

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

MAESTRÍA DE REDES DE COMUNICACIÓN

Plan de Proyecto Tesis

Estudio del protocolo Openflow usando el modelo de red definida por Software (Software Define Networks). Caso de estudio la Universidad Técnica de Manabí.

Alumno

Wilson Ignacio Intriago Romero

Quito - 2017

Director de Tesis

Dr. Gustavo Chafla Altamirano

AUTORÍA

AUTORÍA Yo, Wilson Intriago Ignacio Romero, portador de la cédula de ciudadanía No.1309039244, declaro bajo juramento que la presente investigación es de total responsabilidad del autor, y que se he respetado las diferentes fuentes de información realizando las citas correspondientes. Esta investigación no contiene plagio alguno y es resultado de un trabajo serio desarrollado en su totalidad por mi persona.

DEDICATORIA

"***Un sueño no se hace realidad a través de magia: conlleva sudor, determinación y Trabajo duro" Colin Powell"***

A cada una de las personas que creyeron en mí y que siempre me daban palabras de aliento para seguir adelante, a aquellos que esperaban que lograra terminar mi TESIS y que pudiera dar un paso importante en mi vida. Les dedico la culminación de este proyecto a mis padres, a mis hermanas familiares y amigos que me dieron apoyo moral siempre e incondicionalmente.

AGRADECIMIENTO

*"**La frase más humilde que puedo decir es que "sin DIOS no soy nada". Pero la más Poderosa es que "con Él lo puedo todo**"*

La vida es un gran camino en el que vamos descubriendo cada día cosas nuevas, es como si fuese el libro con más posibilidades que existe porque sentimos que podemos escribir en ella cualquier posibilidad, ya que la vida está llena de posibilidades. Si entiende que puedes escribir tu propio libro de la vida, pues hazlo con mucho esfuerzo y dedicación para que tengas un bello futuro. Siempre darle gracias a la vida por lo que te da, te dio y te dará.

Agradezco a Dios y a todos mis Santos de devoción por ayudarme a concluir con éxito mi TESIS, a mi Padre que desde el cielo me cuida, ilumina y protege, a mi madre por su fortaleza, sacrificio, enseñanzas, comprensión y amor y por quien lucho por darle lo mejor, a mis hermanas, mi familia, al Director de mi trabajo de Tesis Dr. Gustavo Chafla Altamirano y a los lectores Ing. Juan Francisco y Damián Nicolalde, a mis amigos por su amistad, consejos, ayuda confianza, apoyo y motivación. Por siempre gracias a todos.

ÍNDICE CONTENIDOS

DEDICATORIA ..... **III**

AGRADECIMIENTO ..... **IV**

ÍNDICE CONTENIDOS ..... **V**

ÍNDICES DE FIGURA ..... **VII**

RESUMEN ..... **IX**

SUMMARY ..... **XI**

ANTECEDENTES **4**

JUSTIFICACIÓN **7**

OBJETIVOS **9**

Objetivo General 9
Objetivos específicos 9
CAPÍTULO II 10
2. Marco teórico 10
2.1. Redes definidas en software 10
2.2. Openflow 12
2.3. Implementación y aplicación de OpenFlow 14
2.4. OpenFlow Switch 15
2.5. Tablas de OpenFlow 18
2.5.1. Tabla de Flujos 18
2.6. Tabla de Grupo 20
2.6.1. Estructura de una Entrada de Grupo (Identificador de grupo, Tipo de grupo, Contadores y Cubos de acción). 20
2.6.2. Canal OpenFlow 22
2.6.3. Visión general de Protocolo OpenFlow: 22
2.6.4. Mensajes Controlador a Switch 22
CAPÍTULO III 27
3. Metodología 27
3.1 Hipótesis 27
3.1.1 Variables *27*
3.2 Metodología 27
3.2.1 Técnica de recolección de datos 27
3.3 Herramientas a utilizar en el proyecto 30
3.3.1 Virtual Box 30
3.3.2 VMware Workstation 31
3.3.3 SSH Secure Shell Client 32
3.3.3.1 Características de SSH 32
3.3.4 Xterm 33
3.3.4 Dpctl 34
3.3.5 Wireshark 35
3.3.6 Iperf3 37
3.3.7 Mininet 38

3.3.7.1 ¿Por qué Mininet es genial? 39
3.3.7.1.2 Cuáles son las limitaciones de Mininet? 40
3.3.8 Cbench 42
CAPÍTULO IV 43
4. DESARROLLO DE LA PROPUESTA 43
4.1 Escenario y Pruebas iniciales 43
4.1.1 SDN con OpenDaylight 43
4.2.2 OpenFlow y SDN 47
4.3 Escenario practico de OPENDAYLIGHT con el protocolo Openflow 52
4.3.1 Escenario practico de OPENDAYLIGHT diagrama autoridades 52
4.3.2 Escenario practico de OPENDAYLIGHT con topologías 57
4.5 Diagrama de redes de comunicaciones de la UTM 61
CAPÍTULO V 62
5. CONCLUSIONES Y RECOMENDACIONES 62
5.2 Recomendaciones 63
BIBLIOGRAFÍA 64

ÍNDICES DE FIGURA

Figura 1: Switch OpenFlow dedicado. 16

Figura 2: Estructura de un Switch OpenFlow 17

Figura 3. Flujo de paquetes del procesamiento Pipeline 20

Figura 4, Comando Xterm 34

Figura 5: Topología en OPENDAYLIGHT 43

Figura 6: Construcción del proyecto OpenDaylight. 43

Figura #7: Construcción del proyecto. 44

Figura 8: Uso de Mininet 44

Figura 9: Uso de Mininet 46

Figura 10: Regla conmutador en mininet. 46

Figura 11: emular 1 switch con 3 hosts. 47

Figura 12: Ejecución de Wireshark 47

Figura 13: Prueba de conectividad mininet 48

Figura 14: Listado de flujos de openflow 48

Figura 15: Negociado del protocolo OPENFLOW. 49

Figura 16: Mensaje OFPT_FEATURES_REQUEST. 49

Figura 17: Mensaje OFPT_FEATURES_REQUEST. 50

Figura 18: Configuración Conjunto 50

Figura 19 Solicitud de varias respuestas 51

Figura 20: Modos de flujos 51

Figura 21: Acceso a OPENDAYLIGHT 52

Figura 22: Creación de 4 switch 52

Figura 23: Agregar Links switch con sus hosts 52

Figura 24: Grafico de la red con 4 switch en OPENDAYLIGHT 53

Figura 25: Verificar conexión con Pingall 53

Figura 26: Grafico final con OPENDAYLIGHT 54

Figura 27: Tabla de flujos en el switch4 54

Figura 28: Creación 10 host topología single. 55

Figura 29: Topología de red de Autoridades FCI 56

Figura 30: Mostramos la salida de las interfaces 56

Figura 31: Creamos la topología linear 57

Figura 32: Topología linear opendaylight mininet 57

Figura 33: Topología Tree opendaylight mininet 58

Figura 34: Diagrama de Topología Tree opendaylight mininet 59

Figura 35: Topología Tree opendaylight mininet 59

Figura 36: Diagrama de Topología Tree opendaylight con switches 60

Figura 37: Diagrama de Topología Tree opendaylight con hosts 60

Figura 38: Diagrama de redes de comunicaciones UTM 61

RESUMEN

Las redes de comunicaciones han tenido un crecimiento exponencial de tráfico que circula por la red y cada día a un ritmo mayor, lo cual demanda de métodos de comunicación más eficiente. Se propone una solución que permita cambiar la forma de comunicación de las redes, en la cual se le proporcione mayor inteligencia a la misma es así que nace la arquitectura de redes definidas por software.

Esta herramienta ha revolucionado las comunicaciones con el control de dispositivos desde un software exterior, con la ayuda del protocolo OpenFlow lo que le permite la programabilidad de las redes sin necesita que se exponga la estructura interna de los equipos de networking para esto se han desarrollado herramientas para su experimentación para esta tesis se utiliza el emulador Mininet y el software OpenDaylight para la creación de los escenarios prácticos.

Teniendo en cuenta que esta arquitectura SDN en nuestro país es reciente en esta investigación de la tecnología del protocolo OpenFlow, se ha buscado información sobre sus características, funcionalidad, arquitectura, plataformas de desarrollo, para esta tesis se desarrollaron escenarios prácticos utilizando Mininet y OpenDaylight para la prueba de esta herramienta.

El efecto de la virtualización de servidores es que los flujos de tráfico difieren sustancialmente del modelo tradicional de cliente-servidor. Por lo general, existe una cantidad considerable de tráfico entre los servidores virtuales, para fines tales como el mantenimiento de imágenes consistentes de la base de datos y la invocación de funciones de seguridad como el control de acceso. Estos flujos de servidor a servidor cambian en ubicación e intensidad a lo largo del tiempo, lo que exige un enfoque flexible para administrar los recursos de la red.

