos dependiente de un proveedor MPLS (Multiprotocol Label Switching), haciendo mucho más fácil integrar tecnologías nuevas o posiciones nuevas en la Red de Área Extendida. Pero inversamente, estas tecnologías permiten al proveedor de servicios aumentar y extender las implementaciones MPLS existentes. Además, posibilita responder a las nuevas necesidades de los clientes rápidamente dentro del sistema MPLS correcto [1]. En este trabajo se utiliza el plano de datos MPLS conjuntamente con el plano de control basado en OpenFlow. El objetivo general es configurar flujos de datos entre dispositivos finales con etiquetado MPLS en una arquitectura SDN.

1. **MPLS**

MPLS es un mecanismo eficiente de reenvío de paquetes que brinda potencialidades en la capa de enlace de datos y la capa de red. Transporta datos de nodos presentes en una red a otra utilizando etiquetas en lugar de direcciones basadas en IP. MPLS usa etiquetas para identificar la ruta de los paquetes que deben atravesar la red. Los componentes principales de una arquitectura MPLS son el enrutador de conmutación de etiquetas (LSR, Label Switching Router), enrutador de borde (LER, Label Edge Router) y la trayectoria de conmutación de etiquetas (LSP, Label Switched Path). El LSR actúa como un conmutador de tránsito en la red troncal MPLS. Los paquetes IP etiquetados se reciben a través de los LSP y finalmente son enviados a la salida por el LER [4].

2. **SDN**

Las SDN se definen como una arquitectura de red dinámica, gestionable, adaptable, de costo eficiente. Lo cual la hace ideal para las altas demandas de ancho de banda y la naturaleza dinámica de las aplicaciones actuales. Esta arquitectura separa el plano de control (encargado de la inteligencia de la red, selección de protocolo, balanceo de carga, correspondería al software) y el plano de datos (responsable del reenvió de paquetes, es decir, se encarga de envío y recepción de estos a través de la red, que se correspondería con el hardware). Esta separación permitie que el control de la red pueda ser completamente programable logrando que las aplicaciones y servicios de red se abstraigan de la infraestructura de red subyacente [5].
El administrador de red puede moldear el intercambio de información desde una consola de control centralizada sin afectar conmutadores individuales. Además, puede cambiar las reglas de los conmutadores de la red siempre y cuando sea necesario agregando o eliminando prioridades, bloqueando o permitiendo el tráfico de paquetes específicos con un control detallado algo que no ocurre en un entorno de red tradicional [6, 7].
SDN es una arquitectura muy joven, en la cual se ha pasado de una estructura totalmente cerrada y limitada a una estructura abierta, donde el Plano de Control (la inteligencia de los dispositivos de red) y el Plano de Datos (responsable del envío de los paquetes) se encuentran desacoplados. De esta forma se separan las funciones de gestión de red y de conmutación de paquetes. En una arquitectura cerrada la escalabilidad es limitada y las prestaciones de servicios dependen de cada casa fabricante. Por ejemplo, para poder usar nuevos protocolos de enrutamiento, VPNs, control de acceso, administración, o cualquier otro requerimiento ha sido necesario esperar la actualización de los sistemas y esto conlleva a ser totalmente dependientes del fabricante. En una red convencional cuando un paquete llega a un conmutador, este envía cada paquete al mismo destino por la misma trayectoria y trata a todos los paquetes de la misma manera; mientras que en las SDN es posible tener acceso a las tablas de flujo del conmutador, con la finalidad de modificar, eliminar o añadir nuevas reglas de flujo, lo cual permite controlar el tráfico de la red de manera personalizada.

SD-WAN

SD-WAN es un ecosistema de hardware (incluidos los equipos de las instalaciones del cliente, como dispositivos de borde), software (incluidos los controladores) y servicios que permiten el rendimiento, confiabilidad y seguridad WAN de nivel empresarial en una variedad de formas. En primer lugar, permite el enrutamiento dinámico y eficiente del

tráfico basado en políticas a través de múltiples redes WAN. Los proveedores de servicios pueden usar MPLS, Internet público, LTE, banda ancha u otros métodos para optimizar el uso del ancho de banda al menor costo posible. En segundo lugar, el elemento "definido por software" da servicios a los proveedores y sus clientes con una mayor flexibilidad cuando se trata de automatizar procesos claves. Esto acelera y simplifica el proceso de configuración, adición, eliminación y administración de servicios de red basados en negocios [8, 9]. En la figura 1 se muestra la arquitectura SD-WAN.

