TRABAJO FINAL INTEGRADOR

TITULO

INTEGRACION DE REDES IP UTILIZANDO SDN

Alumno:
Ing. José Luis Gabriel Rodríguez Farfán

Carrera:

Especialización en Telecomunicaciones
Instituto Tecnológico de Buenos Aires

Tutor: Dr. Ing. Gustavo Hirchoren

BUENOS AIRES - ARGENTINA
2017

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS:

OBJETIVOS

I GENERAL

Este trabajo busca, a partir del concepto de las redes definidas por software “SDN”, observar la capacidad de interconexión de un switch tradicional con un switch híbrido OpenFlow, y el proceso de gestión de los flujos, a través de un controlador SDN como lo es OpenDayLight Beryllium.

II ESPECÍFICOS

- Se busca estudiar las ventajas que ofrece la arquitectura SDN para la gestión de los recursos, en las redes de datos.

- Se espera observar la interacción de los equipos físicos, con el router OpenFlow a través del controlador OpenDayLight.

- Analizar la respuesta de conmutación de la red entre una sucursal y el equipo de la casa central para observar los flujos, de la conexión hacia el controlador.

RESUMEN

Con el nuevo paradigma que son las redes definidas por software “SDN”, se buscó conceptualizar su operación a través de documentos y algunos tutoriales que ofrece el ONF (Open Networking Fundation), y la universidad de Princeton a través del portal Courcera, donde utilizando un emulador que opera bajo Linux, como lo es “Mininet” que permite generar topologías y por medio de controladores como NOX, POX y Ryu se gestionan los flujos de la red virtualizada [29].

En el transcurso de esta revisión bibliográfica se observó un controlador de última generación, que también interactúa con Mininet, posteriormente se halló una presentación realizada en la ciudad de Venecia (el 20/3/2014) sobre equipos Mikrotik, donde Andis Arins [1] muestra cómo a través de un router RB750GL utilizando un controlador como lo es POX, se gestionan los flujos de datos.

Después de consultar en el ONF (Open Networking Fundation), la definición del protocolo OpenFlow, y comprender su funcionamiento e interacción con el controlador SDN OpenDayLight, se planteó que la forma más adecuada de interconectar una red de equipos Mikrotik con OpenDayLight, es a través de OpenFlow. Por esta razón se conectaron dos routers Mikrotik a través de un switch core conectado al controlador SDN para poder observar los flujos de datos OpenFlow

En el capítulo uno se encuentra una breve contextualización histórica y teórica de las redes definidas por software, observando cómo se convierten en una herramienta para gestionar el alto volumen de datos y virtualizar su operación.

En el capítulo dos se encuentran los parámetros de configuración en la maqueta implementada, simulando la conexión de una red compuesta por un punto central y dos sitios remotos, para lo cual se utilizaron tres equipos Mikrotik conectados a través de OpenFlow 1.0.

En el capítulo tres se encuentran los resultados del trabajo y las conclusiones se presentan en el capítulo cuatro.

En el capítulo cinco se dan recomendaciones para el desarrollo de futuros trabajos, más una breve reseña de algunos fabricantes que integraron OpenFlow a sus equipos SDN.

En el capítulo seis se incluyen los anexos con el código XML generado por el controlador SDN a través de OpenFlow definiendo los flujos gestionados desde OpenDayLight.

En el capítulo siete se encuentra la bibliografía utilizada para el desarrollo de este trabajo y se finaliza con un glosario con los términos específicos.

PLANTEO DEL ÁREA DE ESTUDIO

Como hipótesis se buscó observar qué tan accesible es una red IP a través de OpenFlow; y se encontró que una red IP es gestionable a través de este protocolo utilizando como recurso de acceso un switch Mikrotik, al cual se instaló y habilitó el protocolo OpenFlow, para que se conecte al controlador SDN OpenDayLight.

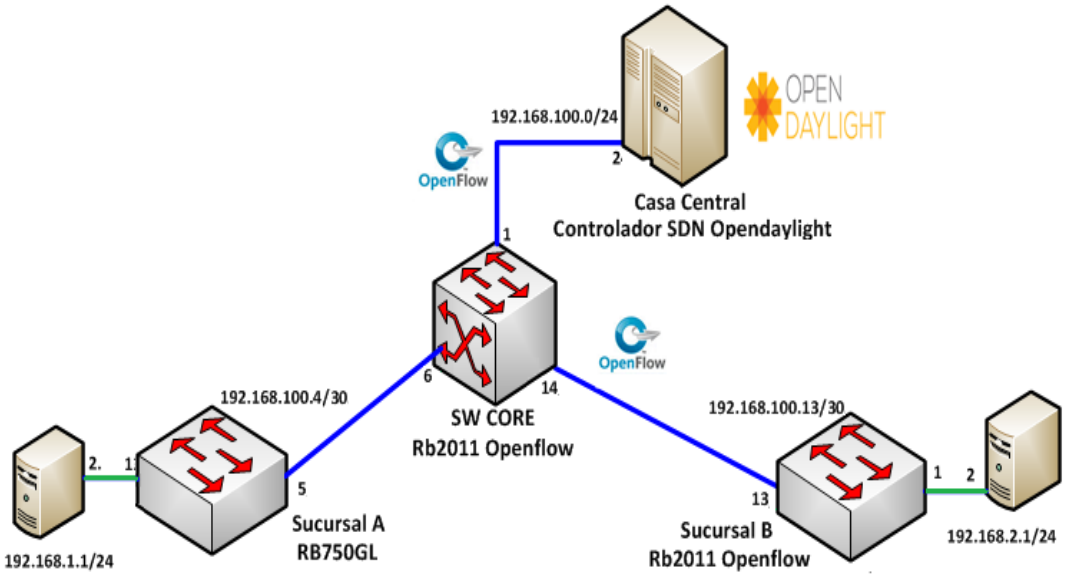

Figura 1: Topología propuesta para observar la interacción con SDN/ OpenFlow

Para este fin se implementó la topología que aparece en la figura 1, donde se conecta un router capa cuatro (con funcionalidad OpenFlow) al controlador SDN OpenDayLight, para establecer el canal OpenFlow y abstraer la gráfica con la topología de la red; para luego gestionar la adición del flujo de datos y poder observar la respuesta de los flujos a través del controlador SDN.

Después de establecer la gestión de la red a través de OpenFlow se buscó resolver las siguientes preguntas planteadas en el anteproyecto original:

- ¿Es provechosa la administración de los flujos de red a través de SDN?
- ¿Se puede utilizar SDN para administrar conexiones redundantes?
- ¿Qué ventajas trae integrar una red IP a una solución SDN y hasta qué punto es beneficioso agregar un router híbrido a una red ya implementada?

Capítulo 1

1 MARCO TEÓRICO

1.1 Redes Definidas por Software (SDN)

Las redes SDN virtualizan la infraestructura de red, separando el plano de control del plano de distribución o forwarding. Esto favorece una arquitectura centralizada: dinámica, automatizada, y rentable [3]. Esta nueva arquitectura, surge para cubrir los requerimientos del plano de control implementándolo a través de software.

En las redes de datos se cuenta con dos recursos: el plano de control que utiliza información de la señalización para tomar decisiones de control, mientras el plano de datos está encargado de la transmisión de datos a los usuarios [5].

1.2 Evolución de las Redes Definidas por Software

Las redes definidas por software son el resultado de la innovación y la confianza en el desarrollo de estándares abiertos. SDN se desarrolla junto con OpenFlow como resultado del trabajo de doctorado de Martín Casado en la Universidad de Stanford, donde fue alumno del Dr. Nick McKeown, profesor de ingeniería eléctrica e informática [1]. Su trabajo genera la primera propuesta de OpenFlow: un API de comunicaciones para un switch virtual.

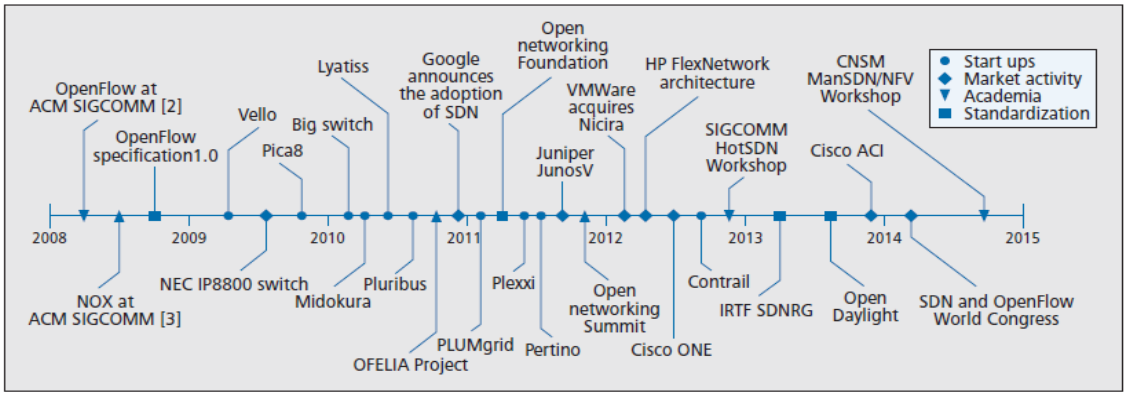

Figura 2: Línea de tiempo evolución del SDN 2008-2015 [2]

Esta idea se desarrolla y consolida, hasta el punto que Google adopta este concepto en el 2011 para aplicarlo en sus centros de datos consolidándose como un protocolo de comunicaciones abierto; con lo cual es creada la (ONF), auspiciada por Linux Fundation, junto a los fabricantes de hardware y actores de los centros de datos como: HP, Cisco, Juniper, Google, Facebook, Amazon,Yahoo, entre otros. Este proceso se observa en la figura 2.

Fruto de este desarrollo se generan controladores como: Ryu y OpenDayLight, entre otros. Este último se consolida entre los fabricantes por su estabilidad, y amplia capacidad de recursos, ya que su desarrollo es generado a través de grupos de programadores que aportan código a los proyectos colaborativos activos en comunidades como lo es www.github.com. Un controlador SDN se encarga de coordinar todos los flujos de datos dentro de una red SDN.

AÑO	CONTROLADOR
abr-13	Daylight
feb-14	Hydrogen
sep-14	Hellium
jun-15	Lithium
feb-16	Beryllium

Tabla 1: Versiones lanzadas del controlador OpenDayLight [3]

Desde su aparición en 2013 OpenDayLight ha presentado cada año una nueva versión, fruto de las líneas de código aportadas por grupos de desarrolladores auspiciados por las empresas "socio", y los usuarios que participan de esta comunidad siendo una de las más recientes y estables la versión OpenDayLight "Beryllium" lanzada el 16 de febrero del 2016; posteriormente fue lanzada la versión de Boron en noviembre del 2016 pero actualmente continúa en evolución. El histórico de las versiones ODL se puede observar en la tabla 1.

CONTROLADOR	OPENDAYLIGHT	RYU	FLOODLIGHT	ONOS
LINEAS DE CODIGO	2500 000	116 000	100 000	44000
DESARROLLO ANUAL	3	4	4	1

Tabla 2: Líneas de código aportadas a cada controlador [4]

En la tabla 2 están descritas el número de líneas aportadas a cada uno de los principales controladores SDN, donde OpenDayLight se consolida como el controlador con mayor número de líneas de código aportadas por sus patrocinadores y usuarios con 2.500.000 líneas aportadas [4].

1.2.1 ¿Cómo funciona SDN?

Las redes definidas por software tienen definido un plano de control centralizado a través de un controlador que dispone de recursos para gestionar las interfaces físicas a través del API Southbound y protocolos como OpenFlow, que permiten separar el plano de control del plano del plano de forwarding, o distribución. A nivel de aplicación el plano de control se comunica a través de un API Northbound que mejora la administración de los datos, como aparece en la figura 3.

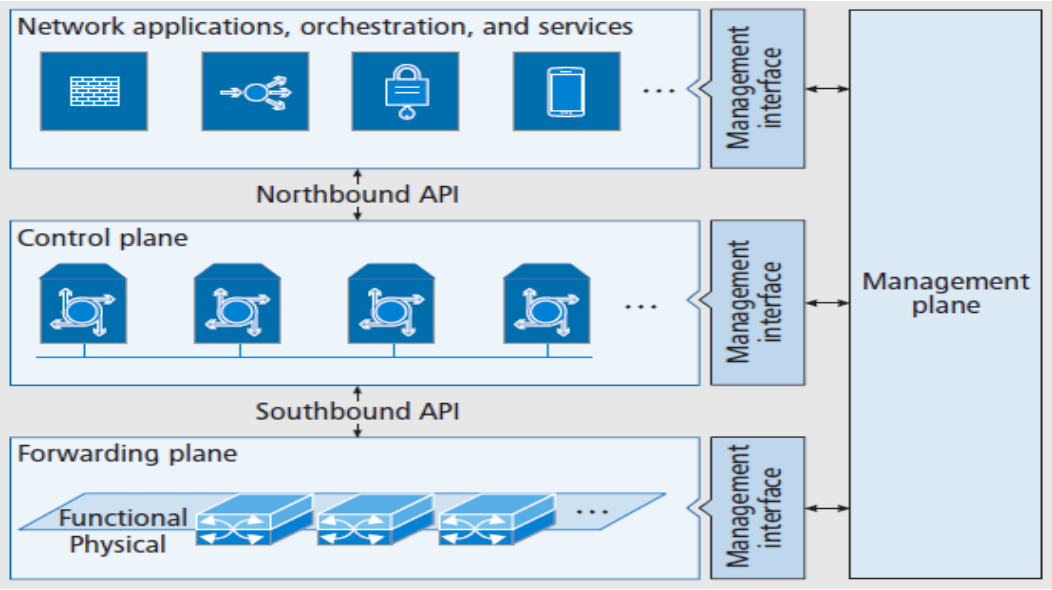

Figura 3: Arquitectura y detalle de una solución SDN [7].

El ONF aclara que la arquitectura SDN está compuesta por tres diferentes capas accesibles a través de API [3].

- **The Application Layer** (Network Aplications, Orchestation and Services): en esta capa, se encuentran las aplicaciones del usuario final, las cuales reciben e interactúan con el negocio; en la capa de aplicación están los servicios.
- **The Control Layer:** proporciona el control lógico centralizado, con la función de supervisar el reenvío de paquetes a través de una interfaz activa.
- **The Infraestructure Layer:** lo componen los elementos de la red y los dispositivos que proporcionan la conmutación y reenvío de paquetes [5].

La comunicación entre las capas se realiza a través de los recursos proporcionados por las API, a nivel superior entre el control y la aplicación interviene el API Northbound; y para comunicar el control con el forwarding (distribución) se utiliza el API Southbound [7].

1.3 OPENFLOW

Es un protocolo standard de comunicación definido entre el control y la capa de forwarding; en la arquitectura SDN, OpenFlow permite acceso directo y manipulación de los equipos en los planos de red y forwarding como switch físicos o virtuales activos en un hypervisor [9].

OpenFlow facilita la interoperabilidad de los diferentes fabricantes, con sus equipos y las API, permitiendo controlar una red programada de forma independiente de los switch.

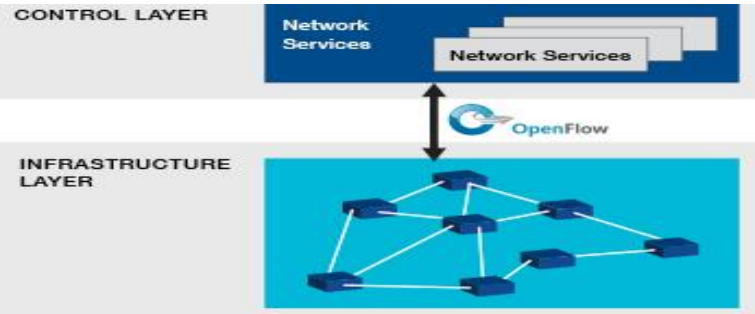

Figura 4: Operación OpenFlow en la Arquitectura SDN [30]

1.3.1 Puertos OpenFlow

Los switch OpenFlow se conectan entre sí a través de puertos OpenFlow, donde un paquete se puede reenviar de un switch OpenFlow a otro switch OpenFlow, definiendo: un puerto de salida en el primero y un puerto de entrada en el segundo.

Los puertos OpenFlow están definidos como puertos físicos, puertos lógicos y puertos locales reservados (si en el equipo son soportados).

- **Puertos standard:** pueden utilizarse como puertos de entrada o salida, cuando son utilizados en grupo estos puertos son contadores y tienen un estado en la configuración.

- **Puertos lógicos:** estos puertos no corresponden directamente a las interfaces físicas del switch. Cuentan con un alto nivel de abstracción; están definidos en forma de túneles, interfaces loopback o puertos y conexiones de agregación.

- **Puertos reservados:** en estos puertos se especifican las acciones genéricas de reenvío, como el envío al controlador, inundaciones, o reenvío como procesamiento de switch tradicional.

1.3.2 Componentes de un Switch OpenFlow

El switch está compuesto varias tablas de flujos con sus respectivos grupos de tablas asignadas, en las cuales un flujo de datos es comparado, para aplicar reglas de búsqueda y distribución de paquetes.