Mediante la realización de este proyecto se pretende una descripción de SDN, teniendo como protagonista al protocolo OpenFlow y su aplicación mediante el diseño del escenario virtual sobre la herramienta Mininet y OpenDaylight que

soporte este protocolo, en la red académica de la Universidad Técnica de Manabí el cual permitirá realizar un diseño de la red definida por software permitiendo mejorar la administración y el tráfico de la red

Al Realizar un proyecto de esta magnitud será de gran aporte y nos dará una visión diferente al momento de construir redes, porque con las redes definidas por software SDN openflow, ya no es necesarios el hardware sino el software el encargado de controlar la red obteniendo de esta manera que la evolución de las redes se realice a la misma velocidad que se desarrolla el software logrando la reutilización de código para dar solución a una alternativa abaratando costos de implementación y no estar limitado a utilizar equipos de un mismo fabricante.

SUMMARY

Communications networks have had an exponential growth of traffic that circulates through the network and every day at a higher rate, which demands more efficient communication methods. A solution is proposed that allows to change the form of communication of the networks, in which greater intelligence is provided to the same is how the architecture of networks defined by software is born.

This tool has revolutionized the communications with the control of devices from an external software, with the help of the OpenFlow protocol which allows the programmability of the networks without needing to expose the internal structure of the networking equipment for this tools have been developed for his experimentation for this thesis the Mininet emulator and the OpenDaylight software are used to create the practical scenarios.

Taking into account that this SDN architecture in our country is recent in this investigation of OpenFlow protocol technology, we have searched for information about its characteristics, functionality, architecture, development platforms, for this thesis practical scenarios were developed using Mininet and OpenDaylight for the test of this tool.

The effect of server virtualization is that traffic flows differ substantially from the traditional client-server model. In general, there is a considerable amount of traffic between the virtual servers, for purposes such as the maintenance of consistent images of the database and the invocation of security functions such as access control. These server-to-server flows change in location and intensity over time, requiring a flexible approach to managing network resources.

Through the realization of this project a description of SDN is sought, having as protagonist the OpenFlow protocol and its application through the design of the virtual scenario on the tool Mininet and OpenDaylight that supports this protocol,

in the academic network of the Technical University of Manabí which will allow a design of the network defined by software allowing to improve the administration and traffic of the network.

When carrying out a project of this magnitude, it will be a great contribution and will give us a different vision when building networks, because with the networks defined by SDN Openflow software, the hardware is no longer needed but the software is in charge of controlling the network, obtaining from This way that the evolution of the networks is carried out at the same speed as the software is developed, achieving the reuse of code to solve an alternative by reducing implementation costs and not being limited to using equipment from the same manufacturer.

INTRODUCCIÓN

El OpenFlow es un protocolo que constituye la base de las redes abiertas definidas por software (Software-Defined Networking, SDN) basadas en estándares. OpenFlow empezó a desarrollarse en los sectores académico y empresarial para centralizar las decisiones de paquetes permitiendo a los usuarios crear políticas y encontrar rutas con ancho de banda disponible, baja latencia o congestión y menos saltos entre switches y routers de diferentes fabricantes. Fueron las universidades de Stanford y California en Berkeley quienes llevaron las riendas en primera instancia.

OpenFlow es un estándar abierto que puede ser implementado en conmutadores Ethernet, routers y puntos de acceso inalámbrico (AP). En el marco OpenFlow, el reenvío de paquetes (plano de datos) y las decisiones de encaminamiento (plano de control) se ejecutan en diferentes dispositivos. Interruptores OpenFlow que se encargan del reenvío de paquetes, mientras que los controladores configuran la tabla de reenvío del interruptor en función de cada flujo, para permitir el aislamiento de flujo y de recursos. Se considera la ruta de datos y el análisis de la implementación OpenFlow en PCs basados en Linux. Se compara la conmutación OpenFlow, de capa dos de conmutación Ethernet y de capa tres el rendimiento de enrutamiento IP. Se analiza el Reenvío de rendimiento y la latencia de paquetes en condiciones subcargados y sobrecargados, con diferentes patrones de tráfico. Se analiza la escalabilidad del sistema utilizando diferentes tamaños de tabla de reenvío, y se mide la equidad en la distribución de recursos.

El nacimiento de las aplicaciones en tiempo real, el video streaming; la masificación de las redes sociales, la introducción del cloud computing y muchos otros

Servicios, han dado como resultado el crecimiento exponencial del tráfico que circula por la red. A pesar de ello se continúa utilizando las mismas tecnologías que hace cincuenta años y los avances en cuanto a nuevas formas de comunicación y tratamiento de la información son casi inexistentes. Este presente proyecto permite realizar un estudio del estado del arte de las redes basada en software que mediante el desarrollo de aplicaciones permitan definir políticas [01] (Cuella, 2014).

Las Redes Definidas por Software o SDN (Software Defined Network por sus siglas en inglés) constituyen una tecnología en la que se desacopla el plano de datos del plano de control y en la que se emplean controladores, que son los responsables de gestionar la información de reenvío de los conmutadores o switch en redes cableadas. Hoy en día se han desarrollado muchos controladores SDN, tanto de código abierto como comerciales, por lo que uno de los principales aspectos a tener en cuenta en el entorno actual de las Redes Definidas por Software. Las SDN también permiten simplificar los dispositivos de red, pues ya estos no tienen que entender y procesar varios protocolos estándares, sino simplemente aceptar instrucciones de los controladores SDN.

Es así que nace el concepto de Software Defined Networking, el cual está revolucionando el campo de las comunicaciones mediante el control de los dispositivos de la red desde un software exterior, con la ayuda del protocolo OpenFlow creado especialmente para este propósito. OpenFlow propone nuevas características que permiten la programabilidad de las redes y la experimentación sin la necesidad de que los fabricantes de equipos de networking expongan la estructura interna de sus productos; basta con que se adicione compatibilidad con el estándar en los equipos de cada casa fabricante. De igual forma se han desarrollado herramientas para experimentar con SDN, tal es el caso del emulador Mininet y VNX. [02] (Barona, 2013)

SDN permite la gestión del plano de control utilizando software y un protocolo denominado OpenFlow, con el cual es posible administrar este plano mediante equipos servidores denominados controladores, mientras que el plano de datos sigue siendo manejado por los equipos de conectividad. La red universitaria carece de una buena administración y el presente estudio permitirá elaborar una propuesta para mejorar el tráfico de la red y ofrecer un excelente servicio a la comunidad universitaria.

Mediante la realización de este proyecto se pretende una descripción de SDN, teniendo como protagonista al protocolo OpenFlow y su aplicación mediante el diseño del escenario virtual sobre la herramienta VNX que soporte este protocolo, en la red académica de la Universidad Técnica de Manabí el cual permitirá realizar un diseño de la red definida por software permitiendo mejorar la administración y el tráfico de la red.

ANTECEDENTES

Actualmente las tendencias que mayor demanda tienen y motivan a la industria del networking a volver a examinar y estructurar las arquitecturas de red tradicionales se constituyen por servicios que se han convertido en actividades de uso cotidiano como el envío de correo electrónico, el comercio electrónico, las tecnologías de búsqueda, las redes sociales, los servicios móviles y los servicios de Cloud Computing, video conferencia (UNESCO, 2004).

Las redes evolucionan constantemente con la finalidad de trasmitir más de un servicio por el mismo medio como en la actualidad las redes convergentes son más dinámicas y difíciles de administrar. SDN permite la gestión y control utilizando software y protocolo denominado OpenFlow, el mismo que permite centralizar las decisiones de migración de paquetes de modo que la red se puede programar independiente de la conmutadora y equipo, con la finalidad de mejorar los servicios de comunicación se realizó un estudio del protocolo openflow en la Universidad Técnica de Manabí.

Esta tecnología ha permitido la interconexión de redes académicas y de investigación que utilizan dispositivos propietarios al interior de la red y como equipos de borde tarjetas NetFPGA. [03] (Blandón Gómez, 2013).

Proyectos exitosos en el mundo de SDN Nokia

Nokia ofrece seguridad a los transportistas SDN, lo que permite despliegue comercial a gran escala de servicios de red basados en SDN, donde brinde las capacidades de automatización y control de Nokia SDN que permitan a los operadores hacer los procesos de aseguramiento mucho más eficiente e impactantes.

Hoy en día, los operadores deben asegurarse de que los servicios de red bajo demanda que traen al mercado se entregan utilizando el aprovisionamiento automatizado que hace el mejor uso de los activos disponibles mediante el control en tiempo real de la red. Sin embargo, a medida que los proyectos de SDN pasan de ensayos en vivo limitados a un amplio despliegue y comercialización, los proveedores

están descubriendo que necesitan seguridad además de la automatización con optimización en línea. [29] (NOKIA, 2016)

Intel y América Móvil se unen para optimizar redes (SDN)

América Móvil ha seleccionado a Intel para llevar a cabo el proyecto de transformación de su red hacia Funciones de Red Virtualizada (NFV) y Redes Definidas por Software (SDN). Con esto, se busca optimizar costos de operación de red y simplificar el despliegue de nuevos servicios en las operaciones de América Móvil en América Latina y Europa.

Las Redes Definidas por Software permiten separar la capa de control de datos, habilitan un control centralizado y ofrecen la capacidad de programar y automatizar el comportamiento de la red, dando como resultado un gobierno completo de las redes.