Las soluciones SD-WAN ofrecen beneficios potenciales, entre los que se encuentran:

- Mejor uso de las tecnologías de internet al mejorar el ancho de banda y el costo de rendimiento.
- Mejor uso de los recursos para el acceso a internet.
- Conectividad optimizada.
- Ahorro de costos y mejoras a los servicios MPLS.

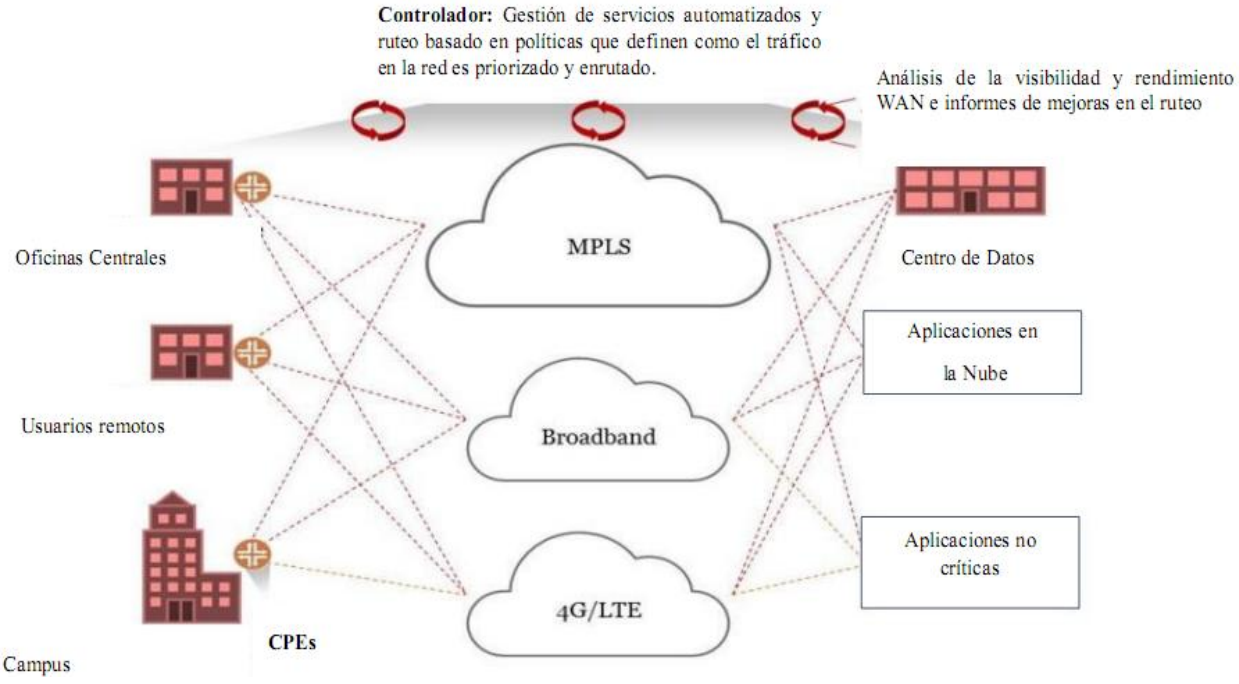

Figura 1: Arquitectura SD-WAN [10].

Openflow

Openflow es el protocolo que posibilita la implementación de SDN tanto en software como en hadware [11-13]. El protocolo OpenFlow estandariza los mensajes intercambiados entre el controlador y el conmutador. En general, estos mensajes contienen instrucciones de cómo el conmutador debe manejar tipos específicos de paquetes y recopilar estadísticas de estos flujos. Un conmutador OpenFlow tiene una o múltiples tablas de flujo para determinar cómo reenviar los paquetes. Estas tablas funcionan como tablas de búsqueda, de las cuales el conmutador aprende qué hacer con los paquetes. Cuando llega un nuevo flujo, el conmutador chequea su tabla de flujo [14].

La entrada de flujo se utiliza para identificar y procesar paquetes. Contiene un conjunto de campos de coincidencia, prioridad, contadores, tiempo de espera y un conjunto de instrucciones. El campo de coincidencia es un campo contra el cual el paquete está emparejado. A partir de la versión OpenFlow 1.3, se tienen 40 campos de coincidencia. Estos varían desde la direcciones fuente y destino hasta la etiqueta MPLS [15].
Los conmutadores OpenFlow pueden estar basados en hardware o en software. Conmutadores basados en software tales como Open vSwitch y ofsoftswitch se pueden instalar en una computadora de uso general, agregando y cambiando las funcionalidades de este [15].