El controlador gestiona al switch a través del protocolo OpenFlow, el cual permite añadir, actualizar y borrar los flujos de entrada, tanto de forma proactiva como reactiva [8].

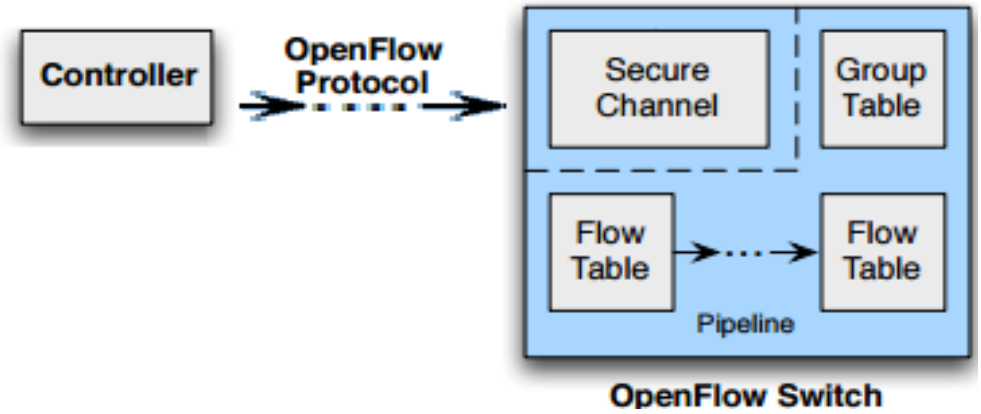

Figura 5: Comunicación segura para un switch OpenFlow hacia el controlador [11]

El controlador se comunica con otros switches o un controlador SDN externo, a través de un canal OpenFlow, el cual es una abstracción que se asocia al hardware real del switch, para ser mapeado continuamente y que la red OpenFlow se comporte de forma coherente [8].

Adicionalmente se generan estadísticas con un registro, el número de paquetes y bytes para cada flujo, como también el tiempo desde que se aplicó la regla a un paquete, para facilitar la eliminación de los flujos inactivos y permitir la llegada de nuevos flujos.

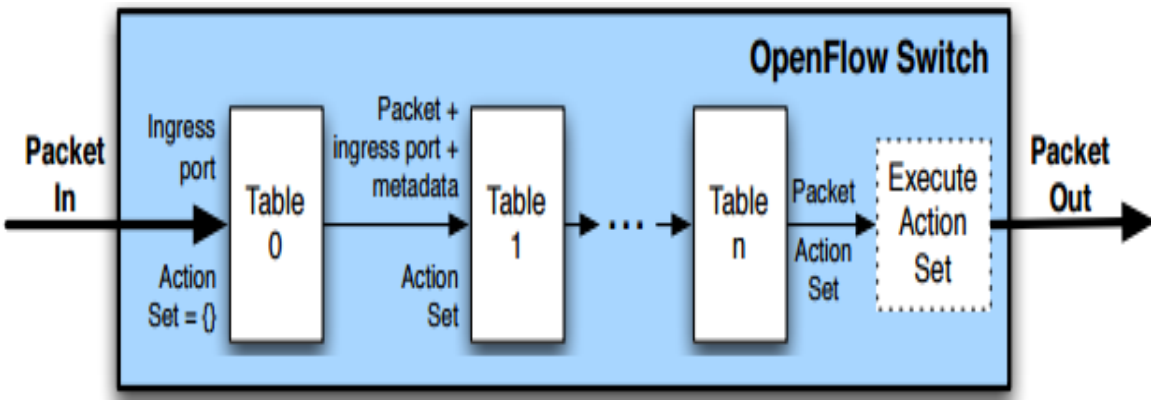

Figura 6: Procesamiento en las tablas de un Flujo de los canales de datos [11]

Tras recibir un paquete, el switch OpenFlow en cada una de las tablas de flujo, valida si se produce match y al coincidir con alguna tabla le aplica las acciones definidas; adiciona la metadata y de ser necesario, le permite el paso a la siguiente tabla, como aparece en la figura 6 y 7.

El switch inicia la búsqueda en la primera tabla de flujo, y hace procesamiento en paralelo.

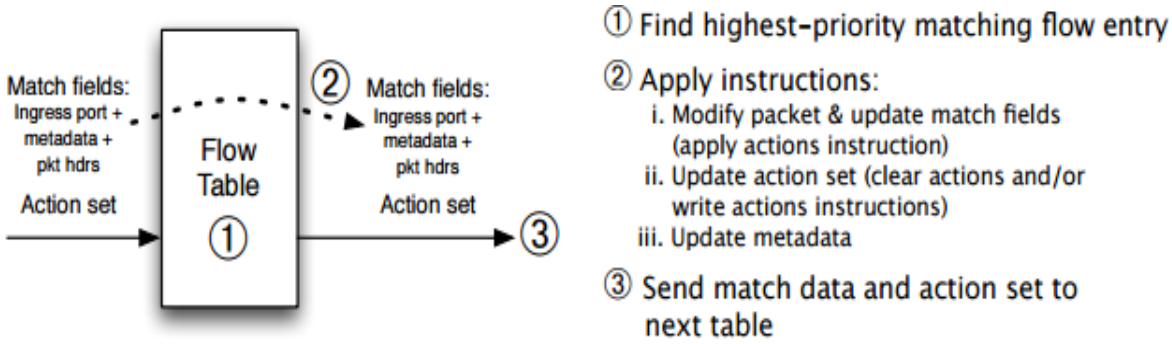

Figura 7: Procesamiento de un paquete en las tablas de un Flujo [11]

1.3.3 Tipos de Switch OpenFlow

Un switch híbrido puede conectar el tráfico de OpenFlow con Ethernet a través de los puertos reservados. El hardware del switch OpenFlow puede ser virtualizado, para soportar múltiples instancias de conmutación.

En los switches que sólo hablan OpenFlow se reciben el procesamiento de los paquetes enviados de otros switches OpenFlow; mientras que en los switches híbridos se soporta la operación OpenFlow y de forma normal la conmutación IP.

1.3.4 El Canal OpenFlow

Esta interfaz conecta a cada switch OpenFlow con el controlador; para configurar, administrar el switch, recibir eventos y coordinar los flujos de paquetes.

En la comunicación, se establece un canal OpenFlow; el cual es usualmente encriptado utilizando Transport Layer Security (TLS) aunque corre directamente sobre TCP/IP. Si falla la conexión, es detectada por los valores TCP y se aplican los tiempos de espera de la sesión TLS.

El puerto de transporte para comunicarse en OpenFlow 1.0 es el #6633; mientras en la versión 1.3 utiliza el puerto 6653. Esta comunicación en OpenFlow inicia la negociación con la versión más alta, y en su defecto toma la más baja [11].

1.3.5 Encripción TLS

Transport Layer Security es el modo seguro por defecto en OpenFlow, para el cual la versión 1.2 de TLS o superior, proporciona autenticación y cifrado de la conexión para que se conecte el switch con el controlador por el puerto 6633; quedando el controlador en modo escucha del puerto predeterminado.

Otra alternativa es utilizar TCP plano, e implementar un túnel IPSec o VPN para definir una red física separada sin conexiones URI y TCP válidos, entre otras medidas para mantener la privacidad e integridad del controlador, evitando su suplantación u otros tipos de ataques al canal OpenFlow [11].

1.4 Micro-Servicios

Para hablar de los componentes de OpenDayLight es necesario contextualizar cómo funcionan y qué son los micro-servicios. En todos los desarrollos de software hay una capa de servicios, soportada por otros recursos de apoyo. Generalmente los servicios se agrupan compartiendo el mismo contexto, después pueden aislarse mejorando su funcionalidad la cual es publicada a través de REST.(Representación del estado de transferencia) .

Al estar REST publicando los servicios, se convierten en aplicaciones independientes, con lo cual se aíslan en un contenedor independiente, que separa esta aplicación de las otras.

Con esta arquitectura se consigue facilitar el despliegue de aplicaciones, que se reducen a micro-servicios, los cuales son alcanzables desde todo tipo de dispositivos ya que es publicado vía REST [33], como aparece en la figura 8.

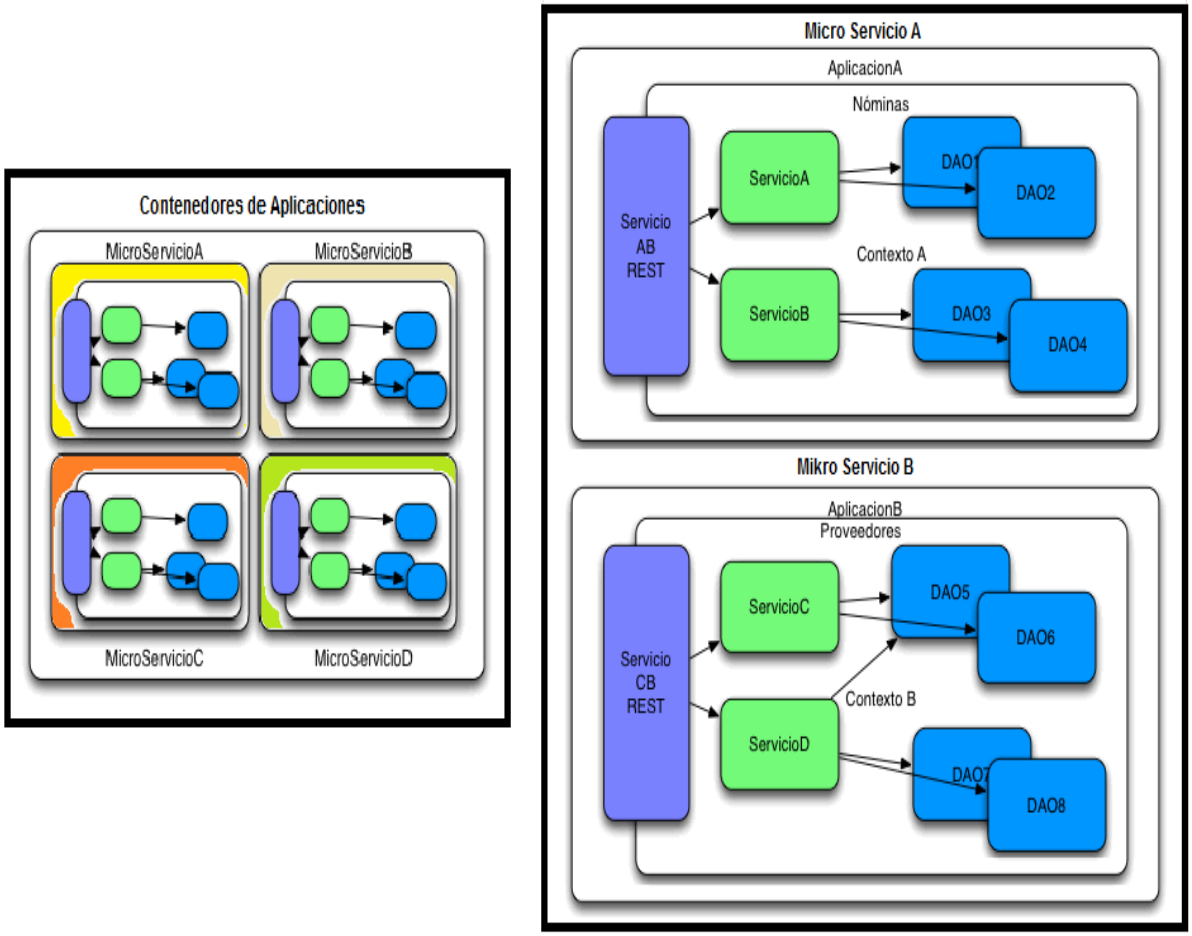

Figura 8: Jerarquía: contenedor, micro-servicio, aplicación, servicio, grupo de servicios, aplicaciones y recursos comunes. [33]

1.5 El Controlador SDN OpenDayLight

El controlador de red SDN ODL, es una plataforma modular que reutiliza servicios e interfaces comunes para construir y ejecutar aplicaciones, aprovechando las funcionalidades de otros paquetes, exportando servicios a través de interfaces Java. Muchos de estos servicios funcionan bajo el modelo proveedor-consumidor a través de una capa de adaptación llamada MD-SAL [12].

Desde allí MD-SAL (la capa de abstracción del servicio) interactúa a nivel superior con el API REST, que conecta directamente el control con la capa de aplicación a través del API Northbound, y de igual forma comunica el control con la capa de Forwarding a través de MD-SAL, junto con el API Southbound y protocolos como OpenFlow como aparece en la figura 9.

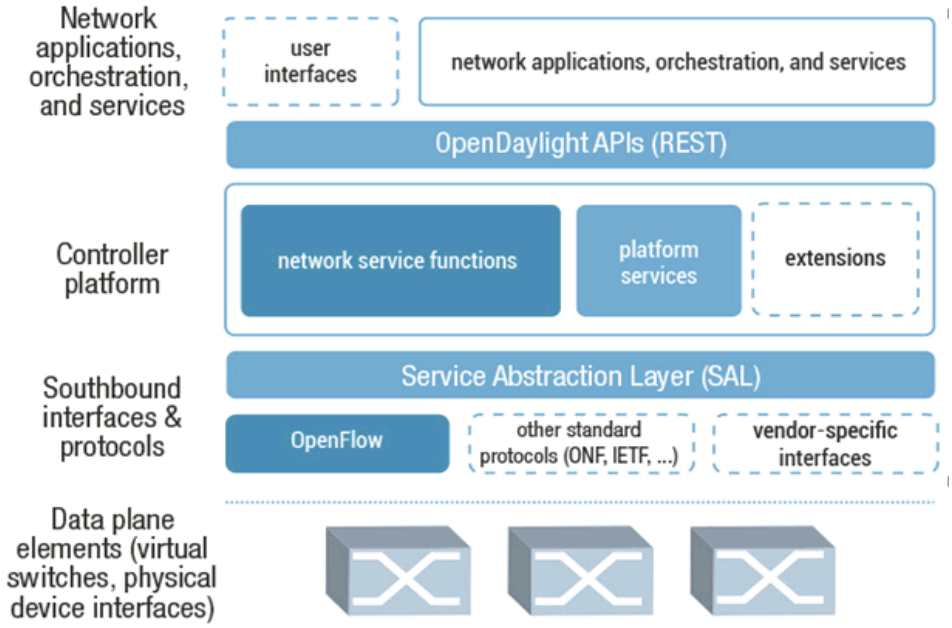

Figura 9: Bloques Northbound y Southbound de ODL [17]

OpenDayLight utiliza a Maven POM.xml (Project Object Model), una herramienta de Java, para poder generar dependencias entre otros módulos y componentes externos. Maven está construido con una arquitectura basada en plugins que permite utilizar cualquier aplicación a través de una entrada estándar [14].

1.5.1 Módulos de OpenDayLight

ODL es una plataforma en la cual sus módulos reutilizan servicios e interfaces comunes a través de Java. Muchos de estos servicios están construidos en un modelo proveedor-consumidor, a través de la capa de adaptación de servicio MD-SAL, que reúne las funcionalidades de muchas aplicaciones, las cuales prestan sus servicios, para que desarrolle las tareas del controlador SDN.

El core de la plataforma es una librería de almacenamiento que conserva dos grupos: los datos de configuración (que mantienen el estado de la red deseada) y los datos de funcionamiento (que representan el estado real de la red sobre la base de datos de los elementos de red gestionados). De esta forma, todas las llamadas generadas por eventos y datos van de “Proveedor” a un “Consumidor” a través de este almacén utilizando la lógica de MD-SAL.

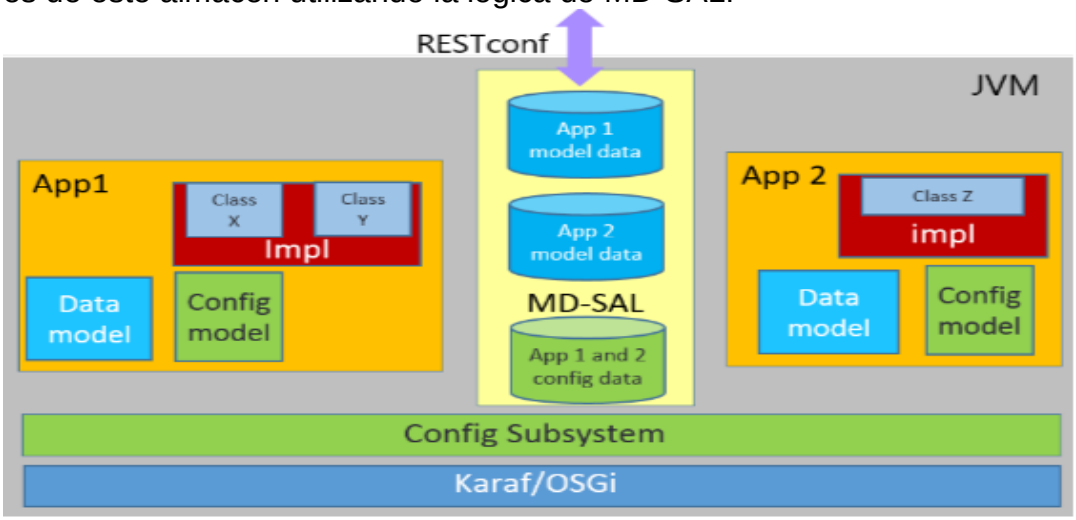

Figura 10: Marco del Service Abstraction Layer en MD-SAL [24]

Un usuario puede publicar datos a través de MD-SAL o REST, almacenando los objetos individuales con la jerarquía padre-hijo, los cuales son accesibles a través de la instancia Yang. Para este fin OpenFlow 1.1 los conecta como protocolo que se guarda en el almacén de datos. Accediendo a los detalles del nodo conector, a través del identificador de nodo mediante Yang o utilizando la URL de la configuración del flujo de datos a través de Rest-Conf, se puede consultar, crear o modificar los flujos a partir de nuevas tablas de flujo [24].