De esta forma, SDN consigue optimizar el uso de los recursos de red, aumentar su agilidad y flexibilidad, permitiendo la innovación y el dinamismo, mientras que se reducen gastos operativos y de capital. [30] (Intel América, 2015).

Alcatel, China Mobile y China Unicom invierten en banda ancha

Alcatel-Lucent, China Mobile y China Unicom firmaron un acuerdo de colaboración por valor de 1,176 millones de euros para proporcionar acceso a banda ancha de mayor rapidez y para migrar gradualmente hacia redes dinámicas basadas en la nube.

Alcatel-Lucent apoyará a ambas compañías en la entrega de conexiones fijas y móviles a internet de mayor rapidez y más accesibles en todo el país, para ayudar a China a

responder a la creciente necesidad de conectividad de personas y cosas", informó la empresa franco-estadounidense.

Los contratos, de un año de duración cada uno, ascienden a 728 millones de dólares con China Mobile y a 577millones de dólares con China Unicom.

Alcatel-Lucent suministrará sistemas de acceso de banda ultra ancha fijo y móvil, enrutamiento IP, redes ópticas ágiles y capacidades de Virtualización de Funciones de Red (NFV), así como las tecnologías de Redes Definidas por Software (SDN) de Nuage Networks. [31] (Alcatel, 2015)

Cisco con SDN

El controlador SDN abierto Cisco es la distribución comercial de Cisco del controlador de red definida por software abierto OpenDaylight (SDN).

El controlador SDN abierto Cisco es una distribución comercial de OpenDaylight que ofrece la agilidad del negocio mediante la automatización de la infraestructura de red basada en estándares.

Construido como una plataforma de redes SDN altamente escalable, el Open SDN Controller abstrae la complejidad de la gestión de redes heterogéneas para mejorar la prestación de servicios y reducir los costos operativos. Como un software basado en código abierto, el Open SDN Controller avanza continuamente a través de la innovación continua de Cisco y el soporte de la comunidad OpenDaylight. [32] (Cisco, 2015).

JUSTIFICACIÓN

El desarrollo de este proyecto es importante debido a que permite mejorar el rendimiento de las redes actuales, además se analizaran nuevos conceptos, los cuales aportan al entendimiento de un nuevo tipo de arquitectura. Con el estudio de redes definidas por software se podrá solucionar las limitaciones que hoy en día presentan las redes de datos, como la necesidad de configurar los dispositivos de la red de manera individual, la dependencia de os fabricantes soporte para nuevos servicios

Las empresas principalmente son las más beneficiadas por que manejan grandes cantidades de información proveedores de servicios de telecomunicaciones, entidades bancarias entre otros, el potencial que ofrecen las redes definidas por software en cuanto al manejo de tráfico se refiere es de gran envergadura.

El desarrollo de este proyecto será de gran utilidad por lo que las redes definidas por software SDN openflow, presentan grandes ventajas, y una de las relevantes es la seguridad en comparación con una red normal que se debe configurar de forma manual cada dispositivo de la misma siendo esta una tarea tediosa en la cual se podrían cometer errores en las cuales las redes pueden ser atacadas, mientras que en las redes definidas por software al poseer un controlador encargado de administrar toda la red la seguridad será mayo debido a que se centraliza en un solo punto.

Realizar un proyecto de esta magnitud será de gran aporte y nos dará una visión diferente al momento de construir redes, ya que con las redes definidas por software SDN openflow, ya no es necesarios el hardware sino el software el encargado de controlar la red obteniendo de esta manera que la evolución de las redes se realice a la misma velocidad que se desarrolla el software logrando la reutilización de código para dar solución a una

alternativa abaratando costos de implementación y no estar limitado a utilizar equipos de un mismo fabricante.

OBJETIVOS

Objetivo General

Presentar el estudio del protocolo Openflow usando el modelo de red definida por Software (Software Define Networks). Caso de estudio la red académica de la Universidad Técnica de Manabí.

Objetivos específicos

- Investigar como el protocolo Openflow puede contribuir al mejoramiento de la redes de comunicaciones de la Universidad Técnica de Manabí.
- Estudiar la nueva arquitectura de redes asistidas por software, entre los dispositivos de red y el software que los controla.
- Presentar los principales conceptos de la tecnología en base a la revisión del estado del arte de SDN y la herramienta Openflow.
- Diseñar escenarios prácticos usando herramientas SDN y Openflow

CAPÍTULO II

2. Marco teórico

2.1. Redes definidas en software

En la actualidad, la especificación de mayor popularidad para la creación de una red definida por software, es un estándar abierto denominado OpenFlow, que permite a los administradores de red controlar de forma remota tablas de enrutamiento, el protocolo OpenFlow se constituye en la base de redes abiertas definidas por software (Software-Defined Networking, SDN) que están basadas en estándares. Empezó a desarrollarse en el 2007, es el resultado de la estrecha colaboración de sectores académicos y empresariales. Las universidades de California y Stanford en Berkeley quienes llevaron las iniciativas en primera instancia. [6] (Figuerola, 2013).

Las redes definidas por software (SDN) son una manera de abordar la creación de redes en la cual el control se desprende del hardware y se le da el mismo a una aplicación de software llamada controlador. El término SDN (Software Defined Network o red definida por software) se ha venido acuñando en los últimos dos años para hacer referencia a una arquitectura de red que permite separar el plano del control, del plano de datos, para conseguir redes más programables, automatizables y flexibles. Con SDN se virtualiza la red independizándola de la infraestructura física subyacente.

La tecnología SDN elimina la inteligencia de las redes tradicionales del hardware, delegando las capacidades de toma de decisiones en el servidor, la tecnología desacopla la capa de datos (data layer: información real que viaja en una red) de la capa de control (control layer: la tecnología que determina cómo, cuándo y dónde la información flujos), y cada capa está automatizada.

Esto significa que el control independiente de cada capa es posible sin el costoso y complicado hardware y software (por ejemplo, routers y switches) o la necesidad de profesionales altamente calificados para gestionar dichos dispositivos. [7] Andres, (2014).

En una red definida por software, un administrador de red puede darle forma al tráfico desde una consola de control centralizada sin tener que tocar conmutadores individuales, el administrador puede cambiar cualquier regla de los conmutadores de red cuando sea necesario, dando o quitando prioridad, e incluso bloqueando tipos específicos de paquetes con nivel de control muy detallado.

Eso es especialmente útil en una arquitectura de múltiples inquilino (multi-tenant architecture) de computación en nube, porque permite al administrador manejar cargas de tráfico de manera flexible y más eficiente, esencialmente, permite al administrador usar menos conmutadores pequeños y costosos, también tener mayor control sobre el flujo del tráfico de red.

De igual manera flexibilidad, con la respectiva facilidad de gestión y programación son algunas de las ventajas atribuidas a las redes definidas por software, en contraposición a las redes utilizadas en la actualidad, lo cierto es que, con la explosión de los data centers, el auge del cloud junto con las mayores demandas de ancho de banda y conectividad, se ha incrementado la complejidad de la red.

La vertiginosa evolucion tecnologica implica la utilizacion de una serie de sistemas, dentro de las cuales se cita: redes definidas por hardware [8] (Fundación Telefónica, 2015).

Por el crecimiento de ancho de banda, si antes una red tenía cuatro nodos ahora puede tener 4.000. Si esta red de 4.000 nodos requiere 4.000 sistemas de configuración, el grado de complejidad es muy grande. Todos los fabricantes vieron que era necesario

buscar nuevas estrategias para que la programación no fuera nodo a nodo, sino de una manera centralizada mediante software.

Con SDN se virtualiza la red independizándola de la infraestructura física subyacente y creando redes lógicas con objeto de cumplir los requisitos de rendimiento, escalabilidad y agilidad necesarios en modelos de cloud computing. Permite aprovisionar de forma programática las redes virtuales, asociadas a cargas de trabajo de forma dinámica y situadas, movidas o escaladas en cualquier lugar del centro de datos o incluso entre centros de datos geográficamente dispersos. Entonces, cuando el switch recibe el mensaje Ping Reply, el switch ya tiene la regla para envía el paquete. [9] (Velásquez, 2015).

2.2. Openflow

La idea base es práctica, se pone a explotar el hecho de que mucho de los routers y switches Ethernet contienen tablas de flujo (construidas típicamente de TCAMs), mismos que corren en línea para la implementación de firewalls, QoS (Quality of Service), NATs, también para la recolección de estadísticas; se ha identificado un interesante conjunto de funciones que por medio de switches y routers pueden hacerse, siendo el principal objetivo de OpenFlow explotar dichas funcionalidades comunes. De tal manera que permita facilitar el manejo de aplicaciones reduciendo al máximo problemas de compatibilidad de arquitecturas y de emulación de sistemas operativos de otras plataformas. [10] (Pantoja, 2013)

La virtualización tiene un activo rol, al compararse los múltiples sistemas abiertos utilizando los switches de diferentes lugares se puede apreciar que suele presentarse un gap (hueco) entre los atributos existentes y aquellos deseados de nuestra parte para estas tecnologías, ante la necesidad de abarcar una mayor cantidad de posibles propiedades, de allí que el protocolo novedoso OpenFlow sea necesario. [11] (OPENFLOW, 2011).