OVSDB

Aunque OpenFlow es el protocolo que permite la comunicación entre el controlador y el conmutador. La gestión del conmutador en sí no es realizada por OpenFlow. Para ese propósito, se pueden usar dos protocolos diferentes, OVSDB y OF-Config, este último es un protocolo de gestión desarrollado por la ONF y se supone que funciona con cualquier dispositivo Openflow, Mientras que el protocolo OVSDB está específicamente desarrollado para Open vSwitch. OVSDB funciona con cualquier implementación de software y hardware de OVS [16].
OVSDB gestiona las operaciones de conmutación, como la creación de interfaces, la configuración de políticas de QoS. Una instancia OVS consiste en un servidor de base de datos (ovsdb-server) y un daemon de conmutador virtual (ovs-vswitchd). Los clústeres de gestión y control son los administradores OVSDB y el controlador OpenFlow, que pueden estar ubicados en el mismo o en diferentes dispositivos [15].

Controlador Opendaylight

OpenDaylight (ODL) se presenta como un proyecto cuya intención es ir más allá de las implementaciones SDN con Openflow, con una plataforma modular y flexible que actúa como controlador. La arquitectura del controlador ODL se caracteriza por ser ampliable, escalable y multiprotocolo o microservicios, o sea, compuesta por un conjunto de servicios o protocolos particulares, independientes entre sí, cada uno de los cuales se centra en un aspecto particular, que se comunican entre sí para realizar aplicaciones más complejas.
Proporciona una abstracción de servicios impulsada por modelos que permite a los usuarios desarrollar aplicaciones que funcionan fácilmente a través de una amplia variedad de hardware y protocolos. Además, contiene complementos internos que agregan servicios y funcionalidades a la red. Por ejemplo, tiene complementos dinámicos que permiten recopilar estadísticas y obtener la topología de la red [17].

Para la administración, SDN y OpenDaylight abren la puerta a capacidades sofisticadas que de otro modo no serían factibles. La visibilidad global y de todo el dominio aumenta la capacidad de administrar una supervisión efectiva de amenazas en toda la red. La capacidad de aplicar políticas basadas en aplicaciones, servicios y grupos (a diferencia de los recursos físicos) permite una administración más relevante y más detallada. Los flujos están particularmente bien adaptados a las aplicaciones de seguridad que buscan proteger las comunicaciones de extremo a extremo independientemente de las limitaciones de la red. Finalmente, el control programático a través de una API abierta basada en la intención

permite el control dinámico de la política y la capacidad de modificar el comportamiento de la red para mejorar la seguridad y el rendimiento [18].

CONFIGURACIÓN DE FLUJOS MPLS EN UNA ARQUITECTURA SDN

El plano de control del diseño consta del controlador SDN Opendaylight para implementar mecanismos de control de MPLS. Este usa el protocolo OpenFlow para comunicar e instalar entradas de flujos de datos, etiquetado MPLS y descubrimiento de topología en los conmutadores SDN. Mientras que el plano de datos se implementó a través del emulador de redes virtuales Mininet.
Tan pronto como la red se inicializa, el controlador inicia los servicios como monitoreo, descubrimiento de topología, y asignación de recursos. El controlador obtiene las políticas que son implementadas por el administrador de red y descubre los caminos diseñados. Con el controlador centralizado se tiene una vista global de la topología y una mejor programabilidad. Esto optimiza las formas de descubrimiento del camino MPLS y genera las etiquetas y las distribuye a los conmutadores SDN usando los protocolos apropiados.

El plano de datos consiste en un conjunto de conmutadores SDN virtuales. Los conmutadores SDN que están en contacto directo con el controlador, actúan como enrutadores de borde (LER), asignando la etiqueta MPLS a los paquetes entrantes. Los paquetes son reenviados a través de la trayectoria de conmutación de etiquetas (LSP), donde cada enrutador de conmutación de etiquetas (LSR) toma decisiones de enrutamiento de acuerdo a la etiqueta de cada paquete. De esta forma se fortalecen las decisiones del controlador. En cada salto, los enrutadores MPLS extraen la etiqueta existente y la reemplazan por una nueva para realizar el siguiente salto en el encaminamiento del paquete.