1.5.2 OSGI: Open Services Gateway Initiative

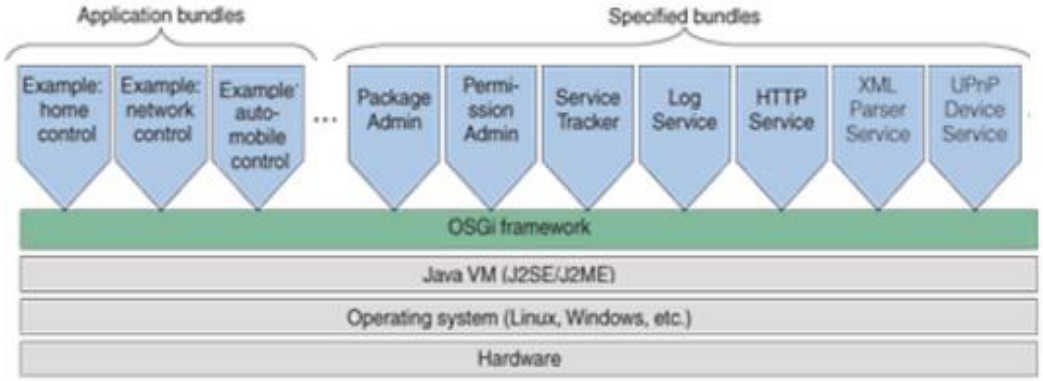

Figura 11: Open Services Gateway Initiative. OSGi [14]

OSGI, tiene como objetivo definir las especificaciones de software que permiten diseñar plataformas compatibles que generan múltiples servicios [13].

El Framework OSGI es el corazón del controlador, el middleware que permite vincular dinámicamente plugins para los protocolos Southbound en desarrollo.

1.5.3 Karaf

Es un entorno de ejecución en Apache, un contenedor para OSGI que proporciona un pequeño ecosistema con recursos que permite el despliegue en caliente de aplicaciones (Karaf detecta el tipo de archivo que es y trata de implementarlo). Es gestionable a través de la consola de Unix y está centrado en el directorio raíz. Al detectar cambios en el archivo de forma dinámica él lo vuelve a cargar [15].

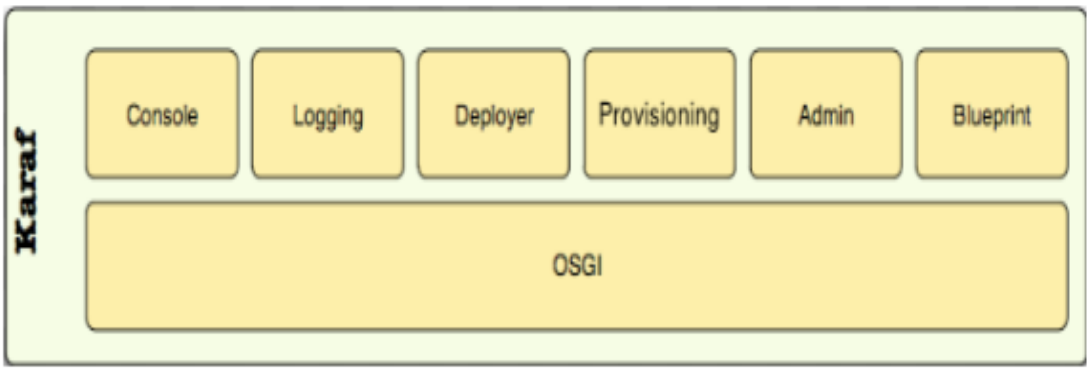

Figura 12: Componentes y recursos de Karaf

Karaf proporciona un grupo de paquetes con dependencias opcionales y datos que permiten instalar otros plugins en una solución de micro-servicios, para la integración de sistemas de datos más grandes.

1.5.4 Service Abstraction Layer

La solicitud de servicio en el Service Abstraction Layer, se correlaciona con el plugin adecuado utilizando el protocolo Southbound más apropiado para interactuar con un dispositivo de red dado. Cada plugin es independiente uno del otro y están menos vinculados con la SAL [17].
SAL ofrece servicios básicos, como la detección de los dispositivos que son utilizados por los módulos Northbound, como el administrador de topología de la red para construir las capas de topología del dispositivo.

1.5.5 API REST

El controlador ofrece APIs abiertas en la capa Northbound que son usadas por las aplicaciones; la característica que el marco OSGI y Rest es bidireccional y compatible con las API Northbound [17]. Las API Northbound actúan como gestores del equipo anfitrión, para el programador de flujo, en el enrutamiento estático, entre otras funciones.

1.5.6 Modelo de YANG "Yet Another Next Generation"

YANG es un lenguaje para describir la estructura básica de algunos datos de aplicaciones que se almacenan en una jerarquía de árbol dentro de los contenedores.

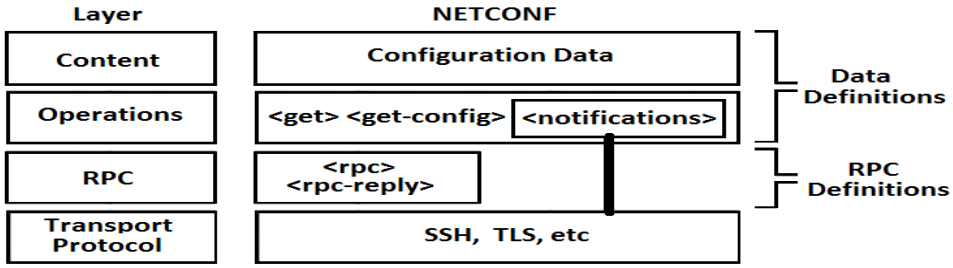

Figura13: Modelo de Capas YANG a nivel de red NETCONF (22)

Yang puede ser utilizado, para definir el formato, de eventos o notificaciones emitidas por los elementos de red, y permite modelar datos para definir la firma de procedimientos remotos, que pueden ser invocados en los elementos de red a través del protocolo NETCONF. YANG es un lenguaje modular que representa la estructura de datos en un árbol XML [16].

1.5.7 NETCONF RESTCONF Y YANG

El factor más prometedor de las redes definidas por software es el poder generar automatización y predicción de tráfico; al centralizar el tráfico en un controlador permite producir la integración de protocolos como NETCONF, que define un conjunto de operaciones, que se pueden realizar en la configuración de los almacenes de datos, el CRUD (Crear, Leer, Actualizar y Borrar) [39]
YANG define la sintaxis y la semántica de contenido del almacén de datos, configuración, datos de estado, operaciones de RPC y notificaciones de eventos.

RESTCONF no especifica recursos de operaciones obligatorios, la semántica de cada operación determina como se accede a los datastores. Los datos de configuración se pueden modificar con los comandos: DELETE, PATCH, POST y PUT. Los datos se codifican con XML o JSON. [39]

RESTCONF combina la simplicidad de HTTP con la predictibilidad y potencial de automatización de una API basada en esquemas. Conociendo los módulos YANG utilizados por el servidor, un cliente puede derivar todas las URL de recursos de gestión y la estructura adecuada de todas las solicitudes y respuestas RESTCONF.
De esta manera los URI para las operaciones RPC y el contenido del almacén de datos específicos del modelo de datos son predecibles, basándose en las definiciones del modelo YANG.
Las funcionalidades de estos protocolos en el caso de OpenDayLight permiten: modificar, agregar, crear, editar y borrar flujos y tablas de datos en la aplicación Yang UI [39].

1.5.8 Dlux

Es la interfaz de usuario para la gestión de red del controlador ODL, que recibe información de los diversos módulos interdependientes a través de los servicios de la MD-SAL. Dlux también utiliza los servicios de la MD-SAL para obtener información de la red y lo utilizan para proporcionar funciones de administración de red [19].

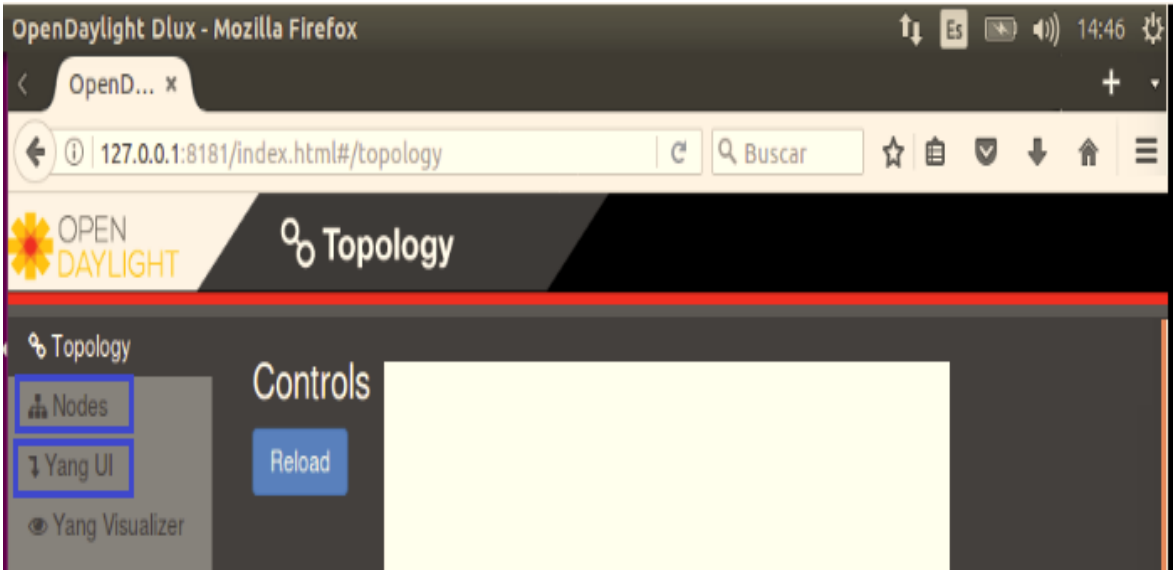

Figura 14: Consola gráfica YANG ODL

Para iniciar sesión, después de implementar Dlux con Karaf, la URI de inicio de sesión es http:/:8181/Dlux/index.html, siendo admin el único tipo de usuario disponible para Dlux en esta versión [17].

La interfaz gráfica de Dlux tiene en la parte superior izquierda, la pestaña Topology en la cual se encuentra una abstracción de la red que tiene conectada, como se aprecia en la figura 14. En la pestaña de Nodes se encuentra una tabla con los nodos descubiertos y el tráfico que se está cursado sobre cada interfaz.
En la pestaña de Yang UI se encuentra una consola que permite configurar y definir las políticas implementadas en la red. En la pestaña Yang Visualizer se encuentra un árbol con todos los nodos de la red.

1.6 Router Mikrotik RB750 GL

Es un switch de capa cuatro con facilidades para generar router virtuales a través de la función MetaRouter, maneja conexiones MPLS, permite crear VRF y servicios de firewall, cuenta con cuatro puertos 10/100/1000 Gigabit Ethernet, tiene recursos POE (In), tiene 64 MBytes de RAM y actualizando su versión de Router OS se puede habilitar la funcionalidad de OpenFlow [24].

Figura 15: Router Mikrotik RB750GL y Mikrotik RB2011IL-IN [24]

1.6.1 Router Mikrotik RB2011IL-IN

Es un switch capa cuatro de la familia RouterBoard, con facilidades para generar router virtuales a través de la función Metarouter, maneja conexiones MPLS, presenta funcionalidades POE, (In/Out) maneja cinco puertos 10/100 Ethernet, cinco puertos Gigabit Ethernet y un slot para conectar un SFP monomodo o multimodo, cuenta con recursos SDN.

Es virtualizable, cuenta con una capacidad de almacenamiento de 128 MBytes y 128 MBytes de RAM, para desarrollo de aplicaciones; maneja una potencia de hasta 30dBm en su conexión inalámbrica, cumpliendo con el estándar 802.11b/g/n [25].
Estos equipos fueron seleccionados por su alta performance, sus amplios recursos de conectividad, su completo set de herramientas, flexibilidad y su bajo costo.

Capítulo 2

Este capítulo se puede desglosar en tres partes: se inicia con las adecuaciones e implementaciones necesarias para producir la interconexión de los equipos Mikrotik, a nivel IP y habilitar la conexión OpenFlow en el Switch Mikrotik.

En la segunda parte se describe la configuración y operación del controlador SDN OpenDayLight, en la forma como se estableció el canal OpenFlow y los datos que se capturaron en el momento de la conexión.

Finalmente se encuentra la operación del controlador OpenDayLight donde se muestra como abstrae la gráfica con los datos de los equipos de la red conectados al controlador ODL y como a través de la consola Yang UI fue posible traer la definición de los flujos OpenFlow de la red activa.

2.1 Adecuación de los equipos

Para configurar los equipos Mikrotik, se habilita el API de Winbox, (Api de configuración de Mikrotik. que permite conectarse por dirección MAC o IP, a la sesión web de los equipos Mikrotik, utilizando el usuario y contraseña admin. Los equipos Mikrotik también son accesibles por línea de comando, vía telnet o SSH.

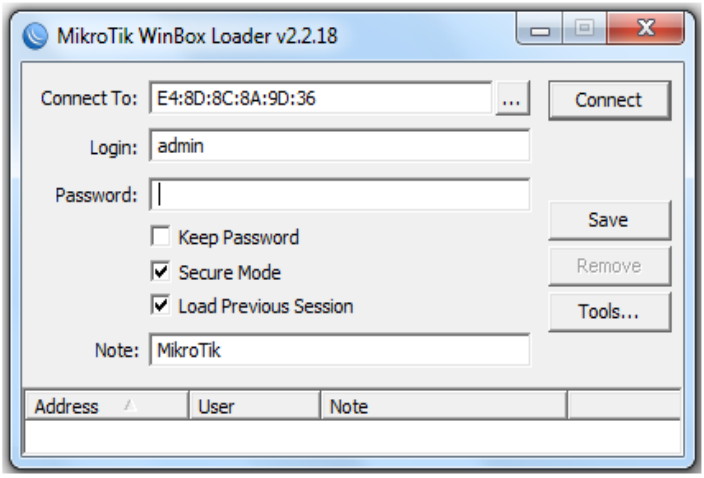

Figura 16: Consola Winbox para gestión de los Switch Mikrotik.

En la versión del firmware nativo de los RB750GL y RB2011IL-IN no está disponible la funcionalidad de OpenFlow, pero realizando una actualización del RouterOS a una versión como la 6.36 o superior, los switches quedan habilitados para OpenFlow.

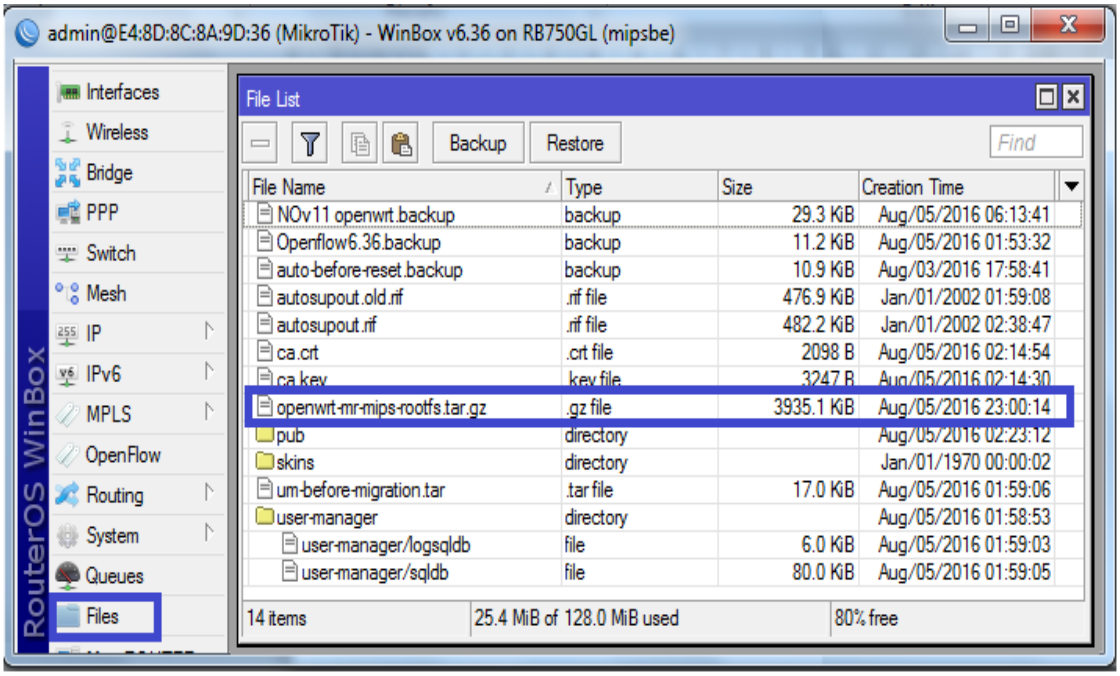

Figura 17: Actualización del IOS del Mikrotik RB750GL.