El OpenFlow consta de forma básica de un protocolo abierto que permite programar la tabla de flujo en diferentes routers y switches, además del administrador de red que permite dividir el tráfico en flujos de investigación y producción, los investigadores tienden a controlar sus propios flujos porque escogen las rutas que los paquetes siguen, así como el procesamiento que reciben; en este orden de ideas, los investigadores prueban y experimentan con nuevos modelos de seguridad, nuevos protocolos, esquemas de direccionamiento y también alternativas a la tradicional IP; en tanto, en la misma red es aislado el tráfico de producción y se procesa de forma tradicional y estándar.

SDN proporciona dos elementos de mucha importancia, APIs que permite a los usuarios de la nube construir y gestionar sus redes virtuales en los data centers, y el protocolo Openflow, que puede ser utilizado para la provisión de las tablas de enrutamiento y la funcionalidad básica de conmutación que utilizan los hypervisores mediante los switchers que permiten la conexión del tráfico entre máquinas virtuales e interfaces físicas [12] (Plaza, 2015).

El proceso de un paquete que entra al conmutador es el siguiente:

• Los campos de cabecera de los paquetes son extraídos y usados para buscar coincidencias.

• Se compara con las reglas definidas para cada entrada en la tabla de flujos OpenFlow, teniendo en cuenta el orden descendente de prioridad. Un paquete puede ser coincidente con una entrada particular de la tabla de flujos usando uno o más campos del paquete. Un campo en la tabla de flujos puede tener el valor ANY o cualquiera, que coincidirá con todos los paquetes.

• Si hay una coincidencia, las acciones especificadas en esa entrada se realizan sobre el paquete. Si no, los primeros 200 bytes del paquete son enviados al controlador para su procesado.

2.3. Implementación y aplicación de OpenFlow

Al ser un concepto nuevo y sin estándares claros, siguen apareciendo nuevas implementaciones constantemente, tanto en el mercado como en el campo de la investigación. Se puede diferenciar según:

Hardware. Su principal ventaja es el mayor rendimiento respecto a la implementación software. Hay que tener en cuenta que el plano de datos es definido según el fabricante. Una aplicación puede funcionar en un conmutador y no en otro, ya que no todos implementan las mismas características de OpenFlow.

Software. Son un conjunto de técnicas relacionadas con el área de redes computacionales. Los dos proyectos más importantes son: NOX, el primer controlador OpenFlow y Mininet, un entorno virtual que se detallará más adelante. Existen controladores software como ProgrammableFlow Controller de NEC, Big Network Controller de Big Switch o proyectos "open source" como OMNI, Trema, Ryu, FloodLight u OpenDayLight. En cuanto al conmutador, el más aceptado es Open vSwitch. [13] (Openvswitch, 2015).

También, cabe citar algunas características de OpenFlow adicionales a las propias de SDN, como:

• El tráfico experimental y el de producción pueden compartir el mismo conmutador, definiendo por una parte, en una entrada de la tabla de flujo, la acción de enviar el paquete al datapath o procesamiento normal del dispositivo; por otra, el tratamiento de investigación.

• El controlador tiene un conocimiento amplio de la red, y no local como es tradicional. Actualiza dinámicamente el reenvío, ya que puede modificar las entradas de la tabla de flujo en cualquier momento y en caso de fallo de enlace, este elemento puede encontrar nuevas rutas más rápidamente. Mediante herramientas como FlowVisor se puede compartir el control de un conmutador entre varios controladores o balance de carga con ejemplos como Plug-n-Serve.

• Abstracción de flujos. Con una sola entrada en la tabla, se puede gestionar un flujo de tráfico completo. [14] (Serrano, 2015).

2.4. OpenFlow Switch

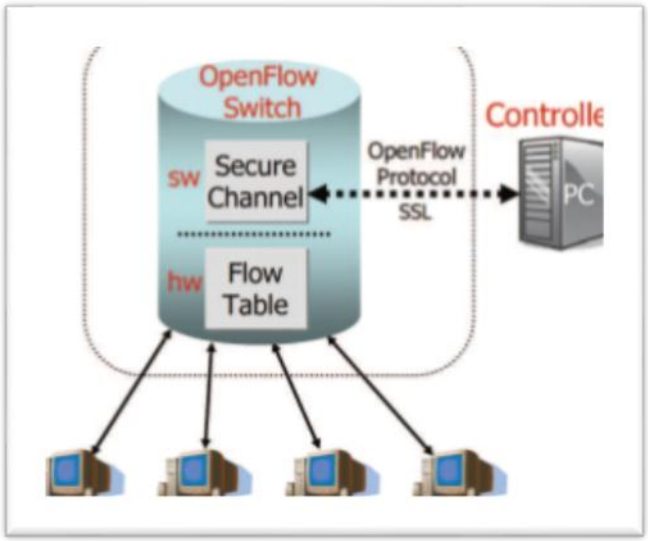
El camino de datos (datapath) del switch OpenFlow radica en una tabla de flujo y otra acción que se asocia con la entrada de flujo, el conjunto de acciones que soporta un switch OpenFlow es extensible aunque se establecen mínimos requisitos para la totalidad de switches. Se Para implementar el protocolo OpenFlow se debe partir de una premisa, debe contarse con un Switch OpenFlow, cuyos básicos requerimientos se proceden a explicar:

El switch OpenFlow tiene 3 partes, una Tabla de Flujo, en primer lugar, con acción asociada a cada entrada de flujo, la cual indica al switch la forma como procesarlo; en segundo lugar, canal seguro que permite conectar al switch con el proceso de control remoto, esto recibe el nombre de controlador (controller), con ello permite que los comandos y paquetes se puedan intercambiar entre el switch y controlador por medio del protocolo OpenFlow, que proporciona camino abierto y estandarizado para que se produzca la comunicación en cuestión. Ante ello, por medio del uso del protocolo, el controlador puede actualizar, añadir y borrar las entradas de flujo, de forma reactiva al igual que proactiva.

Es de mucha importancia que se dividan los switches en dos categorías principales: Switches OpenFlow, estos no soportan el procesamiento de nivel 2 y nivel 3, también Switches Ethernet y habilitadores de OpenFlow, tienen interfaces del protocolo como característica extra.

Switches OpenFlow dedicados: soporta los formatos de encabezados y acciones que son definidas en especificación del switch Openflow, dicho switch tiene rutas de datos que permite el reenvío de paquetes entre puertos, de igual manera como sucede con el servidor controlador. Se procede en la siguiente figura #1 a mostrar un Openflow dedicado.

Figura 1: Switch OpenFlow dedicado. [15] Escobar, (2015).



Las entradas de flujo pueden tener acción asociada simple, tres básicas que soportan los switches OpenFlow son:

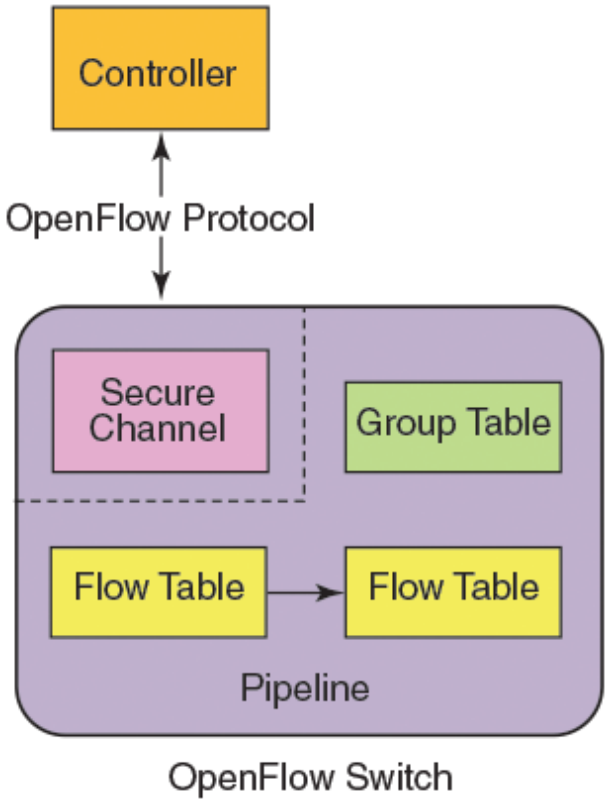
1. Envían el flujo de paquetes hasta un puerto establecido, esto hace que los paquetes se enruten por medio de la red, en la mayoría de switches dicha función se espera sea tomada en la velocidad de línea.

2. El encapsulamiento y envío de estos flujos de paquetes al controlador (*controller*), se dan por el canal seguro donde son encapsulados y enviados al controlador,

de forma típica se utiliza el primer paquete en un nuevo flujo, donde el controlador decide si el flujo deba ser añadido a una tabla de flujos, en muchos experimentos se puede utilizar para el envío de los paquetes en su totalidad al controlador de tal forma que sean procesados.