Implementación

En la herramienta Mininet se emularon todos los elementos de la red como hosts, enlaces y conmutadores en una máquina con el sistema operativo Ubuntu 16.04. Como controlador se utilizó la Opendaylight en su versión Boron-SR4. Para el diseño de la red se programó un script en el lenguaje Python, el cual es inicializado a través de Mininet. La versión del protocolo Openflow que se utilizó fue Openflow 1.3 debido a las características de la versión de Mininet utilizada. La topología de la red es visualizada a través de la interfaz gráfica DLUX que proporciona el controlador Opendaylight, como se muestra en la figura 2. En la red los conmutadores openflow: 1 y openflow: 3 actúan como LERs, mientras que el conmutador openflow: 2 actúa como LSR.

Figura 2: Topología de la red.

Para la configuración y gestión de los flujos MPLS entre el host 1 y el host 2 se utilizó el protocolo OVSDB para establecer las acciones básicas PUSH, POP y SWAP de MPLS y establecer el LSP estático. Las configuraciones también pueden realizarse a través de algún gestionador de solicitudes HTTP como Postman que interactúe con la Rest API del controlador. En este caso se realizan desde el terminal de configuración del dispositivo openflow: 1 mediante el comando xterm s1. En la figura 3 se muestra la configuración de

los flujos a través de la herramienta de línea de comandos ovs-ofctl. Esta herramienta permite la gestión y monitoreo de conmutadores openflow a través de un terminal.

```
root@daniel:~# ovs-ofctl -O OpenFlow13 add-flow s1 in_port=2,ip,actions=push_mpls:0x8847,set_field:10-\>mpls_label,output:1
root@daniel:~# ovs-ofctl -O OpenFlow13 add-flow s2 in_port=1,dl_type=0x8847,mpls_label=10,actions=set_field:11-\>mpls_label,output:2
root@daniel:~# ovs-ofctl -O OpenFlow13 add-flow s3 in_port=1,dl_type=0x8847,mpls_label=11,actions=pop_mpls:0x0800,output:2
root@daniel:~#
root@daniel:~# ovs-ofctl -O OpenFlow13 add-flow s3 in_port=2,ip,actions=push_mpls:0x8847,set_field:20-\>mpls_label,output:1
root@daniel:~# ovs-ofctl -O OpenFlow13 add-flow s2 in_port=2,dl_type=0x8847,mpls_label=20,actions=set_field:21-\>mpls_label,output:1
root@daniel:~# ovs-ofctl -O OpenFlow13 add-flow s1 in_port=1,dl_type=0x8847,mpls_label=21,actions=pop_mpls:0x0800,output:2
```

Figura 3: Configuración de los flujos MPLS en cada conmutador.