Al configurar el protocolo OpenFlow por línea de comando como recomienda la guía de Mikrotik, se perdió la gestión de los equipos, razón por la cual se llevó el equipo a su condición inicial para recuperar la gestión. Al activar OpenFlow utilizando el modo gráfico, se consiguió establecer la conexión contra el controlador; reservando los cinco puertos del primer módulo del switch para la conexión local IP y el otro módulo para OpenFlow.

2.1.1 Implementación e Integración de la Maqueta

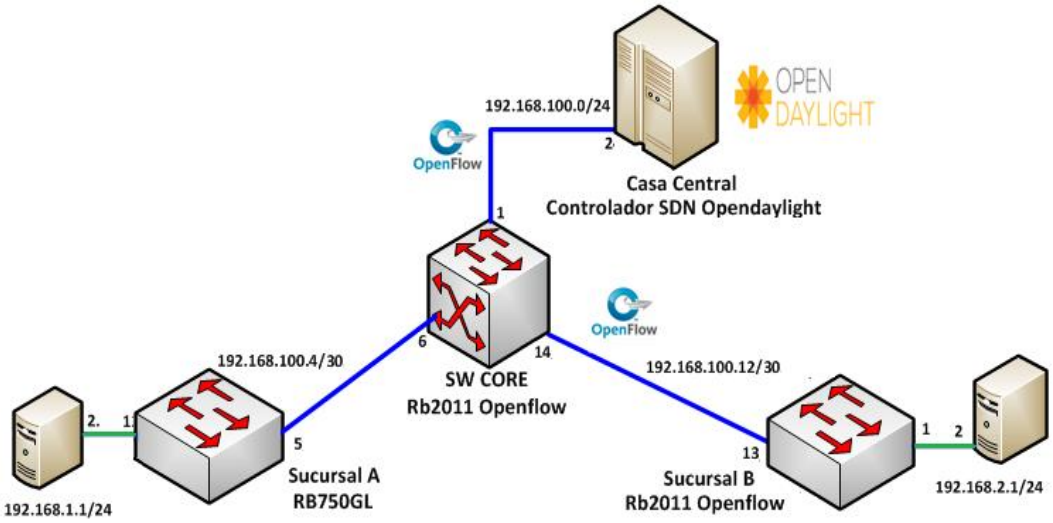

Figura 18: Conectividad IP entre el controlador y los Switch Mikrotik.

El montaje inicial, está compuesto por tres switches: el primero corresponde a un RB2011IL-IN que funciona como nodo principal o casa central, el cual se conecta

al controlador SDN OpenDayLight, a través del puerto 6, para establecer la conexión OpenFlow; adicionalmente se conecta la sucursal A y B a través del puerto 7 y 8 habilitados para OpenFlow. En el caso de la sucursal A está conectado un RB750GL y en la sucursal B un RB2011IL-IN.

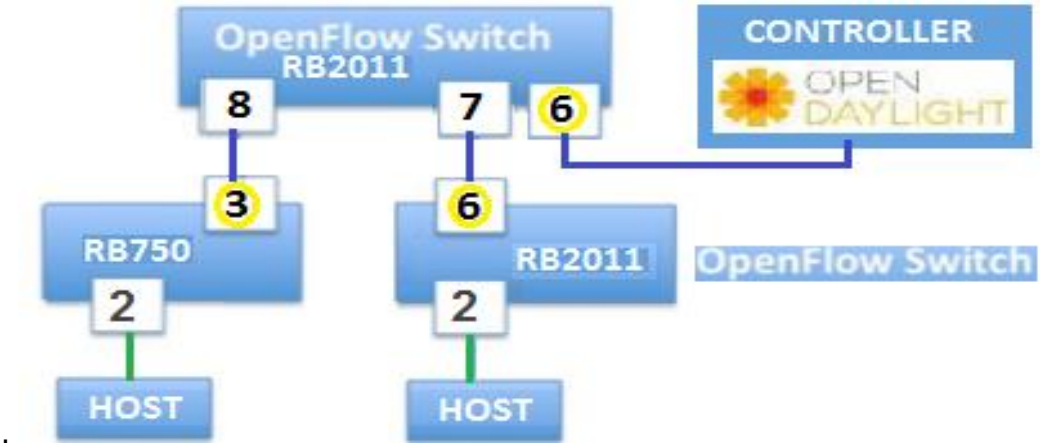

Figura 19: Cableado de la maqueta.

En la sucursal A el switch RB750GL, está conectado al switch Core a través del puerto tres y el host de la sucursal se conecta a través del puerto dos. En la sucursal B. El switch RB 2011IL-IN tiene conectado por el puerto 6 al switch core de la casa central y por el puerto 2 el host de la sucursal.
En la tabla tres, aparece el direccionamiento definido para la red bajo el segmento: 192.168.100.0/24, que conecta a todos los equipos al Switch Core a través de OpenFlow, como se ilustra en la figura 18 y 19.

Puerto	Sucursal A	Sucursal B	Casa Central	Función
ETH 2	192.168.1.1	192.168.2.1		**LAN IP**
ETH 6		192.169.100.13	192.169.100.1	**WAN OpenFlow**
ETH 3	192.169.100.5			**WAN OpenFlow**
ETH 7			192.168.100.6	**Conecta contra A**
ETH 8			192.168.100.14	**Conecta contra B**

Tabla 3: Direccionamiento conexión controlador SDN a Switch OpenFlow

Después de establecer la conectividad IP, entre el controlador SDN y el puerto seis del Switch core, RB2011IL-IN con la dirección IP 192.168.100.1/24, se define un flujo en la pestaña OpenFlow, para que se conecte con el controlador a través del puerto 6633 y se obtiene visibilidad OpenFlow de las interfaces 7 y 8 sobre el switch de Casa central, estableciendo el canal OpenFlow.

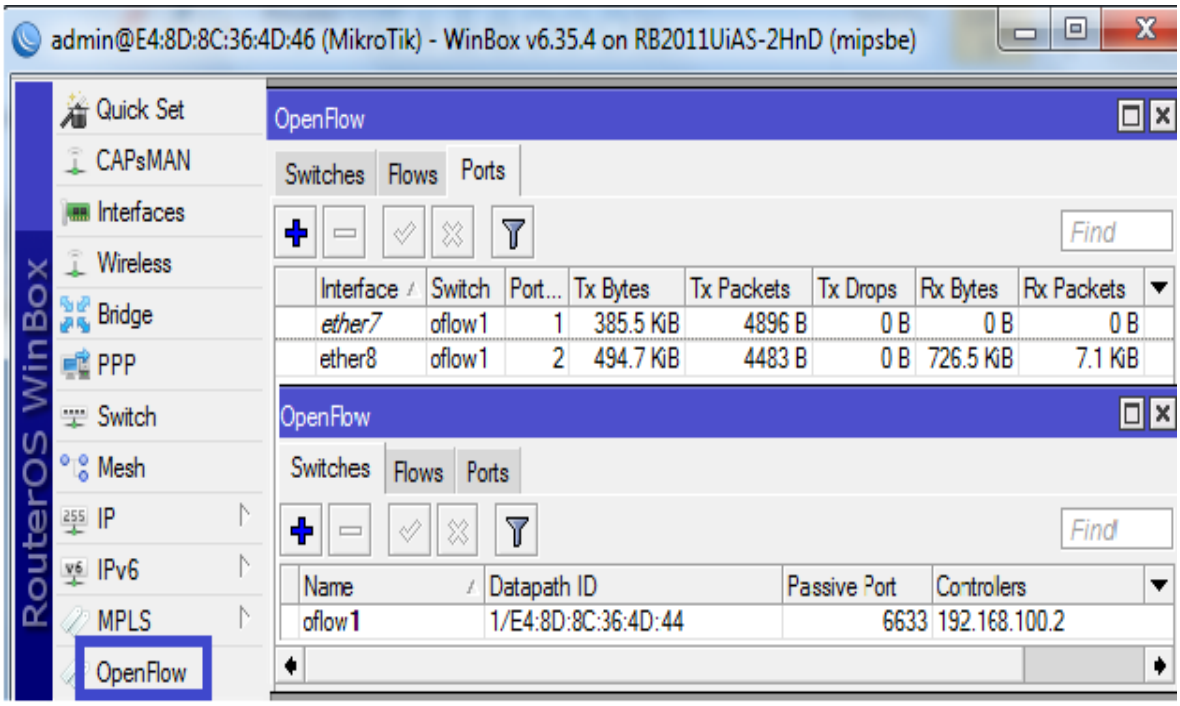

Figura 20: Configuración Switch OpenFlow

En la figura 20, se observa la configuración del canal OpenFlow implementado sobre los puertos OpenFlow del Switch Core RB2011IL-IN (correspondiente a la casa central). Esta configuración es similar en los tres switches, ya que en todos los casos sólo se define la conexión de la interfaz física saliente en el switch contra el puerto lógico OpenFlow 6633 (oflow1) y la dirección del controlador de destino.
Este procedimiento se repite para los otros dos switches con los datos que aparecen en la tabla 4, donde se describen los puertos que conectan a cada switch a través de OpenFlow.

FLOW	MASTER	ASIGNACION	ROUTER	Puerto
1	ETH 6	ETH7,ETH8	CC RB2011IL-IN	6633
1	ETH 3	ETH4,ETH5	SUC-A RB750GL	6633
1	ETH 6	ETH7,ETH8	SUC2 RB2011IL-IN	6633

Tabla 4: Asignación de puertos para los Switches OpenFlow

Después de habilitar la conexión IP y OpenFlow en los switches se procede a habilitar el controlador en la PC que se conecta al puerto seis del Mikrotik RB2011IL-IN, switch core que está configurado para OpenFlow. Inicialmente a través de la consola de Ubuntu, se crea una carpeta con los archivos de la instalación del controlador SDN OpenDayLight, seguidos de Karaf, Restconf, Dlux, para habilitar las API que sirven al controlador, como aparece en la figura 21.

```
Sudo apt-get install Maven
Sudo apt-get install openkdk-8-jre
cd ODL/bin
./Karaf
feature:install odl-restconf
feature:install odl-l2switch-switch
feature:install odl-dlux-all
```

Figura 21: Comandos para iniciar OpenDayLight

Posteriormente se habilitan las funciones gráficas del controlador, para luego validar la conectividad IP y el establecimiento del canal OpenFlow contra el switch core para permitir gestionar los datos de la red a través de OpenDayLight.

2.2 Habilitando el Controlador SDN OpenDayLight

2.2.1 Integración del Switch Mikrotik con ODL

Después de habilitar la función de OpenFlow en los switches Mikrotik, especificando las interfaces y puertos dedicados para este fin, se procede a ejecutar el modo gráfico del controlador SDN OpenDayLight, que está conectado al switch Mikrotik. Esto permite llamar a la funcion Karaf desde el directorio raíz, habilitando las funciones del modo gráfico y la consola DLUX, para acceder a los recursos de Yang, que junto con el API Rest, como aparece en la figura 22.

```
jl@Lab:~$ sudo apt-get install maven
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
maven ya está en su versión más reciente (3.3.9-3).
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 267 no actualizados.
jl@Lab:~$
jl@Lab:~$ cd ODL/bin
jl@Lab:~/ODL/bin$ ./karaf
karaf: JAVA_HOME not set; results may vary
OpenJDK 64-Bit Server VM warning: ignoring option MaxPermSize=512m; support was
removed in 8.0
/home/jl/ODL/data/log/karaf.log (Permiso denegado)
jl@Lab:~/ODL/bin$ sudo ./karaf
karaf: JAVA_HOME not set; results may vary
OpenJDK 64-Bit Server VM warning: ignoring option MaxPermSize=512m; support was
removed in 8.0
```

Figura 22: Ventana de ejecución para Karaf

Para acceder a la consola del lenguaje de modelado Yang, se carga la dirección del controlador en un navegador como aparece en la figura 23, obteniendo acceso a las funcionalidades de Yang UI, disponible con el usuario y pasword admin.

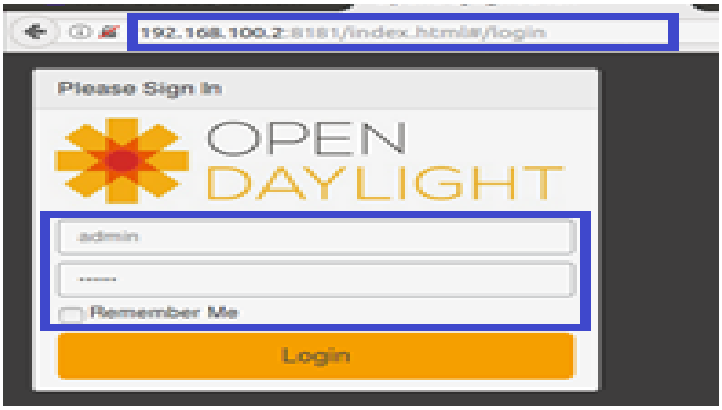

Figura 23: Acceso al hypervisor ODL

Luego de ingresar a la consola gráfica del controlador se despliega la función Topology con lo cual se consigue que el controlador ODL realice la abstracción de la red y genere la gráfica con los switches hablitados para OpenFlow, como aparece en la figura 24.

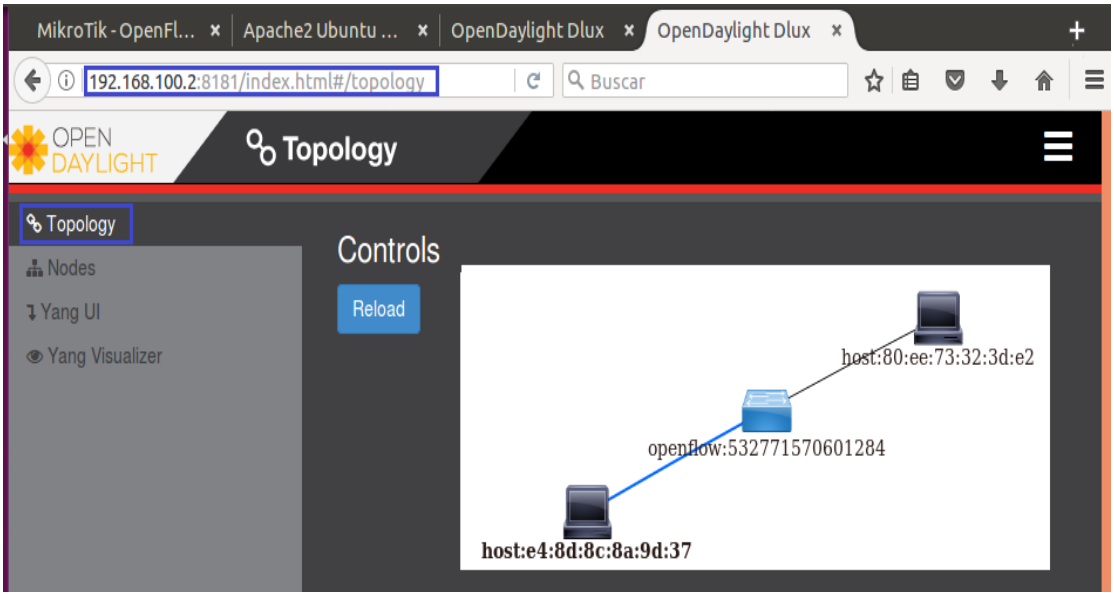

Figura 24: Abstracción de la red con un nodo y dos host

En la abstracción gráfica de la red activa a través de OpenFlow, se observa el switch core Mikrotik RB2011IL-IN, con el Identificador de Nodo 532771570601284 asignado por el controlador SDN, y el host que está conectado al switch de la sucursal “A” con la Mac Address e4:8d:8c:8a:9d:37. En el caso de la sucursal B tomó la Mac Address del host que está conectado al switch de la sucursal B.

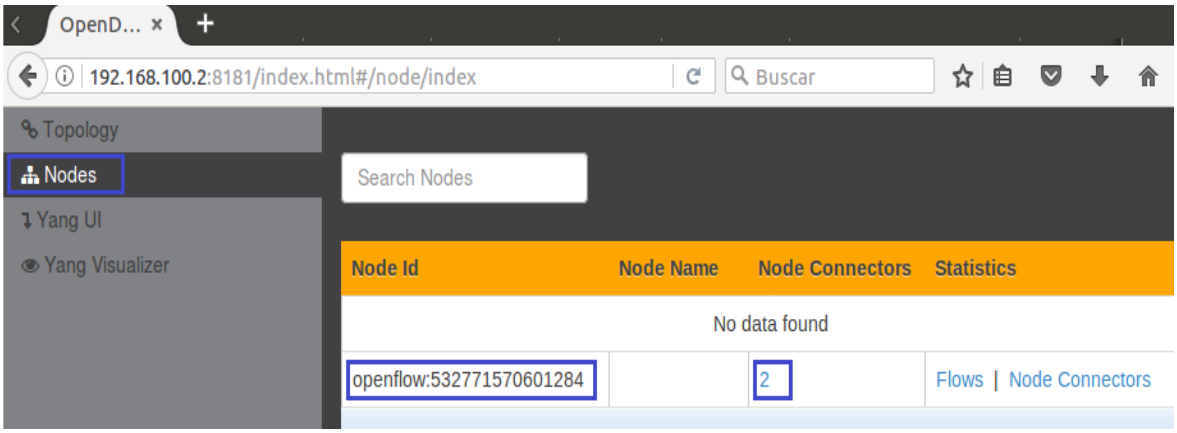

Figura 25: Hypervisor conectividad de los Nodos

Posteriormente en la pestaña Nodes se observa la información de conectividad de los nodos OpenFlow con sus flujos y puertos con MAC Address asignadas.