3. Lanzamiento del flujo de paquetes, mismos que pueden ser usados para seguridad, con ello se frena la negación de ataques de servicios, al igual que, para disminuir el descubrimiento de tráfico falso de broadcast por medio de los usuarios finales.

Figura 2: Estructura de un Switch OpenFlow, [31] Brocade, (2016).



2.5. Tablas de OpenFlow

2.5.1. Tabla de Flujos

Cada tabla de flujo en el switch tiene conjunto de entradas, donde cada una de estas representa a los contadores (counters), campos coincidentes (match fields), así como una serie de instrucciones a aplicarse a los paquetes coincidentes.

Cada tabla de flujo contiene los siguientes componentes:

Principales componentes de una entrada en una tabla de flujo

MATCH FIELDS	COUNTERS	INSTRUCTIONS

a) Match Fields, campos coincidentes: sirven para detectar paquetes, mismos que consisten en puerto de entrada y cabeceras, también opcionalmente como metadata especificada por tabla previa.

b) Counters, contadores: permiten actualizar los paquetes coincidentes.

c) Instrucciones, instructions, permite modificar todo el conjunto de acciones así como el procesamiento canalizado.

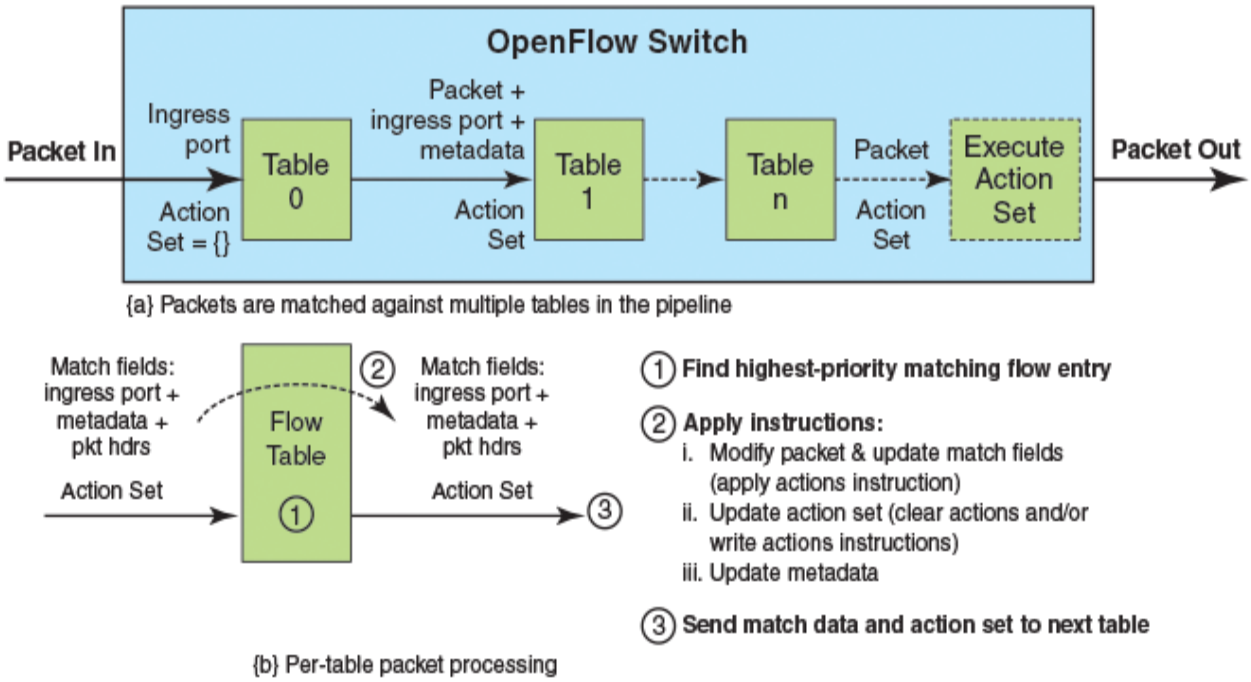
La coincidencia de los paquetes empieza en la tabla de primera fila, puede continuar a tablas de filas adicionales, dichas entradas permiten la coincidencia de paquetes en orden de prioridad, donde el resto de coincidencias que se ubican ordenada a cada tabla utilizada. Si la entrada de coincidencia es encontrada, las instrucciones que están asociadas con cada fila específica se ejecutan, de no encontrar la coincidencia en alguna de las filas de tabla, la salida va a depender de la configuración del switch: siendo posible que el paquete se remita al controlador por medio de la canal OpenFlow, desechado o que puede continuar a la verificación de otras filas de tablas.

Un switch que está alineado con OpenFlow puede ser de dos formas: Solo OpenFlow o un Híbrido OpenFlow. Los switches "solo OpenFlow" soportan únicamente operaciones OpenFlow, en esos switches todos los paquetes son procesados por la tubería (pipeline) OpenFlow y no pueden ser procesados de alguna forma alternativa.

Los switches híbridos OpenFlow soportan tanto operaciones con OpenFlow como aquellas de switching Ethernet estándar, por ejemplo: Switching Ethernet L2 tradicional, aislamiento de VLAN, enrutamiento L3, ACL y procesamiento de calidad de servicios. Estos switches deberían proveer de un mecanismo de clasificación aparte de OpenFlow que enrute tramas tanto para una tubería OpenFlow como para una estándar. Por ejemplo un switch puede utilizar una etiqueta VLAN o un puerto de entrada del paquete para decidir si procesar el paquete utilizando una tubería u otra. Así como también podría dirigir todos los paquetes directamente a la tubería OpenFlow, esto será determinado según el mecanismo de clasificación sea configurado.

Un switch híbrido OpenFlow puede también permitir a un paquete ir desde la tubería OpenFlow a la tubería normal a través de los puertos virtuales NORMAL e "INUNDACIÓN". La tubería OpenFlow de cada switch contiene múltiples tablas de flujo que a su vez contiene varias. La tubería de procesamiento OpenFlow define como los paquetes interactúan con aquellas tablas. Un switch OpenFlow con una única tabla de flujo es válido, en este caso el procesamiento de tubería se ve altamente simplificado.

Figura 3. Flujo de paquetes del procesamiento Pipeline [31] Brocade, (2016).



2.6. Tabla de Grupo

Una tabla de grupo consiste en entradas de grupo. La habilidad de un flujo para apuntar a un grupo permite a OpenFlow representar métodos adicionales de transmisión.

Cada entrada de grupo contiene lo siguiente:

2.6.1. Estructura de una Entrada de Grupo (Identificador de grupo, Tipo de grupo, Contadores y Cubos de acción).

Group identifier	Group type	Counters	Action buckets

a) Identificador de Grupo: un Integer de 32 bits sin firmar que sirve únicamente para identificar al grupo.

b) Tipo de Grupo: Determina la semántica del grupo.

c) Contadores: Permiten la actualización de la información una vez que los paquetes son procesados por un grupo.

Tipos de Grupos

Se encuentran definidos los siguientes tipos de grupos.

Todos: Ejecutan todas las acciones dentro del grupo. Este grupo se utiliza para transmisión multicast o broadcast. El paquete se clona de forma efectiva para cada cubo; un paquete por cubo es procesado. Si un cubo dirige un paquete explícitamente fuera del puerto de ingreso, este paquete clonado es declinado. Si el controlador desea transmitir por el puerto de ingreso, el grupo debería incluir un cubo "extra" que incluya una acción de salida al puerto virtual OFPP_IN.

Seleccionado: Ejecuta una acción del grupo. Los paquetes son enviados a un cubo único del grupo, basado en un algoritmo de selección de interrupción computada. Toda la configuración y estado del algoritmo de selección es externo a OpenFlow. Cuando un puerto en específico de un cubo de un grupo determinado se vienen abajo, el switch puede restringir la selección de cubos dentro del conjunto remanente (aquellos con acciones de transmisión a puertos activos) en lugar de paquetes declinados destinados a ese puerto. Este comportamiento puede reducir la ruptura de un link o switch caído.

Indirecto: Ejecuta una acción definida en este grupo. Permite múltiples flujos o grupos para apuntar a un identificador de grupo más rápido, soportando convergencia más rápida y eficiente (por ejemplo: para siguientes saltos de transmisión IP). Este tipo de grupo es idéntico al grupo todos solo que con un solo cubo.

Conmutación por error rápida: Ejecuta el primer cubo vivo (activo).Cada acción está asociada con un puerto específico y/o un grupo que controlar su estado activo, esto le permite al switch cambiar la transmisión sin requerir un round-trip (ida y vuelta) al controlador. Si ningún cubo está vivo, los paquetes son descartados. Este tipo de grupo debe implementar el mecanismo de conexión de vida.

2.6.2. Canal OpenFlow

El canal OpenFlow es la interfaz que conecta cada switch OpenFlow a un controlador. A través de esta interfaz, el controlador configura y administra el switch, recibe eventos del switch y envía paquetes fuera de este.

Entre el camino de datos y el canal OpenFlow, la interfaz se implementa de forma específica, sin embargo todos los mensajes del canal OpenFlow deben poseer un formato acorde al protocolo OpenFlow. El canal OpenFlow es usualmente encriptado utilizando TLS (Transport Layer Security), pero este canal puede correr directamente en TCP (Transmission Control Protocol). La capacidad de soportar múltiples controladores simultáneos no está actualmente definida.