Primeramente se añaden los flujos a cada dispositivo de la red en el sentido del LSP 1 (S1>S2>S3). En el dispositivo openflow: 1 o S1, el flujo añadido va a efectuar la acción PUSH_MPLS, la cual establece la etiqueta con valor 10 a todos los paquetes que entran por el puerto 2 y salen por el puerto 1. Seguidamente el dispositivo openflow: 2 o S2 va a recibir estos paquetes con etiqueta 10 por el puerto 1 y va a realizar la acción SET_FIELD, cambiando la etiqueta a 11 y reenviando el paquete por el puerto 2. Finalmente el dispositivo openflow: 3 o S3 recibe estos paquetes con etiqueta 11 por el puerto 1 y realiza la acción POP_MPLS, para retirar la etiqueta y enviar el paquete por el puerto 2. Para la completa configuración de la red se realiza la configuración anterior, pero en sentido del LSP 2 (S3>S2>S1). Para ello los paquetes que entran al S3 recibirán la etiqueta con valor 20, luego el S2 intercambia la etiqueta por el valor 21 y finalmente el S1 retira la etiqueta.
Después de configurar los flujos MPLS, es necesario comprobar que estos estén funcionando correctamente. Para el monitoreo de los flujos se realizó una prueba de conectividad haciendo ping entre los dispositivos host 1 y host 2 y luego comprobando las tablas de flujos. Los flujos instalados son gestionados en cada dispositivo openflow a través de ovs-ofctl. En cada uno de los dispositivos openflow se muestra su respectiva tabla de flujos. Como resultado se aprecian parámetros como duración, tabla de flujo en la cual es instalado, número de paquetes, número de bytes, puertos de entrada y salida y acciones que realiza el flujo configurado, como muestra la figura 4.

```
daniel@daniel:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=1107.482s, table=0, n_packets=256, n_bytes=25088, ip,in_port=2 actions=push_mpls:0x8847,set_field:10->mpls_label,output:1
 cookie=0x379, duration=1225.946s, table=0, n_packets=0, n_bytes=0, priority=10,mpls,in_port=2,mpls_label=11 actions=pop_mpls:0x0800,output:1
 cookie=0x0, duration=450.350s, table=0, n_packets=256, n_bytes=26112, mpls,in_port=1,mpls_label=21 actions=pop_mpls:0x0800,output:2
 cookie=0x2b00000000000005, duration=1225.936s, table=0, n_packets=247, n_bytes=20995, priority=100,dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x191, duration=1225.946s, table=0, n_packets=0, n_bytes=0, priority=8,ip,in_port=1,nw_dst=10.0.0.2 actions=push_mpls:0x8847,set_field:10->mpls_label,output:2
 cookie=0x2b0000000000001a, duration=1199.135s, table=0, n_packets=12, n_bytes=1046, priority=2,in_port=1 actions=output:2
 cookie=0x2b0000000000001b, duration=1199.134s, table=0, n_packets=7, n_bytes=430, priority=2,in_port=2 actions=output:1,CONTROLLER:65535
 cookie=0x2b00000000000005, duration=1225.935s, table=0, n_packets=0, n_bytes=0, priority=0 actions=drop
daniel@daniel:~$
daniel@daniel:~$
daniel@daniel:~$
daniel@daniel:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s2
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x379, duration=1461.391s, table=0, n_packets=0, n_bytes=0, priority=10,mpls,in_port=2,mpls_label=10 actions=pop_mpls:0x0800,output:1
 cookie=0x0, duration=1052.353s, table=0, n_packets=256, n_bytes=26112, mpls,in_port=1,mpls_label=10 actions=set_field:11->mpls_label,output:2
 cookie=0x0, duration=732.226s, table=0, n_packets=256, n_bytes=26112, mpls,in_port=2,mpls_label=20 actions=set_field:21->mpls_label,output:1
 cookie=0x2b00000000000006, duration=1461.377s, table=0, n_packets=585, n_bytes=49725, priority=100,dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x191, duration=1461.391s, table=0, n_packets=0, n_bytes=0, priority=8,ip,in_port=1,nw_dst=10.0.0.1 actions=push_mpls:0x8847,set_field:11->mpls_label,output:2
 cookie=0x2b00000000000018, duration=1446.582s, table=0, n_packets=10, n_bytes=906, priority=2,in_port=1 actions=output:2
 cookie=0x2b00000000000019, duration=1446.580s, table=0, n_packets=11, n_bytes=976, priority=2,in_port=2 actions=output:1
 cookie=0x2b00000000000006, duration=1461.377s, table=0, n_packets=0, n_bytes=0, priority=0 actions=drop
daniel@daniel:~$
daniel@daniel:~$
daniel@daniel:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s3
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=875.208s, table=0, n_packets=256, n_bytes=26112, mpls,in_port=1,mpls_label=11 actions=pop_mpls:0x0800,output:2
 cookie=0x0, duration=784.994s, table=0, n_packets=256, n_bytes=25088, ip,in_port=2 actions=push_mpls:0x8847,set_field:20->mpls_label,output:1
 cookie=0x2b00000000000007, duration=1459.578s, table=0, n_packets=291, n_bytes=24735, priority=100,dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x2b00000000000016, duration=1453.613s, table=0, n_packets=14, n_bytes=1452, priority=2,in_port=1 actions=output:2
 cookie=0x2b00000000000017, duration=1453.613s, table=0, n_packets=8, n_bytes=500, priority=2,in_port=2 actions=output:1,CONTROLLER:65535
 cookie=0x2b00000000000007, duration=1459.578s, table=0, n_packets=0, n_bytes=0, priority=0 actions=drop
```

Figura 4: Tabla de flujos de cada uno de los conmutadores.

CONCLUSIONES

La combinación de los mecanismos MPLS y la arquitectura SDN permiten la automatización de la red y sus operaciones a través de un control centralizado. Esto proporciona estadísticas de la red para su optimización y planificación. La plataforma de software centralizada que brinda SDN brinda una visión global de la red y múltiples interfaces y mecanismos para recolectar información en tiempo real. Además de la interacción entre las aplicaciones y la red, proporcionando abstracción y programabilidad e incorporando aspectos propios de MPLS como la ingeniería de tráfico y la segmentación del ruteo. Protocolos como OVSDB ofrecen la capacidad de configurar y clasificar los flujos para darles diferentes niveles de prioridad de acuerdo a los requerimientos de desempeño de la red o de determinada aplicación.