En la figura 26 se observa la asignación de puertos en el nodo del Switch OpenFlow que tiene conectado a la sucursal A, en el puerto ocho y la sucursal B a través del puerto nueve. Estos datos son abstraídos de la red por ODL.

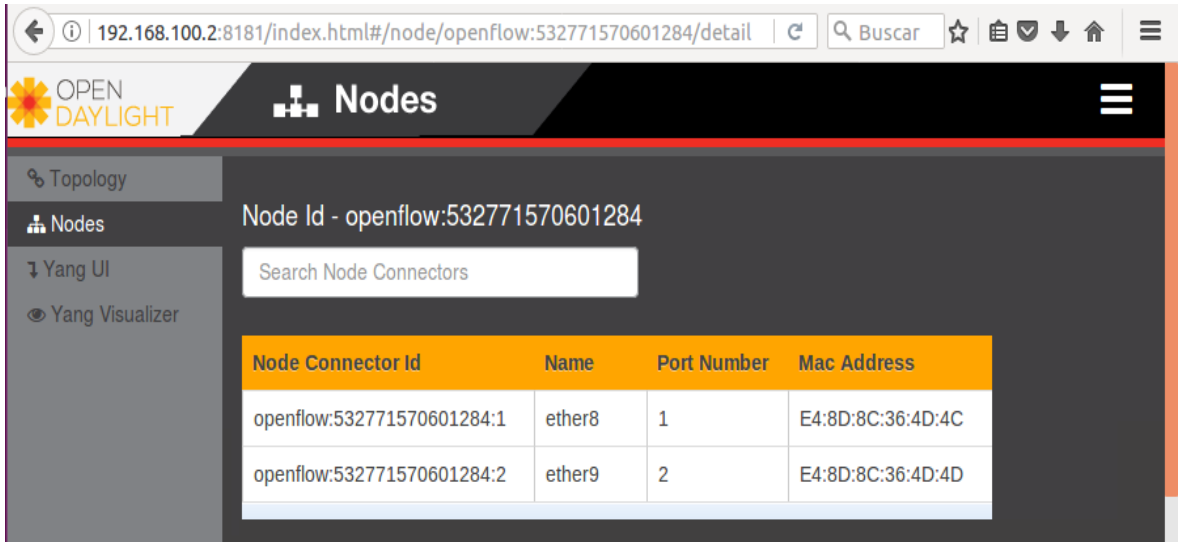

Figura 26: Asignación de puertos en el nodo OpenFlow

Avanzando a la pestaña Flows se puede observar los flujos enviados previamente durante el establecimiento del canal en el switch Mikrotik OpenFlow, donde hay paquetes recibidos y enviados hacia el controlador. En este proceso se observa que hay errores en la recepción de paquetes y frames, los cuales corresponden a los mensajes de metadata enviados y cuya respuesta no se recibió en el controlador porque no está definido el flujo en el Switch Mikrotik [27] como aparece en la figura 27.

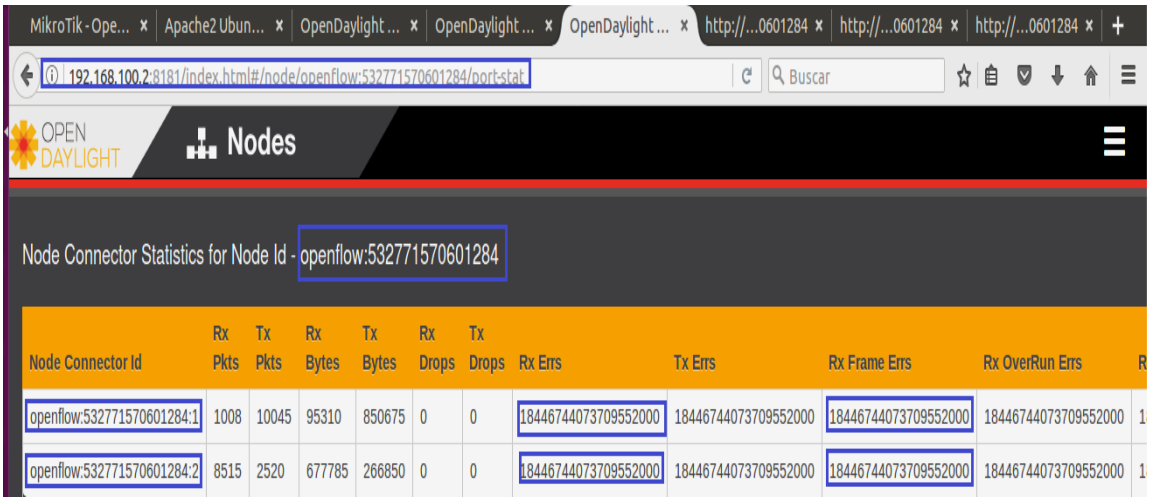

Figura 27: Abstracción de los flujos definidos en el switch OpenFlow

Al habilitar Wireshark se consigue una captura de los flujos de paquetes OpenFlow generados por el controlador OpenDayLight durante el establecimiento de la red. Estos flujos son originados en el controlador con la IP 192.1168.100.2 con destino a la IP 192.168.100.1. Estos paquetes son respuesta del establecimiento del canal y la señalizacion IP necesaria para mantener la conexión.

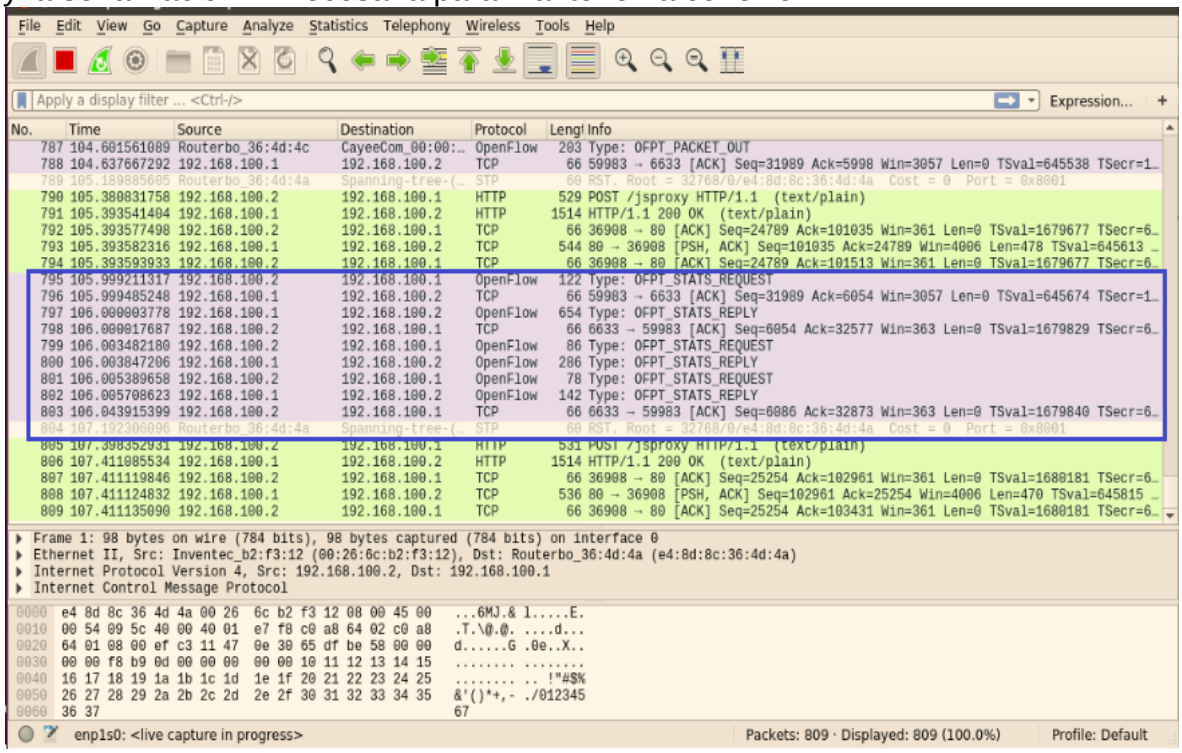

Figura 28: Captura Wireshark para el switch OpenFlow

Del otro extremo, en el switch Mikrotik se puede observar la recepción de los flujos OpenFlow enviados por el controlador en la pestaña de flujos de OpenFlow, como aparece en la figura 29.

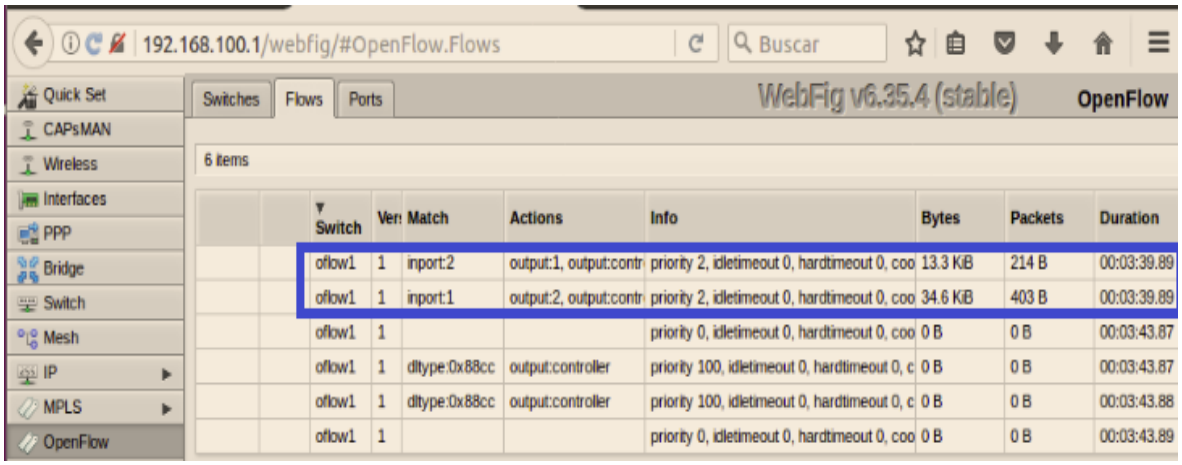

Figura 29: Registro de los flujos recibidos en el Switch OpenFlow

Agregar y cargar los flujos del controlador en el switch se puede hacer a través del protocolo Restconf, editando el archivo de configuracion y cargándolo desde el directorio raíz, como aparece en la figura 30, o simplemente utilizando el modulo Yang UI del controlador OpenDayLight a través del cual se definen los parámetros de la topología que conoce el controlador.

En la figura 30, se observa el despliege del árbol en el modulo Yang, que trae la configuración inicial del nodo OpenFlow con la ruta: **OpenDayLight-inventory:nodes/node/ID_Node**.

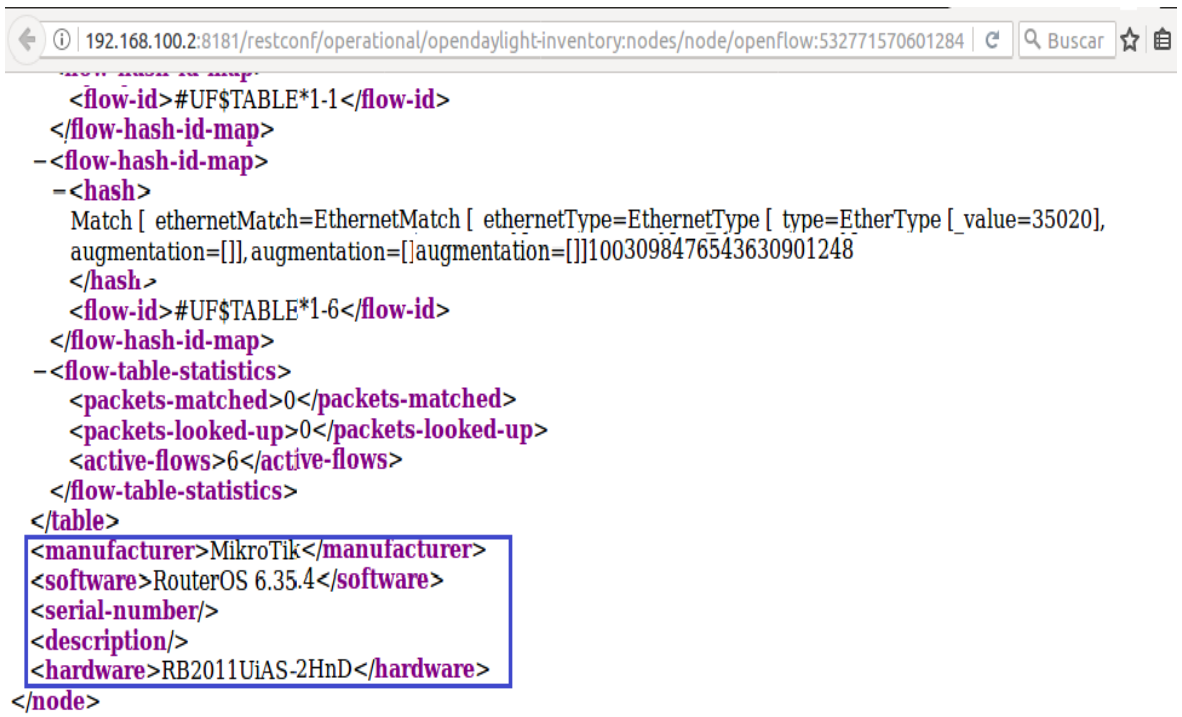

Figura 30: Archivo XML con la Configuración del Nodo OpenFlow

Al reemplazar el ID Node por el identificador de Nodo que trajo OpenDayLight en la pestaña de Topology con la Opción GET, se obtiene la definición inicial de los flujos OpenFlow para el equipo de la red.

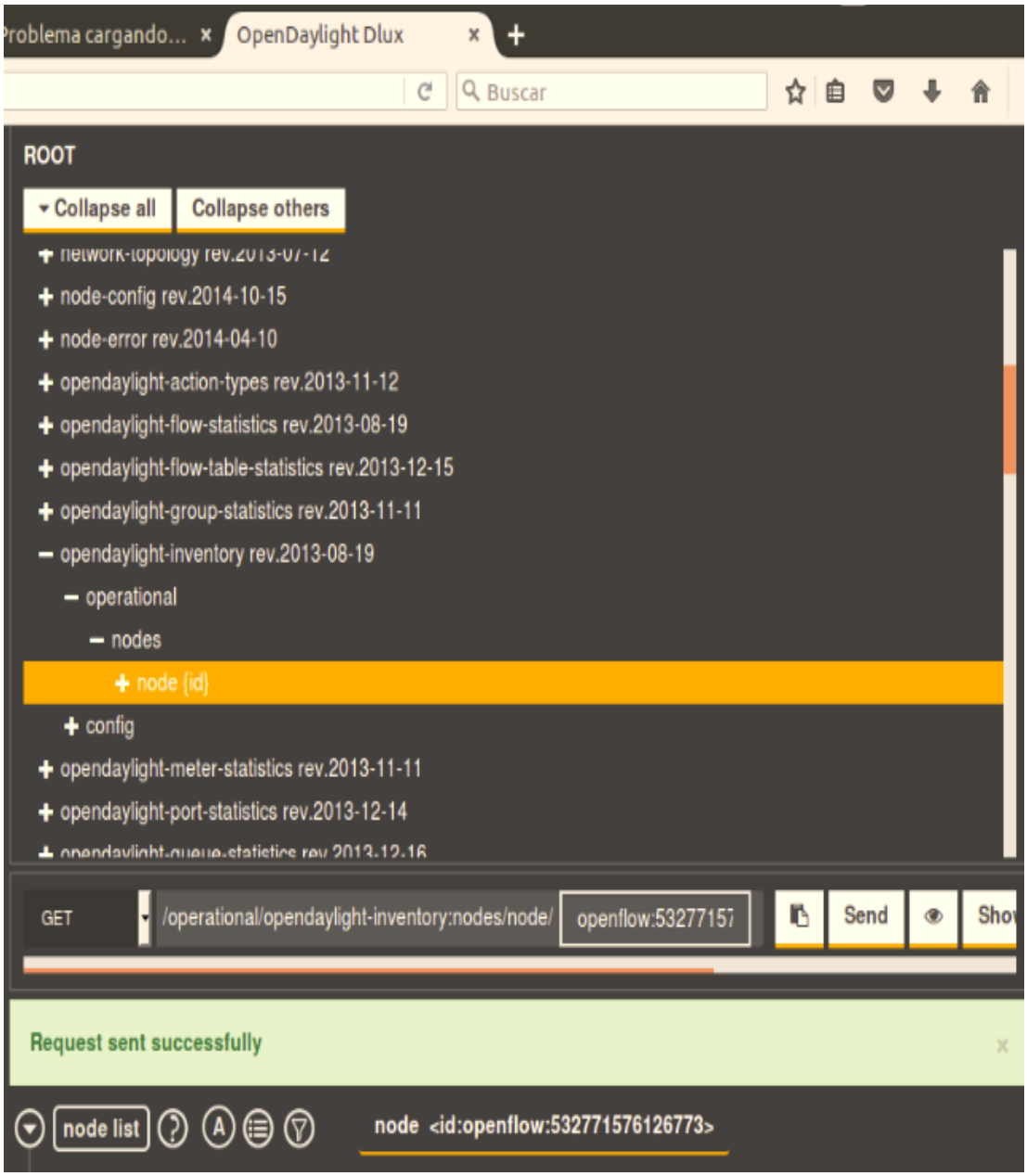

Figura 31: Asignación de los flujos a cada Nodo OpenFlow

Al pulsar la opción de ver en el módulo Yang UI, se genera la ruta a traves del API Rest para consultar el archivo XML a través de un navegador, obteniendo la configuración inicial de los flujos OpenFlow como se observa a continuación en la figura 32.