2.6.3. Visión general de Protocolo OpenFlow:

El protocolo OpenFlow soporta tres (3) tipos de mensajes, de controlador-a-switch, asíncrono, y simétrico, cada uno con múltiples sub-tipos. Los mensajes Controlador-aswitch son iniciados por el controlador y utilizados para administrar o inspeccionar directamente el estado del switch. Los mensajes asíncronos son iniciados por el switch y utilizados para actualizar el controlador de los eventos de red y cambiar al estado de switch. Los mensajes simétricos inician ya sea por el switch o por el controlador sin solicitud. A continuación una descripción detallada de los mensajes presentes en OpenFlow.

2.6.4. Mensajes Controlador a Switch

Los mensajes Controlador/switch son inicializados por el controlador y pueden o no pueden requerir una respuesta por parte del switch.

Características: El controlador puede requerir las capacidades de un switch al transmitir una solicitud de características; el switch debe dar una respuesta que especifique las capacidades y características del switch. Esto es comúnmente realizado al momento que se establece el canal OpenFlow.

Configuración: El controlador es capaz de establecer y pedir parámetros de configuración en el switch. El switch solo responde a una solicitud emitida por el controlador.

Estado de Modificación: los mensajes de estados de modificación son enviados por el controlador para administrar el estado de los switches. El principal propósito es añadir/borrar y modificar flujos/grupos en las tablas OpenFlow y establecer las propiedades del switch.

Lectura-Estado: los mensajes de lectura-estado son utilizados por el controlador para recopilar estadísticas del switch.

Salida-de-Paquete: estos son utilizados por el controlador para enviar paquetes fuera del puerto especificado en el switch y enviar paquetes recibidos a través de mensajes del tipo Entrada-de-Paquetes, los mensajes tipo salida-de-paquete deben contener un paquete lleno o una identificación de buffer (regulador) que haga referencia al paquete almacenado en el switch. El mensaje debe contener de igual forma una lista de acciones que deben ser aplicadas en el orden en el que son especificadas; una lista de acciones vacía (que no posea ningún tipo de especificaciones) declina el paquete.

Barrera: Una solicitud de barrera/ o mensajes de respuesta son utilizados por el controlador para asegurar que las dependencias de los mensajes se han cumplido o que las notificaciones para operaciones se han recibido completamente.

Mensajes Asíncronos: Los mensajes asíncronos son enviados sin que el controlador los solicite al switch. Los switches transmiten mensajes asíncronos al

controlador para denotar la llegada de un paquete, el estado del switch cambia, o denota un error. Los cuatro (04) tipos de mensajes asíncronos se describen a continuación:

Entrada-de-Paquete: Para todos los paquetes que no tienen una coincidencia de entrada de flujo, un evento de Entrada-de-Paquete puede ser enviada al controlador (dependiendo de la tabla de configuración). Para todos los paquetes transmitidos al puerto virtual del controlador, siempre se transmite un evento de Entrada-de-Paquete al controlador.

Si el switch tiene suficiente memoria para amortiguar los paquetes que son transmitidos controlador, el evento contendrá alguna fracción de la cabecera del paquete (la cual tiene un valor por defecto de 128 bytes) y un identificador de buffer que utilizará el controlador cuando el switch se encuentre listo para transmitir el paquete. Los switches que no soporten amortiguación interna (internal buffering) deben transmitir todo el paquete al controlador como parte del evento.

Los paquetes amortiguados serán usualmente procesador a través de un mensaje de tipo Salida-de-Paquete por parte del controlador, o expirarán de forma automática después de un intervalo de tiempo.

Eliminación de Flujo: cuando se añade una entrada de flujo al switch por un mensaje de modificación de flujo, un valor de tiempo al vacío indica cuando la entrada debe ser removida debido a la falta de actividad, de igual forma que el tiempo indica en que momento la entrada debe ser removida, independientemente de su estado de actividad/inactividad.

Estatus del Puerto: Se espera el switch transmita mensajes de estatus del puerto a los controladores a medida que cambie el estado de configuración de puerto. Estos eventos incluyen cambios en los eventos de estatus del puerto (Por ejemplo: Si es ejecutada alguna modificación directamente por un usuario).

Mensajes Simétricos: Los mensajes simétricos son transmitidos sin que haya solicitud previa, en cualquier sentido. Hola: Los mensajes hola son intercambiados entre el switch y el controlador al inicio del establecimiento de la conexión. Eco: La respuesta de eco/mensajes de respuesta puede ser transmitida por parte del switch o del controlador y deben retornar como una respuesta de eco. Pueden ser utilizados para medir la latencia o el ancho de banda de una conexión controlador switch, de igual manera para verificar el tiempo de vida.

Experimentación: los mensajes de experimentación proporcionan una manera estándar para que los switches OpenFlow ofrezcan funcionalidades dentro del espacio de mensajes OpenFlow. Esta área se encuentra en período de pruebas y está previsto sean incorporadas mÁs funcionalidades en futuras revisiones del protocolo.

Establecimiento de la conexión. El switch debe ser capaz de establecer comunicación con un controlador con una dirección IP de usuario configurable, utilizando un puerto en específico, si el switch conoce la dirección IP del controlador, el switch inicia una conexión TCP o TLS estándar con el controlador. El tráfico hacia y desde el canal OpenFlow no fluye a través la tubería OpenFlow.

Por este motivo, el switch debe identificar el tráfico de entrada local antes de verificarlo con las tablas de flujo, en especificaciones de futuras versiones del protocolo, será descrito un protocolo de descubrimiento de controlador dinámico en el cual la dirección IP y el puerto de comunicaciones con el controlador son determinados durante el tiempo de ejecución.

Cuando se establece una conexión OpenFlow, cada lado de la conexión debe transmitir un mensaje OFPT_HELLO con la versión de campo en su valor más alto dentro del protocolo OpenFlow soportado. Al recibir este mensaje, el receptor debe calcular la

versión de protocolo OpenFlow a utilizarse, tomando como parámetro que sea una versión más baja que la del emisor y la del receptor.

Si la versión que se establece es soportada por el receptor, entonces la conexión puede proceder, de otra forma el receptor debe responder con un mensaje OFPT_ERROR con un campo de tipo de OFPET_HELLO_FAILED con el campo de código que diga OFPHFC_COMPATIBLE y opcionalmente un string de tipo ASCII que explica la situación de la data y posteriormente se terminaría la conexión. [04] (Moreno, 2012).

CAPÍTULO III

3. Metodología

La presente investigación es de tipo cualitativa y descriptiva. Las técnicas e instrumentos utilizados para la recolección de datos, además se describirán las herramientas que se utilizarán para este proyecto.

3.1 Hipótesis

La propuesta de implementación del protocolo openflow en el modelo red definida por software optimizara y controlara el tráfico que corre por las redes de comunicaciones de la Universidad Técnica de Manabí.

3.1.1 Variables

Variable independiente

Protocolo OPENFLOW SDN

Variable dependiente

Optimizara y controlara el tráfico que circule por las redes de comunicaciones de la Universidad Técnica de Manabí.

3.2 Metodología

La presente investigación es de tipo descriptiva porqué se desea especificar las característica de los administradores de redes usando las redes definidas por software usando el protocolo Openflow, además también cualitativa porque pretende estudiar la percepción de este nuevo modelo en nuestro entorno y en el uso de redes académicas.

3.2.1 Técnica de recolección de datos

Las técnicas utilizadas fueron observación directa, entrevistas, encuestas a usuarios y profesionales de la Universidad Técnica de Manabí y de otras instituciones que se detallan a continuación.

Método de Observación Directa

Esta técnica permitió observar cómo funciona la red de comunicaciones en la universidad en qué estado se encuentra en que momentos tiene congestionamiento un tráfico alto lo que disminuye la satisfacción de la comunidad universitaria en el uso de este servicio en base a esto se propone el uso de redes definidas por software para mejorar el rendimiento.

Método de Análisis bibliográfico

Libros e Internet.- Estos instrumentos suministraron los datos necesarios para el proceso de investigación.

Documentos de la UTM.- Se utilizó información de la Universidad Técnica de Manabí los planos y diseño de la red, número de equipos con los que cuenta la universidad, ancho de banda asignado a cada Facultad y departamentos, número de empleados, docentes y alumnos que están actualmente.

Método de Encuesta y Entrevista

La técnica cualitativa utilizada en esta investigación, es la entrevista, la cual será aplicada a el Administrador de redes y comunicaciones, Líder del centro de cómputo, Coordinador de Tic de otros centros de cómputo de la provincia, para la obtención de los datos que darán cumplimiento a los objetivos.

La entrevista permitirá analizar sus opiniones, criterios orientaciones y consejos para ser tomados en cuenta en la propuesta.

Las técnicas cuantitativas utilizadas para los usuarios, que permitirá medir si la comunicación actual es satisfactoria, dando con ello cumplimiento a los objetivos de esta investigación, serán las encuestas que permitan conocer la percepción del usuario.