REFERENCIAS

1. [1] D. Kreutz, F. Ramos, P. Verissimo, C. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," Proceedings of the IEEE, vol. 103, pp. 14-76, 2015.
2. [2] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," IEEE Communications Surveys & Tutorials, vol. 17, pp. 27-51, 2015.
3. [3] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, "Survey on network virtualization hypervisors for software defined networking," arXiv preprint arXiv: 1506.07275, 2015.

4. [4] S. PremKumar and V. Saminadan, "Performance evaluation of smart grid communication network using MPLS," in Communication and Signal Processing (ICCSP), 2017 International Conference on, 2017, pp. 2116-2120.
5. [5] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, D. Meyer, and O. Koufopavlou, "Software-defined networking (SDN): Layers and architecture terminology," 2070-1721, 2015.
6. [6] S. Agarwal, M. Kodialam, and T. Lakshman, "Traffic engineering in software defined networks," in INFOCOM, 2013 Proceedings IEEE, 2013, pp. 2211-2219.
7. [7] M. Yu, J. Rexford, M. J. Freedman, and J. Wang, "Scalable flow-based networking with DIFANE," ACM SIGCOMM Computer Communication Review, vol. 41, pp. 351-362, 2011.
8. [8] M. Manel and Y. Habib, "An Efficient MPLS-Based Source Routing Scheme in Software-Defined Wide Area Networks (SD-WAN)," in Computer Systems and Applications (AICCSA), 2017 IEEE/ACS 14th International Conference on, 2017, pp. 1205-1211.
9. [9] L. Vdovin, P. Likin, and A. Vilchinskii, "Network utilization optimizer for SD-WAN," in Science and Technology Conference (Modern Networking Technologies)(MoNeTeC), 2014 First International, 2014, pp. 1-4.
10. [10] O. Michel and E. Keller, "SDN in wide-area networks: A survey," in Software Defined Systems (SDS), 2017 Fourth International Conference on, 2017, pp. 37-42.
11. [11] P. T. Congdon, P. Mohapatra, M. Farrens, and V. Akella, "Simultaneously reducing latency and power consumption in openflow switches," IEEE/ACM Transactions on Networking (TON), vol. 22, pp. 1007-1020, 2014.
12. [12] F. Hu, Q. Hao, and K. Bao, "A survey on software-defined network and openflow: From concept to implementation," IEEE Communications Surveys & Tutorials, vol. 16, pp. 2181-2206, 2014.
13. [13] Y. Hu, W. Wendong, X. Gong, X. Que, and C. Shiduan, "Reliability-aware controller placement for software-defined networks," in Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on, 2013, pp. 672-675.
14. [14] R. Amin, M. Reisslein, and N. Shah, "Hybrid SDN Networks: A Survey of Existing Approaches," IEEE Communications Surveys & Tutorials, 2018.
15. [15] H. Krishna, "Providing end-to-end bandwidth guarantees with openflow," 2016.
16. [16] B. Pfaff and B. Davie, "The open vSwitch database management protocol," 2070-1721, 2013.
17. [17] S. M. Nadaf, A. A. Kumar, H. K. Rath, and A. Simha, "OpenDaylight controller—Enhancements for a smoother software defined network," in 2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), 2017, pp. 1-6.
18. [18] D. Basivireddy, "OpenDaylight integration with OpenStack," International Journal of Science and Research (IJSR).

SOBRE LOS AUTORES

Daniel Iglesias de la Torre: Profesor Instructor, miembro del colectivo de la disciplina de Sistemas de Telecomunicaciones. Actualmente cursa la maestría de Telemática. Sus intereses de investigación se centran en: las redes definidas por software y la virtualización de las funciones de red.

Dr. C. Ing. Félix Florentino Alvarez Paliza: Profesor Titular, Jefe del colectivo de la disciplina de Sistemas de Telecomunicaciones, Coordinador Maestría de Telemática y Coordinador del Programa Doctoral Tutelar de Telecomunicaciones y Electrónica.

MSc. Arelis Ramos Fleites: Profesor Titular, miembro del colectivo de la disciplina de Sistemas de Telecomunicaciones. Sus intereses de investigación se centran en: las redes definidas por software y la virtualización de las funciones de red.