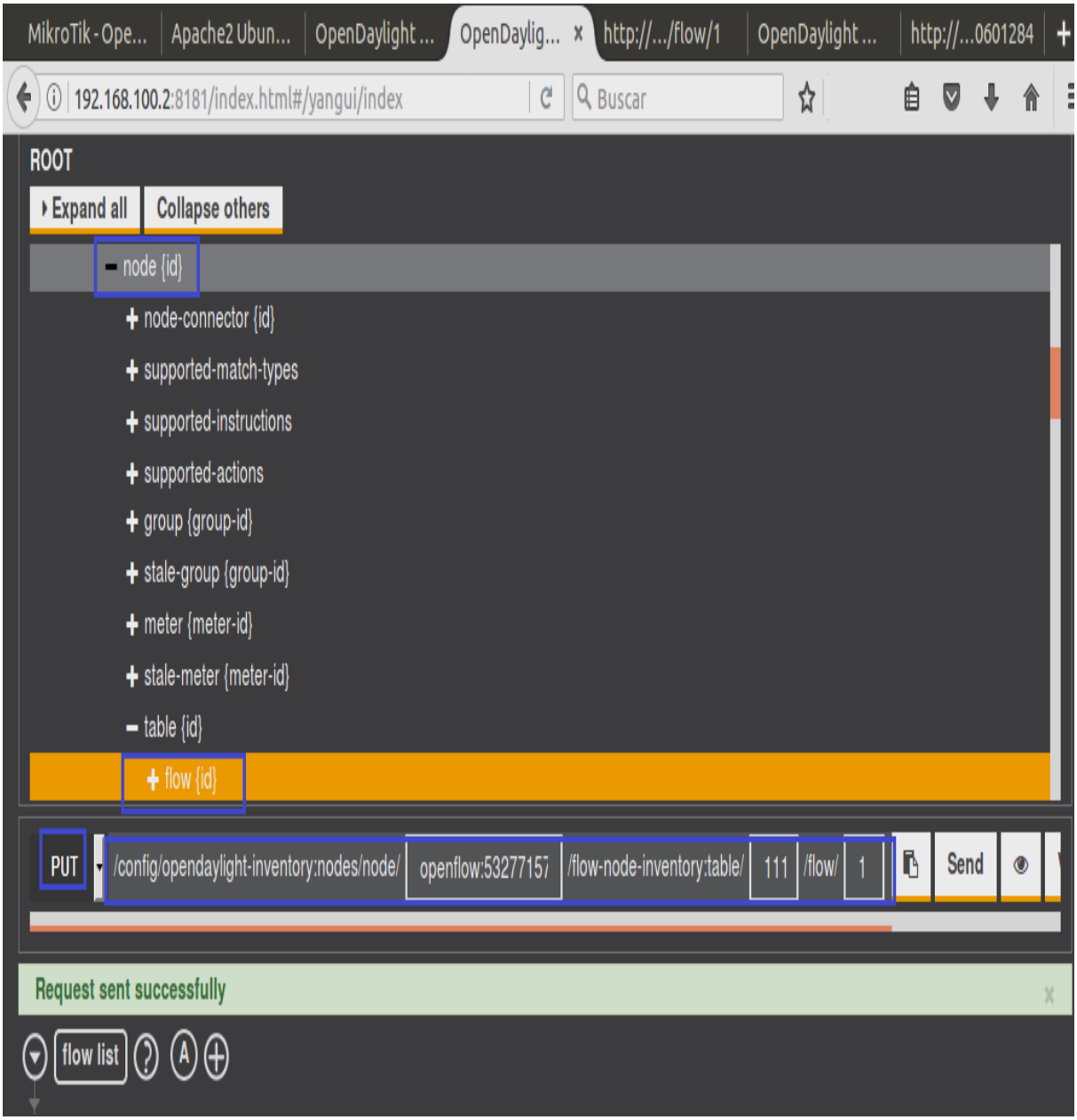

Figura 32: Generación de los flujos activos en el switch Mikrotik.

En la parte inferior del modulo Yang UI se despliegan contenedores, con la opción para adicionar recursos a la configuración activa a traves del API Rest. Específicamente se generan los flujos para permitir o negar el tráfico.

2.3 Gestión de los Flujos por Software

En la pestaña Yang UI se tiene la capacidad de modificar los flujos, definiendo los parámetros para crear o cambiar el flujo, como aparece en la figura 34 y 35, donde se agrega al nodo OpenFlow la tabla 111 con el flujo 1.

Figura 33: Definición del flujo a modificar a traves de la opcion PUT

En los contenedores que se generan se seleccionan las opciones para definir los parámetros a modificar, tales como: IP de origen y destino, MAC Address del puerto de origen, de destino y la opción de ICMP, como aparece en la figura 33.

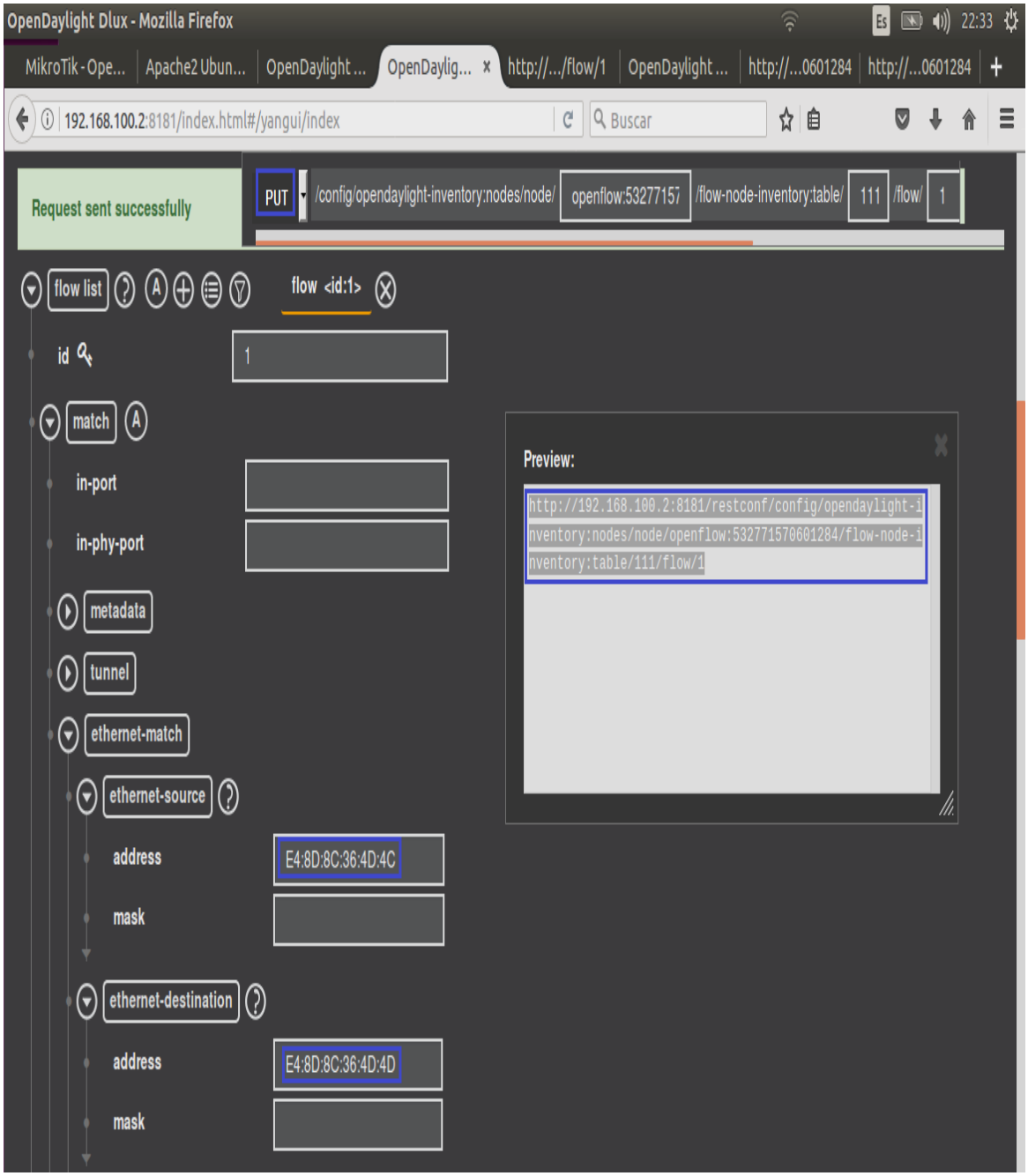

Figura 34: Configuración de los ficheros para modificar los flujos

Después de finalizada la configuración se consulta la ruta para validar el archivo XML con los flujos definidos en OpenFlow como aparece en la opción Preview de la figura 34.

Cargando la dirección obtenida en el preview, en el navegador se accede a los flujos creados en OpenFlow como aparece en la figura 35.

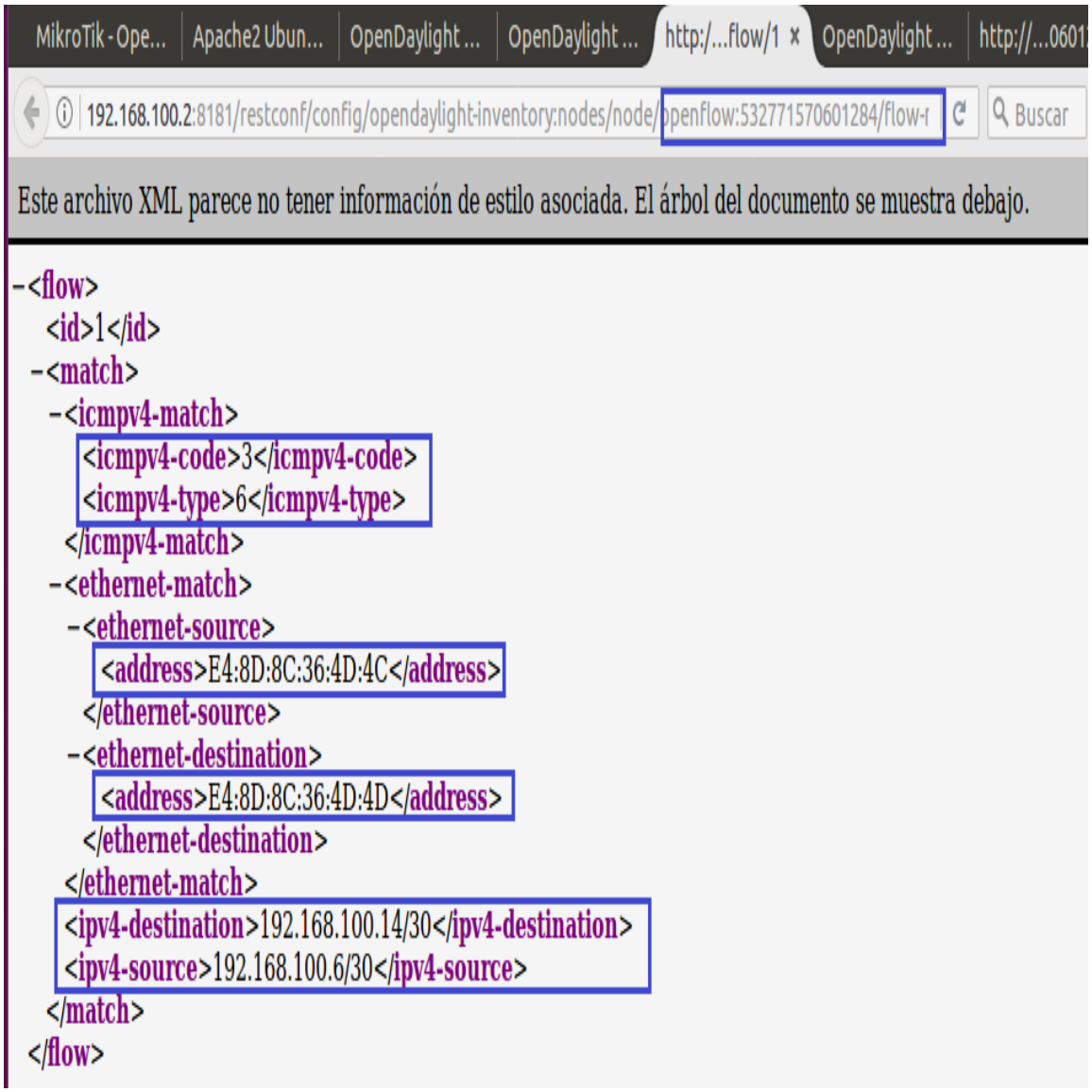

Figura 35: Gráfica con el flujo cargado al controlador

Para consultar el flujo completo y analizar sus variantes se puede obtener, segmentado y en formato de texto exportable, todo el archivo de configuración del Switch OpenFlow a través de la opción History del Modulo Yang UI de OpenDayLight donde se encuentra el historial de las configuraciones realizadas, como aparece en la figura 36.

Los archivos con el texto correspondiente a la definición inicial de los flujos en el switch OpenFlow y la configuración final con los flujos adicionados se encuentran en los anexos del presente documento.

Para tener gestión de los puertos y aplicaciones, OpenDayLight proporciona la opción de integrar el control centralizado y la virtualización de recursos a través de Openstack.

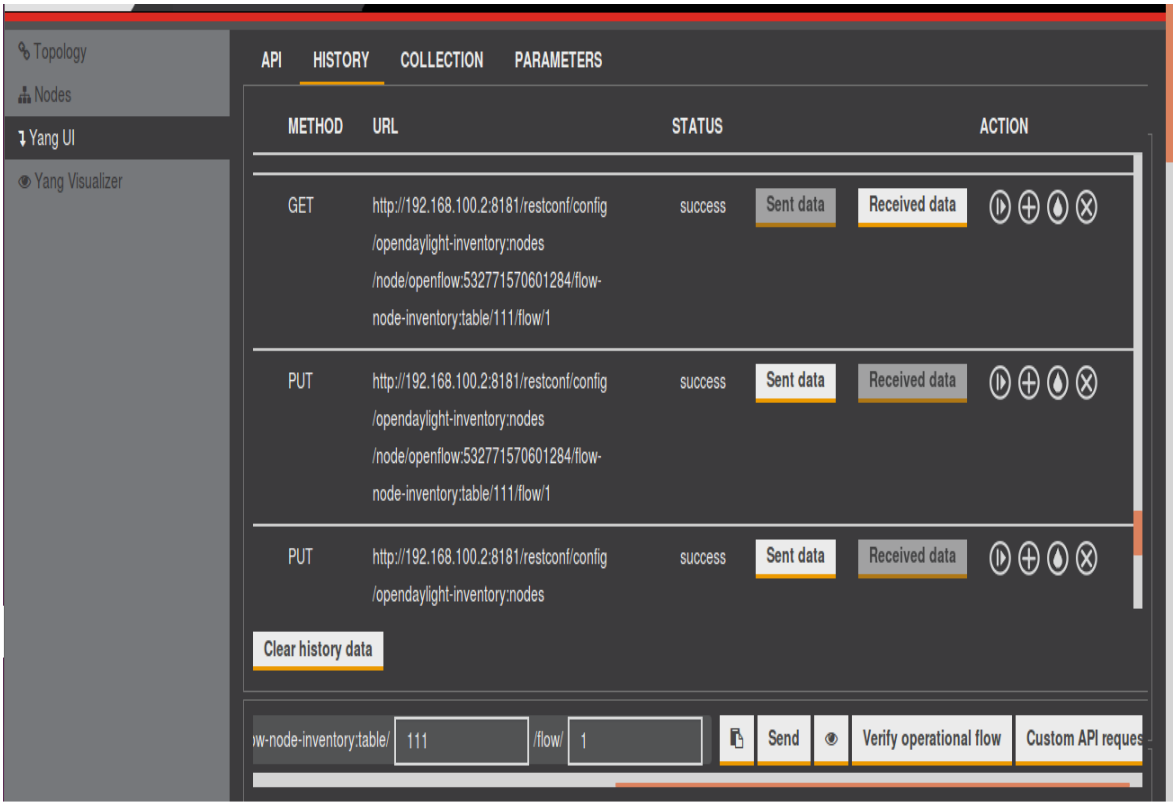

Figura 36: Historial de flujos cargados al controlador

No se realizó esta parte de la implementación ya que el despliegue de Openstack implica disponer de un equipo adicional, con 16 GBytes de memoria de procesamiento para configurar los cuatro principales módulos de Openstack y ésto es otro proyecto adicional.