- **Población y Muestra**

La componen docentes de la Facultad de ciencias informáticas son 35 que son las personas que hace uso de las redes de comunicaciones y envían proyectos a los alumnos con esta encuesta se pretende conocer el conocimiento que tienen los docentes en este nuevo protocolo.

Para esta investigación se tomará el número total de docentes de la facultad de ciencias informáticas para calcular el número de la muestra se usó la siguiente formula además consta en el Anexo A.

Para el cálculo del tamaño óptimo de la muestra se aplica la siguiente fórmula:

( ) ( )

Dónde:

z = constante que determina nivel de confianza.

p = es la proporción de individuos que poseen en la población la característica del estudio.

q = 1 – p

N = Es el tamaño de la población o universo.

e = es el error muestra deseado.

Datos

n=?

Z (Nivel de confianza)=90% que equivale al (1.65)

p=0.50 q=0.5

p=0.50 N=35

e= 0.08

Se concluye que con un nivel de confianza del 90% y un margen de error del 0,08 se deben realizar encuesta a 21 docentes de la facultad de ciencias informáticas.

3.3 Herramientas a utilizar en el proyecto

En la actualidad las universidades se ven en una constante búsqueda de optimizar recursos y mejorar su infraestructura y aprovechar la utilización de su red de comunicaciones con menores costo tomando en cuenta que la economía del país no está en sus mejores momentos la solución es usar sistemas y protocolos abiertos en esta tesis se utilizara la aplicación de Openflow y SDN.

Antes definir su aplicación se describirá las herramientas necesarias para su actividad, las pruebas se realizaran en el sistema operativo Ubuntu 14.04 se pueden realizar en otros sistemas operativos como Windows se recomienda Linux porque involucra la instalación de menos paquetes.

Dentro de estas herramientas tenemos las de virtualización de las que usaremos Virtualbox, VMware Workstation, software terminal putty, terminal de gnome ssh. Por otro lado, para evaluar el uso de la virtualización que pueda suministrar el soporte para el diseño de un método de Capacitación en redes. ([16] Fernández & García, 2011).

3.3.1 Virtual Box

VirtualBox es un potente x86 y AMD64 / Intel64 virtualización producto para la empresa, así como el uso doméstico, no sólo es VirtualBox un extremadamente rico en características, producto de alto rendimiento para clientes empresariales, es también la única solución profesional que está libremente disponible como software de código abierto bajo los términos de la Licencia Pública General de GNU (GPL).

Actualmente, VirtualBox se ejecuta en Windows, Linux, Macintosh y anfitriones Solaris y soporta un gran número de sistemas operativos invitados incluyendo pero no limitado a Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, Windows 10), 3.xy DOS / Windows, Linux (2.4, 2.6, 3.xy 4.x), Solaris y OpenSolaris, OS / 2, y OpenBSD.

3.3.2 VMware Workstation

VMware Workstation Pro transforma el modo en profesionales técnicos a desarrollar, probar, demostrar y desplegar el software mediante la ejecución de múltiples sistemas operativos basados en x86 de forma simultánea en el mismo PC. Construido sobre 15 años de excelencia en la virtualización y ganador de más de 50 premios de la industria, VMware Workstation Pro lleva la virtualización de escritorio a un nivel superior, dando a los usuarios una compatibilidad del sistema operativo sin igual, una rica experiencia de usuario y un rendimiento increíble.

VMware Workstation Pro aprovecha el hardware más reciente para replicar entornos de servidor, escritorio y tabletas en una máquina virtual. Aplicaciones se ejecutan en una amplitud de sistemas operativos, incluyendo Linux, Windows y más en el mismo tiempo en el mismo PC sin necesidad de reiniciar. VMware Workstation Pro hace que sea muy fácil de evaluar nuevos sistemas operativos como Windows 10, aplicaciones de software de prueba y parches, y arquitecturas de referencia en un entorno aislado y seguro. Ningún otro software de virtualización de escritorio ofrece las características de rendimiento, fiabilidad y vanguardia de la estación de trabajo Pro.

Crear máquinas virtuales potentes

Con la capacidad de asignar múltiples núcleos de procesamiento, gigabytes de memoria principal y la memoria de gráficos para cada máquina virtual, la estación de trabajo Pro maximiza los recursos de su PC para ejecutar las aplicaciones más exigentes en un entorno virtual. Estación de Trabajo Pro, incluso permite ejecutar hosts ESXi virtuales y aparatos vSphere, que le permite crear todo un Software Defined Data Center en tu ordenador. [18] (Vmware, 2015).

3.3.3 SSH Secure Shell Client

SSH (o Secure SHell) es un protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor y que permite a los usuarios conectarse a un host remotamente. A diferencia de otros protocolos de comunicación remota tales como FTP o Telnet, SSH encripta la sesión de conexión, haciendo imposible que alguien pueda obtener contraseñas no encriptadas.

SSH está diseñado para reemplazar los métodos más viejos y menos seguros para registrarse remotamente en otro sistema a través de la shell de comando, tales como telnet o rsh. Un programa relacionado, el scp, reemplaza otros programas diseñados para copiar archivos entre hosts como rcp. Ya que estas aplicaciones antiguas no encriptan contraseñas entre el cliente y el servidor, evite usarlas mientras le sea posible. El uso de métodos seguros para registrarse remotamente a otros sistemas reduce los riesgos de seguridad tanto para el sistema cliente como para el sistema remoto. [19] (Red Hat Enterprise, 2005)

3.3.3.1 Características de SSH

El protocolo SSH proporciona los siguientes tipos de protección:

- Después de la conexión inicial, el cliente puede verificar que se está conectando al mismo servidor al que se conectó anteriormente.
- El cliente transmite su información de autenticación al servidor usando una encriptación robusta de 128 bits.

Todos los datos enviados y recibidos durante la sesión se transfieren por medio de encriptación de 128 bits, lo cual los hacen extremamente difícil de descifrar y leer.

El cliente tiene la posibilidad de reenviar aplicaciones X11 desde el servidor. Esta técnica, llamada reenvío por X11, proporciona un medio seguro para usar aplicaciones gráficas sobre una red.

Ya que el protocolo SSH encripta todo lo que envía y recibe, se puede usar para asegurar protocolos inseguros, el servidor SSH puede convertirse en un conducto para convertir en seguros los protocolos inseguros mediante el uso de una técnica llamada reenvío por puerto, como por ejemplo POP, incrementando la seguridad del sistema en general y de los datos.

Red Hat Enterprise Linux contiene el paquete general de OpenSSH (openssh) así como también los paquetes del servidor OpenSSH (openssh-server) y del cliente (openssh-clients), consulte el capítulo titulado OpenSSH en el Manual de administración del sistema de Red Hat Enterprise Linux para obtener instrucciones sobre la instalación y el desarrollo de OpenSSH. Observe que los paquetes OpenSSH requieren el paquete OpenSSL (openssl). OpenSSL instala varias bibliotecas criptográficas importantes, permitiendo que OpenSSH pueda proporcionar comunicaciones encriptadas. [20] (Red Hat Enterprise, 2005)

3.3.4 Xterm

El programa xterm es un emulador de terminal para el sistema X Window. Proporciona diciembre VT102 y Tektronix 4014 terminales compatibles para los programas que no pueden utilizar el sistema de ventanas directamente.

Esta versión implementa colores ISO / ANSI utilizando el "nuevo" modelo de color (es decir, el color de fondo de borrado). También implementa la mayor parte de las secuencias de control para VT220, así como características seleccionadas de otros terminales DEC tales como VT320, VT420 y VT520. [21] (Thomas E., 2014)

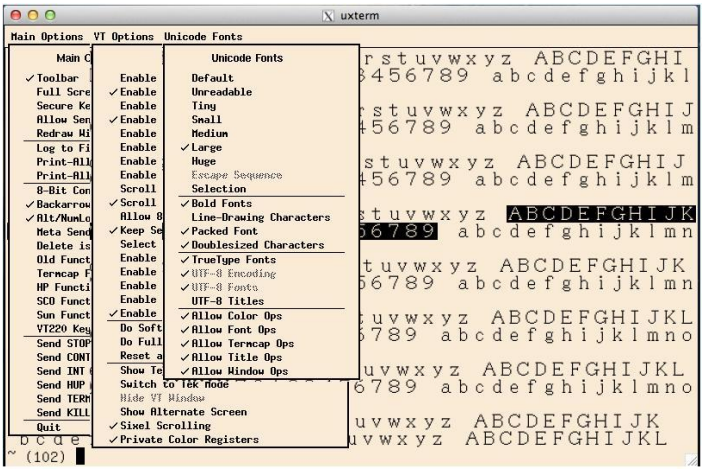
XTerm FAQ

El XTerm FAQ da más información, mostrando cómo resolver problemas con él, así como xterm contraste con otros terminales.

XTerm Cambio de registro

El registro de cambios muestra cuándo y por qué se agregaron o cambiaron características.