No es factible conseguir una respuesta ICMP del flujo creado ya que habilitar esta funcionalidad implica habilitar el flujo de datos para este servicio a través del API Restconf programando en Java o Python, como aparece en la referencia [31].

Openstack es un sistema operativo en la nube que permite controlar recursos de computación, almacenamiento y redes a través de una consola y APIs. Su arquitectura es distribuida, utilizando diferentes nodos especializados en diferentes servicios. Openstack está programado en Python, respetando la arquitectura de aplicaciones de interfaces de programación API y Rest (representación del estado de transferencia) [32].

Neutron es un proyecto de Openstack que proporciona interconexión orientada al servicio, entre interfaces, aplicaciones, servicios virtualizados y otros servicios como Nova (Controlador de Openstack) [33]. Neutron interviene en la virtualización de interfaces de ODL en la capa Northbound facilitando sus recursos para administrar las conexiones virtuales.

2.4 Integración OpenDayLight con Openstack

Openstack es un proyecto de computación en la nube que entrega una infraestructura orientada al servicio. Es un software libre, de código abierto distribuido bajo los términos de la licencia Apache. El proyecto está gestionado por la fundación Openstack. de la cual forman parte más de 200 empresas como AMD, Avaya, Brocade, Canonical, Cisco, Dell, Ericsson, HP, IBM, InkTank, Intel, NEC, Red Hat, SUSE, Linux, VMware y Yahoo!

La tecnología es una serie de proyectos relacionados entre sí. Que controlan estanques de control de procesamiento, almacenamiento y recursos de red a través de un centro de datos, todos administrados a través de un panel de control que permite a los administradores controlar y gestionar los recursos de los usuarios a través de una interfaz web [36].

Para la implementación de Openstack son necesarios como mínimo tres nodos en clúster, un nodo de control, que contiene la gestión de todos los servicios (Nova, Neutron, Glance, Switft, Cinder, Keystone), el nodo de cómputo dos corre Nova-Compute, y Neutron utiliza el back-end, y las Vlan por los túneles de OVS. (Open Virtual Switch)

Después de instalar Openstack, se valida el establecimiento de la conexión con Horizon; se revisa la configuración de Neutron creando dos interfaces, con una subred privada en modo bridge contra la red pública y probando que se conecten [37]. En la página de Ubuntu se encuentra una guía más detallada de la implementación de Openstack [38].

Capítulo 3

3.1 RESULTADOS

- En el montaje realizado, se estableció el canal OpenFlow, entre el controlador y el Switch Mikrotik, utilizando el controlador SDN OpenDayLight, para producir la abstracción de la red conectada al controlador y definir los flujos de datos en los puertos del Switch Core.

- La implementación no es completa ya que para producir conmutación física de los flujos de datos activos en el Switch OpenFlow en el montaje propuesto, es necesario el despliegue de Openstack y por limitaciones técnicas no se llevó a cabo ya que no se contaba con la infraestructura necesaria (16 GBytes de Memoria para definir cuatro máquinas virtuales y un Virtual switch para configurar Nova y Neutron).

- Se observó la gestión del plano de control con la habilitación de los flujos definidos a través del API Rest utilizando los recursos de Yang UI, con la definición del flujo para la tabla 111 (como se aprecia en la Figura 32).

- El desarrollo de este proyecto fue una gran oportunidad para descubrir las características de los equipos Mikrotik, comprobar su compatibilidad con el protocolo OpenFlow y su capacidad de interacción con el controlador SDN OpenDayLight.

- OpenDayLight tiene un alto nivel de integración de servicios a través de OpenFlow pero para gestionar virtualmente los flujos del equipo físico es necesario llevarlo a un nivel de virtualización y una herramienta que es complemento y recomienda ODL es Openstack a través de sus librerías Nova y Neutron.

Capítulo 4

4.2 CONCLUSIONES

- A través del módulo Yang en la pestaña Delux se abstrae una gráfica con los equipos activos en la red junto con los flujos de datos de OpenFlow. Esto permite cambiar la definición de los flujos de datos a través de los contenedores presentes en el módulo Yang UI.

- En los Switch Mikrotik la funcionalidad OpenFlow es un recurso que permite conectarse de forma transparente al controlador SDN para gestionar los flujos y tablas de datos.

- Netconf y REST son recursos que permiten el acceso y modificación de los nodos activos en la red siendo un recurso que conserva un repositorio e historial de las configuraciones realizadas.

- Al implementarse en un entorno productivo una forma de garantizar la integridad, seguridad y mayor estabilidad de la operación del canal OpenFlow es habilitando una conexión segura a través de TLS.

- La flexibilidad de los equipos Mikrotik permite actualizar puntos de la red y sumar funcionalidades como OpenFlow de forma física, o a través de router virtuales o soportadas a través de Openflow en el Switch Mikrotik.

Capítulo 5

5.1 RECOMENDACIONES Y FUTUROS TRABAJOS

OpenDayLight es un controlador SDN robusto pero la mejor forma de aprovechar su funcionalidad en un entorno productivo es integrándolo con Openstack.
Contar con recursos en la nube permite desplegar este tipo de soluciones de una forma más óptima facilitando observar la verdaderas funcionalidades de las redes definidas por software que son optimizar la red, centralizar el procesamiento y reducir los costos de implementación, mantenimiento y operación.
Los equipos Mikrotik son una gran alternativa para despliegues rápidos de bajo costo y respuesta estable; pero para experimentación y pruebas es mejor acudir a una solución virtualizada en la nube.

5.2 Players SDN en la Actualidad

Algunos de los jugadores que ofrecen equipos que participan del grupo de desarrollo de OpenDayLight o Linux Fundation son:

- IBM
  https://www.ibm.com/us-en/marketplace/software-defined-networking
- HP SDN
  https://www.hpe.com/us/en/networking/sdn.html
- Alcatel Nuage
- http://www.nuagenetworks.net/
- Cisco Nexus SDN
  http://www.cisco.com/c/es_ar/solutions/software-defined-networking/overview.html
- Brocade SDN
  http://www.brocade.com/en/products-services/software-networking/sdn-controllers-applications/sdn-controller.html

Algunos fabricantes no participan del grupo de desarrollo de Linux Foundation y OpenDayLight, pero cuentan con foros donde la documentación de sus desarrolladores y usuarios son amplios recursos para ajustar sus productos a los requerimientos de la red, es dispendioso y requiere de un perfil de programador.

- Mikrotik SDN
  http://wiki.mikrotik.com/wiki/Manual:API
- FORTINET SDN
  https://www.fortinet.com/products/SDN-security-appliances/forticore.html
- Dell SDN
  http://en.community.dell.com/techcenter/networking/w/wiki/11762.the-dell-networking-Opendaylight-controller-for-openstack-deployments

Capítulo 6

6.1 BIBLIOGRAFÍA

1. MikroTik + OpenFlow = the future / ANDIS ATNIS MUM USA 2014 Mikrotik powered OpenFlow network make a step in future networking [En línea] https://www.owler.com/reports/mikrotik/mikrotik-posted-a-video--mum-usa-2014--network-arc/1449833250825 [Ultimo acceso: 05/ 05/ 2017] https://mum.mikrotik.com//presentations/IT14/arins.pdf [Ultimo acceso: 05/ 05/ 2017]

2. Thomas D. Nadeau y Ken Gray / SDN networks Orrely digital book. [En línea]http://shop.oreilly.com/product/0636920027577/ViewLarger.do?sortby=publicationDate [Ultimo acceso: 05/ 05/ 2017]

3. Software-Defined Networking:Management Requirements and Challenges/ Juliano Araujo Wickboldt, Wanderson Paim de Jesus, Pedro Heleno Isolani, / Comunication Magazine january 2015 pag 282-289

4. Aporte de contribuciones de Código en por controlador /Blog OpenDayLight: Beryllium Release /Autor Rojër Farrë Consultado 23/02/2016 https://blog.rojerfarre.com/2016/02/23/opendaylight-beryllium-released/

5. Plataforma de control para SDN con OpenDayLight OSCP [En línea] https://wiki.Opendaylight.org/view/Opendaylight_SDN_Controller_Platform_(OSCP):Main Ultimo acceso: 05/ 05/ 2017

6. Realizing the Quality of Service (QoS) in Software-Defined Networking (SDN) based Cloud infrastructure. Autores: Govindarajan, Kannan; Meng, Kong Chee; Ong, Hong; / Publicado en: Information and Communication Technology (ICoICT), 2014 2nd International Conference on 2 Oct 2014

7. Definicion SDN Open Networking fundation/ . [En línea] [Ultimo acceso: 05/ 05/ 2017] https://www.opennetworking.org/images/stories/downloads/SDN-resources/solution-briefs/sb-of-enabled-transport-SDN.pdf

8. Definición OpenFlow Open Networking fundation (ONF) [En línea] https://www.opennetworking.org Ultimo acceso: 05/ 05/ 2017

9. Especificación Switch OpenFlow Version 1.1.0 /2011 [En línea] [Ultimo acceso: 05/ 05/ 2017] http://archive. OpenFlow.org/documents/ OpenFlow-spec-v1.1.0.pdf

10. Guía del Usuario instalación y comienzo OpenDayLight [En línea] https://www.Opendaylight.org/sites/Opendaylight/files/bk-user-guide.pdf [Ultimo acceso: 05/ 05/ 2017]

11. Definición OSGI [En línea] https://es.wikipedia.org/wiki/OSGi Ultimo acceso: 05/ 05/ 2017]

12. Defunción Maven [En línea] https://maven.apache.org) [Ultimo acceso: 05/ 05/ 2017]

13. Definición Karaf [En línea] https://karaf.apache.org/manual/latest/ Ultimo [acceso: 05/ 05/ 2017]

14. Definición OpenDayLight Componentes OpenFlow 1.3 [En línea] https://www.opendaylight.org/sites/opendaylight/files/bk-user-guide.pdf [Ultimo acceso: 05/ 05/ 2017]

15. Definición de Apache Karaf https://wenxueliu.gitbooks.io/Opendaylight-karaf/content/karaf_guide/README.html [En línea] [Ultimo acceso: 05/05/2017]

16. API REST características: [En línea] [Ultimo acceso: 05/ 05/ 2017] https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

17. Definición y características de un switch hibrido [En línea] [Ultimo acceso: 05/ 05/ 2017]https://wiki.Opendaylight.org/images/1/1d/ODL_Hybrid_Mode.pdf

18. Definición de DLUX [En línea] [Ultimo acceso: 05/ 05/ 2017] https://www.Opendaylight.org/sites/Opendaylight/files/bk-user-guide.pdf

19. Enrutamiento entre Host [En línea] [Ultimo acceso: 05/ 05/ 2017] https://wiki.Opendaylight.org/view/Opendaylight_Network_Virtualization_(ONV):Main

20. Controlador OSCP redundante [En línea] [Ultimo acceso: 05/ 05/ 2017] https://wiki.Opendaylight.org/view/Opendaylight_SDN_Controller_Platform_(OSCP):Main

21. MDSAL [En línea] http://SDNhub.org/tutorials/Opendaylight/ [Ultimo acceso: 05/ 05/ 2017]

22. YANG [En línea] https://sreeninet.wordpress.com/2014/06/28/netconf-and-yang/ [Ultimo acceso: 05/ 05/ 2017]

23. Mikrotik RB 750GL [En línea] https://routerboard.com/RB750GL [Ultimo acceso: 05/ 05/ 2017]

24. Mikrotik RB2011IL-INIL-HnD-IN[En línea] [Ultimo acceso: 05/ 05/ 2017] https://routerboard.com/RB2011IL-INIL-INUiAS-2HnD-IN

25. Errores de Metadata [En línea]] https://media.readthedocs.org/pdf/ryu-takahashi/ofctl13_expat/ryu-takahashi.pdf [Ultimo acceso: 05/ 05/ 2017]

26. Errores Metadata [En línea] [Ultimo acceso: 05/ 05/ 2017] https://media.readthedocs.org/pdf/ryu-takahashi/ofctl13_expat/ryu-takahashi.pdf

27. Curso virtual de SDN Portal Courcera Dr.Nick Feamster [En línea] https://es.coursera.org/learn/sdn [Ultimo acceso: 05/ 05/ 2017]

28. ONF definición OpenFlow [En línea] [Ultimo acceso: 05/ 05/ 2017] https://www.opennetworking.org/sdn-resources/sdn-definition

29. Implementación Ping OpenDayLight[En línea] [Ultimo acceso: 05/ 05/ 2017] https://wiki.opendaylight.org/view/Ping

30. OSGi Middleware [En línea] [Ultimo acceso: 05/ 05/ 2017] https://www.ntt-review.jp/archive/ntttechnical.php?contents=ntr200801gls.html

31. Micro-servicios [En línea] http://www.arquitecturajava.com/que-es-un-microservicio/ [Ultimo acceso: 05/ 05/ 2017]

32. Ping en OpenDayLight https://wiki.opendaylight.org/view/Ping

33. Definición de Openstack [En Línea] [Ultimo acceso: 07/ 05/ 2017] https://docs.openstack.org/ocata/networking-guide/index.html

34. Definición de Neutron [En Línea] [Ultimo acceso: 07/ 05/ 2017] http://events.linuxfoundation.org/sites/events/files/slides/ODL%20and%20OpenStack%20-%20Workshop_0.pdf

35. Interacción de Neutron con ODL en virtualización [En Línea] [Ultimo acceso: 07/ 05/ 2017] https://wiki.opendaylight.org/view/OpenStack_and_OpenDaylight

36. Openstack. Definición [En Línea] [Ultimo acceso: 08/ 05/ 2017] https://es.wikipedia.org/wiki/OpenStack#Networking_.28Neutron.29

37. Instalación Openstack por ODL [En Línea] [Ultimo acceso: 08/ 05/ 2017] https://drive.google.com/file/d/0B_rLr6so6DZ8bHdtQkk0Rkk2Wms/view

38. Instalación Openstack con Ubuntu [En Línea] [Ultimo acceso: 08/ 05/ 2017] **https://www.ubuntu.com/cloud/openstack**

39. Definición de Netconf Restconf y Yang: [En Línea] [Ultimo acceso: 6/05/2017] https://tools.ietf.org/html/rfc8040

Capítulo 7

7.1 Glosario

ODL: Sigla abreviada de OpenDayLight.

Openflow: Protocolo de comunicaciones que conecta la capa de control con el forwarding de la red.

hypervisor: Monitor de máquina virtual, es una Consola de monitoreo y configuración que permite aplicar diversas técnicas de control, para manejar diferentes sistemas operativos de una misma computadora como sucede con Yang UI en OpenDayLight.

Yang: Lenguaje para describir la estructura básica de algunos datos de aplicaciones que se almacenan en una jerarquía de árbol dentro de los contenedores.

Micro Servicios: Conocido por la sigla MSA (Microservices Architecture), es una aproximación para el desarrollo de software, consiste en construir una aplicación como un conjunto de pequeños servicios, los cuales ejecutan su propio proceso y se comunican con sistemas ligeros, típicamente una API de recursos HTTP.

Meta Router: Es una Instancia, un módulo de virtualización, propio de Mikrotik, que permite crear máquinas virtuales, para correr RouterOS o Openwrt, dentro del mismo router.

RouterOS: Sistema Operativo de los Router Mikrotik

TLS: Sigla de Transport Layer Security, es la evolución de SSL (Capa de puertos seguros), son protocolos criptográficos que proporcionan comunicaciones seguras, por una red de datos o Internet.

NETCONF: Define el conjunto de operaciones que se pueden realizar en la configuración de datastores (Crear, Leer, Actualizar y Borrar) CRUD.

RESTCONF no especifica recursos de operaciones obligatorios, la semántica de cada operación determina como se accede a los datastores. Los datos de configuración se pueden modificar con los comandos: DELETE,PATCH, POST,y PUT. lOS datos se codifican con XML o JSON

7.2 Abstracción del flujo del nodo OpenFlow configurado en el RB2011ILIN


```
{
 “node”: [
 {
 “id”: “openflow:532771570601284”,
 “node-connector”: [
 {
 “id”: “openflow:532771570601284:1”,
 “flow-node-inventory:configuration”: “”,
 “flow-node-inventory:name”: “ether7”,
 “flow-node-inventory:current-feature”: “”,
 “flow-node-inventory:hardware-address”: “E4:8D:8C:36:4D:4B”,
 “flow-node-inventory:peer-features”: “”,
 “flow-node-inventory:advertised-features”: “”,
 “flow-node-inventory:state”: {
 “link-down”: true,
 “blocked”: false,
 “live”: false
 },
 “flow-node-inventory:supported”: “”,
 “flow-node-inventory:port-number”: “1”,
 “opendaylight-port-statistics:flow-capable-node-connector-statistics”: {
 “receive-drops”: 0,
 “transmit-drops”: 0,
 “bytes”: {
 “received”: 0,
 “transmitted”: 32400
 },
 “receive-frame-error”: 18446744073709552000,
 “receive-over-run-error”: 18446744073709552000,
 “collision-count”: 18446744073709552000,
 “duration”: {},
 “packets”: {
 “received”: 0,
 “transmitted”: 216
 },
 “receive-errors”: 18446744073709552000,
 “transmit-errors”: 18446744073709552000,
 “receive-crc-error”: 18446744073709552000
 }
 },
 {
 “id”: “openflow:532771570601284:2”,
 “flow-node-inventory:configuration”: “”,
 “flow-node-inventory:name”: “ether8”,
```

“flow-node-inventory:current-feature”: “”,
“flow-node-inventory:hardware-address”: “E4:8D:8C:36:4D:4C”,
“flow-node-inventory:peer-features”: “”,
“flow-node-inventory:advertised-features”: “”,
“flow-node-inventory:state”: {
“link-down”: false,
“blocked”: false,
“live”: false
},
“flow-node-inventory:supported”: “”,
“flow-node-inventory:port-number”: “2”,
“opendaylight-port-statistics:flow-capable-node-connector-statistics”: {
“receive-drops”: 0,
“transmit-drops”: 0,
“bytes”: {
“received”: 40500,
“transmitted”: 97971
},
“receive-frame-error”: 18446744073709552000,
“receive-over-run-error”: 18446744073709552000,
“collision-count”: 18446744073709552000,
“duration”: {},
“packets”: {
“received”: 270,
“transmitted”: 867
},
“receive-errors”: 18446744073709552000,
“transmit-errors”: 18446744073709552000,
“receive-crc-error”: 18446744073709552000
}
}
],
“flow-node-inventory:manufacturer”: “MikroTik”,
“flow-node-inventory:software”: “RouterOS 6.35.4”,
“flow-node-inventory:ip-address”: “192.168.100.1”,
“flow-node-inventory:serial-number”: “”,
“flow-node-inventory:description”: “”,
“flow-node-inventory:switch-features”: {
“capabilities”: [
“flow-node-inventory:flow-feature-capability-table-stats”,
“flow-node-inventory:flow-feature-capability-flow-stats”,
“flow-node-inventory:flow-feature-capability-port-stats”
],
“max_buffers”: 0,
“max_tables”: 1
},

“flow-node-inventory:table”: [
{
“id”: 0
},
{
“id”: 1,
“flow”: [
{
“id”: “#UF$TABLE*1-6”,
“match”: {
“thernet-match”: {
“thernet-type”: {
“type”: 35020
}
}
},
“hard-timeout”: 0,
“idle-timeout”: 0,
“priority”: 100,
“table_id”: 1,
“opendaylight-flow-statistics:flow-statistics”: {
“packet-count”: 0,
“byte-count”: 0,
“duration”: {
“nanosecond”: 650000000,
“second”: 4327
}
},
“cookie”: 3098476543630901000,
“instructions”: {
“instruction”: [
{
“order”: 0,
“apply-actions”: {
“action”: [
{
“order”: 0,
“output-action”: {
“output-node-connector”: “CONTROLLER”,
“max-length”: 65535
}
}
]
}
}
]

```
}
},
{
“id”: “#UF$TABLE*1-5”,
“match”: {
“thernet-match”: {
“thernet-type”: {
“type”: 35020
}
}
},
“hard-timeout”: 0,
“idle-timeout”: 0,
“priority”: 100,
“table_id”: 1,
“opendaylight-flow-statistics:flow-statistics”: {
“packet-count”: 0,
“byte-count”: 0,
“duration”: {
“nanosecond”: 670000000,
“second”: 4327
}
},
“cookie”: 3098476543630901000,
“instructions”: {
“instruction”: [
{
“order”: 0,
“apply-actions”: {
“action”: [
{
“order”: 0,
“output-action”: {
“output-node-connector”: “CONTROLLER”,
“max-length”: 65535
}
}
]
}
}
]
}
},
{
“id”: “#UF$TABLE*1-4”,
“match”: {
```

“in-port”: “openflow:532771570601284:2”
},
“hard-timeout”: 0,
“idle-timeout”: 0,
“priority”: 2,
“table_id”: 1,
“opendaylight-flow-statistics:flow-statistics”: {
“packet-count”: 216,
“byte-count”: 32400,
“duration”: {
“nanosecond”: 900000000,
“second”: 4323
}
},
“cookie”: 3098476543630901000,
“instructions”: {
“instruction”: [
{
“order”: 0,
“apply-actions”: {
“action”: [
{
“order”: 1,
“output-action”: {
“output-node-connector”: “CONTROLLER”,
“max-length”: 65535
}
},
{
“order”: 0,
“output-action”: {
“output-node-connector”: “1”,
“max-length”: 65535
}
}
]
}
}
]
}
},
{
“id”: “#UF$TABLE*1-3”,
“match”: {
“in-port”: “openflow:532771570601284:1”
},

“hard-timeout”: 0,
“idle-timeout”: 0,
“priority”: 2,
“table_id”: 1,
“opendaylight-flow-statistics:flow-statistics”: {
“packet-count”: 0,
“byte-count”: 0,
“duration”: {
“nanosecond”: 900000000,
“second”: 4323
}
},
“cookie”: 3098476543630901000,
“instructions”: {
“instruction”: [
{
“order”: 0,
“apply-actions”: {
“action”: [
{
“order”: 1,
“output-action”: {
“output-node-connector”: “CONTROLLER”,
“max-length”: 65535
}
},
{
“order”: 0,
“output-action”: {
“output-node-connector”: “2”,
“max-length”: 65535
}
}
]
}
}
]
}
},
{
“id”: “#UF$TABLE*1-2”,
“match”: {},
“hard-timeout”: 0,
“idle-timeout”: 0,
“priority”: 0,
“table_id”: 1,

“opendaylight-flow-statistics:flow-statistics”: {
“packet-count”: 0,
“byte-count”: 0,
“duration”: {
“nanosecond”: 670000000,
“second”: 4327
}
},
“cookie”: 3098476543630901000
},
{
“id”: “#UF$TABLE*1-1”,
“match”: {},
“hard-timeout”: 0,
“idle-timeout”: 0,
“priority”: 0,
“table_id”: 1,
“opendaylight-flow-statistics:flow-statistics”: {
“packet-count”: 0,
“byte-count”: 0,
“duration”: {
“nanosecond”: 670000000,
“second”: 4327
}
},
“cookie”: 3098476543630901000
}
],
“flow-hash-id-map”: [
{
“hash”: “Match [_inPort=Uri [_value=openflow:532771570601284:1],
augmentation=[]]23098476543630901248”,
“flow-id”: “#UF$TABLE*1-3”
},
{
“hash”: “Match [augmentation=[]]03098476543630901249”,
“flow-id”: “#UF$TABLE*1-2”
},
{
“hash”: “Match [_ethernetMatch=EthernetMatch [_ethernetType=EthernetType
[_type=EtherType [_value=35020], augmentation=[]], augmentation=[]],
augmentation=[]]1003098476543630901249”,
“flow-id”: “#UF$TABLE*1-6”
},
{
“hash”: “Match [_inPort=Uri [_value=openflow:532771570601284:2],

augmentation=[]]23098476543630901249”,
“flow-id”: “#UF$TABLE*1-4”
},
{
“hash”: “Match [augmentation=[]]03098476543630901248”,
“flow-id”: “#UF$TABLE*1-1”
},
{
“hash”: “Match [_ethernetMatch=EthernetMatch [_ethernetType=EthernetType
[_type=EtherType [_value=35020], augmentation=[]], augmentation=[]],
augmentation=[]]100309847654363090124 8”,
“flow-id”: “#UF$TABLE*1-5”
}
],
“opendaylight-flow-table-statistics:flow-table-statistics”: {
“packets-matched”: 0,
“packets-looked-up”: 0,
“active-flows”: 6
}
}
],
“flow-node-inventory:hardware”: “RB2011IL-INIL-INUiAS-2HnD”
}
]
}

7.3 Flujo que Permitir el Tráfico Entre los Puertos del Controlador


```
{
 "node": [
 {
 "id": "openflow:532771570601284",
 "node-connector": [
 {
 "id": "openflow:532771570601284:1",
 "flow-node-inventory:configuration":
"",
 "flow-node-inventory:name": "ether7",
 "flow-node-inventory:current-feature":
"",
 "flow-node-inventory:hardware-
address": "E4:8D:8C:36:4D:4B",
 "flow-node-inventory:peer-features":
"",
 "flow-node-inventory:advertised-
features": "",
 "flow-node-inventory:state": {
 "link-down": true,
 "blocked": false,
 "live": false
 },
 "flow-node-inventory:supported": "",
 "flow-node-inventory:port-number":
"1",
 "opendaylight-port-statistics:flow-
capable-node-connector-statistics": {
 "receive-drops": 0,
 "transmit-drops": 0,
 "bytes": {
 "received": 0,
 "transmitted": 32400
 },
 "receive-frame-error":
18446744073709552000,
 "receive-over-run-error":
18446744073709552000,
 "collision-count":
18446744073709552000,
 "duration": {},
 "packets": {
 "received": 0,
 "transmitted": 216
 },
 "receive-errors":
18446744073709552000,
 "transmit-errors":
18446744073709552000,
 "receive-crc-error":
18446744073709552000
 }
 },
 {
 "id": "openflow:532771570601284:2",
 "flow-node-inventory:configuration":
"",
 "flow-node-inventory:name": "ether8",
 "flow-node-inventory:current-feature":
"",
 "flow-node-inventory:hardware-
address": "E4:8D:8C:36:4D:4C",
 "flow-node-inventory:peer-features":
"",
 "flow-node-inventory:advertised-
features": "",
 "flow-node-inventory:state": {
 "link-down": false,
 "blocked": false,
 "live": false
 },
 "flow-node-inventory:supported": "",
 "flow-node-inventory:port-number":
"2",
 "opendaylight-port-statistics:flow-
capable-node-connector-statistics": {
 "receive-drops": 0,
 "transmit-drops": 0,
 "bytes": {
 "received": 40500,
 "transmitted": 97971
 },
 "receive-frame-error":
18446744073709552000,
 "receive-over-run-error":
18446744073709552000,
 "collision-count":
18446744073709552000,
```

```
 "duration": {},
 "packets": {
 "received": 270,
 "transmitted": 867
 },
 "receive-errors":
1844674407370955200 0,
 "transmit-errors":
1844674407370955200 0,
 "receive-crc-error":
1844674407370955200 0
 }
 }
 ],
 "flow-node-inventory:manufacturer":
"MikroTik",
 "flow-node-inventory:software":
"RouterOS 6.35.4",
 "flow-node-inventory:ip-address":
"192.168.100.1",
 "flow-node-inventory:serial-number":
"",
 "flow-node-inventory:description": "",
 "flow-node-inventory:switch-
features": {
 "capabilities": [
 "flow-node-inventory:flow-feature-
capability-table-stats",
 "flow-node-inventory:flow-feature-
capability-flow-stats",
 "flow-node-inventory:flow-feature-
capability-port-stats"
 ],
 "max_buffers": 0,
 "max_tables": 1
 },
 "flow-node-inventory:table": [
 {
 "id": 0
 },
 {
 "id": 1,
 "flow": [
 {
 "id": "#UF$TABLE*1-6",
 "match": {
 "ethernet-match": {
 "ethernet-type": {
 "type": 35020
 }
 }
 },
 "hard-timeout": 0,
 "idle-timeout": 0,
 "priority": 100,
 "table_id": 1,
 "opendaylight-flow-statistics:flow-
statistics": {
 "packet-count": 0,
 "byte-count": 0,
 "duration": {
 "nanosecond": 650000000,
 "second": 4327
 }
 },
 "cookie": 3098476543630901000,
 "instructions": {
 "instruction": [
 {
 "order": 0,
 "apply-actions": {
 "action": [
 {
 "order": 0,
 "output-action": {
 "output-node-connector":
"CONTROLLER",
 "max-length": 65535
 }
 }
 ]
 }
 }
 ]
 }
 },
 {
 "id": "#UF$TABLE*1-5",
 "match": {
 "ethernet-match": {
 "ethernet-type": {
 "type": 35020
```

```
 }
 }
 },
 "hard-timeout": 0,
 "idle-timeout": 0,
 "priority": 100,
 "table_id": 1,
 "opendaylight-flow-statistics:flow-
statistics": {
 "packet-count": 0,
 "byte-count": 0,
 "duration": {
 "nanosecond": 670000000,
 "second": 4327
 }
 },
 "cookie": 3098476543630901000,
 "instructions": {
 "instruction": [
 {
 "order": 0,
 "apply-actions": {
 "action": [
 {
 "order": 0,
 "output-action": {
 "output-node-connector":
"CONTROLLER",
 "max-length": 65535
 }
 }
 ]
 }
 }
 ]
 }
 },
 {
 "id": "#UF$TABLE*1-4",
 "match": {
 "in-port":
"openflow:532771570601284:2"
 },
 "hard-timeout": 0,
 "idle-timeout": 0,
 "priority": 2,
 "table_id": 1,
 "opendaylight-flow-statistics:flow-
statistics": {
 "packet-count": 216,
 "byte-count": 32400,
 "duration": {
 "nanosecond": 900000000,
 "second": 4323
 }
 },
 "cookie": 3098476543630901000,
 "instructions": {
 "instruction": [
 {
 "order": 0,
 "apply-actions": {
 "action": [
 {
 "order": 1,
 "output-action": {
 "output-node-connector":
"CONTROLLER",
 "max-length": 65535
 }
 },
 {
 "order": 0,
 "output-action": {
 "output-node-connector": "1",
 "max-length": 65535
 }
 }
 ]
 }
 }
 ]
 }
 },
 {
 "id": "#UF$TABLE*1-3",
 "match": {
 "in-port":
"openflow:532771570601284:1"
 },
 "hard-timeout": 0,
 "idle-timeout": 0,
```

```
 "priority": 2,
 "table_id": 1,
 "opendaylight-flow-statistics:flow-
statistics": {
 "packet-count": 0,
 "byte-count": 0,
 "duration": {
 "nanosecond": 900000000,
 "second": 4323
 }
 },
 "cookie": 3098476543630901000,
 "instructions": {
 "instruction": [
 {
 "order": 0,
 "apply-actions": {
 "action": [
 {
 "order": 1,
 "output-action": {
 "output-node-connector":
"CONTROLLER",
 "max-length": 65535
 }
 },
 {
 "order": 0,
 "output-action": {
 "output-node-connector": "2",
 "max-length": 65535
 }
 }
 ]
 }
 }
 ]
 }
 },
 {
 "id": "#UF$TABLE*1-2",
 "match": {},
 "hard-timeout": 0,
 "idle-timeout": 0,
 "priority": 0,
 "table_id": 1,
 "opendaylight-flow-statistics:flow-
statistics": {
 "packet-count": 0,
 "byte-count": 0,
 "duration": {
 "nanosecond": 670000000,
 "second": 4327
 }
 },
 "cookie": 3098476543630901000
 },
 {
 "id": "#UF$TABLE*1-1",
 "match": {},
 "hard-timeout": 0,
 "idle-timeout": 0,
 "priority": 0,
 "table_id": 1,
 "opendaylight-flow-statistics:flow-
statistics": {
 "packet-count": 0,
 "byte-count": 0,
 "duration": {
 "nanosecond": 670000000,
 "second": 4327
 }
 },
 "cookie": 3098476543630901000
 }
 ],
 "flow-hash-id-map": [
 {
 "hash": "Match [_inPort=Uri
[_value=openflow:532771570601284:
1],
augmentation=[]]23098476543630901
248",
 "flow-id": "#UF$TABLE*1-3"
 },
 {
 "hash": "Match
[augmentation=[]]0309847654363090
1249",
 "flow-id": "#UF$TABLE*1-2"
 },
 {
```

```
 "hash": "Match
[_ethernetMatch=EthernetMatch
[_ethernetType=EthernetType
[_type=EtherType [_value=35020],
augmentation=[]], augmentation=[]],
augmentation=[]]10030984765436309
01249",
 "flow-id": "#UF$TABLE*1-6"
 },
 {
 "hash": "Match [_inPort=Uri
[_value=openflow:532771570601284:
2],
augmentation=[]]23098476543630901
249",
 "flow-id": "#UF$TABLE*1-4"
 },
 {
 "hash": "Match
[augmentation=[]]0309847654363090
1248",
 "flow-id": "#UF$TABLE*1-1"
 },
 {
 "hash": "Match
[_ethernetMatch=EthernetMatch
[_ethernetType=EthernetType
[_type=EtherType [_value=35020],
augmentation=[]], augmentation=[]],
augmentation=[]]10030984765436309
01248",
 "flow-id": "#UF$TABLE*1-5"
 }
 ],
 "opendaylight-flow-table-
statistics:flow-table-statistics": {
 "packets-matched": 0,
 "packets-looked-up": 0,
 "active-flows": 6
 }
 }
 ],
 "flow-node-inventory:hardware":
"RB2011IL-INIL-INUiAS-2HnD"
 }
 ]
}
```