Figura 4, Comando Xterm, [21] (Thomas E., 2014)



3.3.4 Dpctl

Dpctl es una utilidad de gestión que permite un cierto control sobre el interruptor OpenFlow. Con esta herramienta es posible añadir los flujos a la mesa de flujo, consultar para funciones de conmutación y el estado, y cambiar otras configuraciones. [22] (Github, 2013)

Un comando Dpctl tiene la siguiente sintaxis

Dpctl [OPTIONS] SWITCH COMMAND [ARG...]

Características

Dpctl envía un mensaje de petición de características para el interruptor, que responde con una cuenta respuesta. [14] (Github, 2013)

Ejemplo de Uso

dpctl unix:/var/run/s1.sock features

El mensaje de respuesta debe tener este aspecto.

feat_repl{dpid="0x000002d5e8cde975", buffs="256", tabs="64", aux_id="0", caps="0x4f"]}

Get-config y set-config

Dpctl puede consultar y establecer los parámetros de configuración de los interruptores.

Ejemplo de Uso

Vamos a preguntar para la configuración del switch. Debería ver los valores predeterminados para las banderas de configuración y el número máximo de bytes de un paquete, MLEN, que el conmutador puede enviar al controlador. [18] (Github., 2013) dpctl unix:/var/run/s1.sock get-config SENDING: conf_req RECEIVED: conf_repl{conf={flags="0x.

Comandos de puerto

Dpctl tienen comandos para obtener las estadísticas del puerto, descripción puerto y para modificar el comportamiento de puerto. [23] (Github., 2013)

Puerto-desc

Este comando envía un mensaje solicitando la descripción puertos, con las configuraciones actuales.

3.3.5 Wireshark

Es un importante analizador de protocolos de red del mundo. Le permite ver lo que está sucediendo en su red a nivel microscópico. Es el estándar de facto (ya menudo de jure) a través de muchas industrias y las instituciones educativas.

Desarrollo Wireshark prospera gracias a los aportes de la creación de redes de expertos en todo el mundo. Es la continuación de un proyecto que comenzó en 1998. [24] (Wireshark, 2014)

Universidad Wireshark Co-fundada por Laura Chappell, instructor inspirador, consultor y experto en Wireshark, ofrece capacitación, certificación de Network

Analyst, y los recursos de todos los niveles de usuarios de Wireshark. [24] (Wireshark, 2014)

Wireshark tiene un rico conjunto de características que incluye lo siguiente:

Inspección profunda de cientos de protocolos, con más que se añade todo el tiempo

- Captura en vivo y análisis fuera de línea De tres paneles estándar explorador de paquetes.
- Multiplataforma: Se ejecuta en Windows, Linux, OS X, Solaris, FreeBSD, NetBSD, y muchos otros.
- Datos de red capturados se pueden consultar a través de una interfaz gráfica de usuario, o por medio de la utilidad TTY-mode TShark
- Los más poderosos filtros de visualización en la industria Análisis VoIP Rich
    - Leer / escribir muchos diferentes formatos de archivo de captura: tcpdump (libpcap), Pcap NG, Catapult DCT2000, Cisco Secure IDS iplog, Microsoft Network Monitor,
- Red Sniffer general (comprimido y sin comprimir), Sniffer Pro y NetXray,
- Red de Instrumentos de Observación NetScreen, snoop, Novell LANalyzer, RADCOM WAN / LAN Analyzer, Shomiti / Finisar Surveyor, Tektronix K12xx,Redes Visuales Visual UpTime, WildPackets EtherPeek / TokenPeek / AiroPeek, y muchos otros
- Archivos de captura comprimidos con gzip pueden ser descomprimidos sobre la marcha
- Datos en tiempo real se pueden leer desde Ethernet, IEEE 802.11, PPP / HDLC, ATM, Bluetooth, USB, Token Ring, Frame Relay, FDDI, y otros (dependiendo de la plataforma)
- Apoyo descifrado para muchos protocolos, incluyendo IPsec, ISAKMP, Kerberos, SNMPv3, SSL / TLS, WEP y WPA / WPA2.

- Reglas para colorear se pueden aplicar a la lista de paquetes para, análisis intuitivo rápida
- La salida puede ser exportado a XML, PostScript, CSV, o texto sin formato [24] (Wireshark, 2014)

3.3.6 Iperf3

iPerf3 es una herramienta para medidas activas del ancho de banda máximo alcanzable en redes IP. Es compatible con la sintonización de diversos parámetros relacionados con la temporización, tampones y protocolos (TCP, UDP, SCTP con IPv4 e IPv6). Para cada prueba se informa el ancho de banda, pérdida, y otros parámetros. Esta es una nueva aplicación que comparte ningún código con el iperf original y tampoco es compatible con versiones anteriores. iperf fue desarrollado originalmente por NLANR / DAST. iPerf3 se desarrolla principalmente por ESnet / Laboratorio Nacional Lawrence Berkeley. Es liberado bajo una de tres cláusulas licencia BSD. [25] (Iperf, 2014)

Características Iperf

TCP y SCTP

Medida ancho de banda

Tallas MSS / MTU y observó tamaños leer.

Soporte para TCP tamaño de la ventana a través de buffers de socket. UDP

El cliente puede crear flujos UDP de ancho de banda especificado.

Pérdida de paquetes Medida

Medida demora jitter

Multidifusión

Multiplataforma: Windows, Linux, Android, Mac OS X, FreeBSD, OpenBSD, NetBSD, VxWorks, Solaris.

Multi-threaded: El cliente y el servidor pueden tener múltiples conexiones simultáneas (-P opción).

Server maneja múltiples conexiones, en lugar de dejar de fumar después de una sola prueba.

Puede funcionar por el tiempo especificado (opción -t), en lugar de una cantidad fija de datos para transferir (-n o la opción -k).

Imprimir periódica, ancho de banda intermedia, jitter y pérdida de los informes a intervalos especificados (opción -i).

Ejecutar el servidor como un demonio (opción -D)

Utilizar arroyos representativos para probar cómo la compresión de la capa de enlace afecta su ancho de banda posible (-F opción).

Nuevo: Ignorar slowstart TCP (-O opción).

Nuevo: ancho de banda objetivo fijado para UDP y (nuevo) TCP (opción -b).

Nuevo: etiqueta de flujo Conjunto IPv6 (opción -L)

Algoritmo Conjunto de control de congestión (opción -C): Nuevo

Nuevo: Uso SCTP en lugar de TCP (opción --sctp)

- Nuevo: Salida en formato JSON (-J opción).
- Nuevo: lectura de disco de prueba (servidor: iperf3 -s / cliente: iperf3 -c testhost -i1 -F nombre de archivo)
- Pruebas de escritura de disco: Nuevo (servidor: iperf3 -s -F nombre de archivo / cliente: iperf3 -c testhost -i1) [21] (Iperf, 2014).

3.3.7 Mininet

MiniNet crea una red virtual realista, corriendo núcleo verdadero, interruptor y código de la aplicación, en una sola máquina (VM, nube o nativo), en el segundo, con un solo comando. [26] (Mininet, 2015)

Porque se puede fácilmente interactuar con la red mediante la MiniNet CLI (y API), personalizar él, compartir con los demás, o implementar en hardware real, MiniNet es útil para el desarrollo, la enseñanza y la investigación.

MiniNet es también una gran manera de desarrollar, compartir y experimentar con OpenFlow y sistemas de redes definidas por software.

MiniNet se desarrolla activamente y apoyado, y se distribuye bajo una licencia BSD Open Source permisiva. Le animamos a contribuir código, informes de errores / correcciones, documentación y cualquier otra cosa que se puede mejorar el sistema. [26] (Mininet, 2015)

3.3.7.1 ¿Por qué Mininet es genial?

1. Es rápido - puesta en marcha de una red simple toma sólo unos segundos. Esto significa que su bucle run-edit-debug puede ser muy rápido.
2. Puede crear topologías personalizadas: un solo interruptor, topologías más grandes de Internet similar, la columna vertebral de Stanford, un centro de datos, o cualquier otra cosa.
3. Puede ejecutar programas reales: cualquier cosa que se ejecuta en Linux está disponible para ejecutar, de los servidores web a TCP herramientas de supervisión de la ventana a Wireshark.
4. Puede personalizar el reenvío de paquetes: interruptores de MiniNet son programables mediante el protocolo OpenFlow. Los diseños de red definidos por software que se ejecutan en Mininet se pueden transferir fácilmente a los conmutadores OpenFlow de hardware para el reenvío de paquetes de velocidad de línea.
5. Puede ejecutar MiniNet en su ordenador portátil, en un servidor, en una máquina virtual, en una máquina Linux nativo (MiniNet se incluye con Ubuntu 12.10+!), O en la nube (por ejemplo, Amazon EC2).

6. Puede compartir y replicar los resultados: cualquier persona con una computadora puede ejecutar el código una vez que haya empaquetado para crear.
7. Se puede utilizar fácilmente: puede crear y ejecutar experimentos MiniNet escribiendo scripts simples (o complejos si es necesario) Python.
8. MiniNet es un proyecto de código abierto, por lo que se le anima a examinar su código fuente en https://github.com/mininet , modificarlo, corregir los errores, problemas de archivo / peticiones de características, y enviar parches / tirar de solicitudes. También puede editar esta documentación para corregir errores o agregar aclaraciones o información adicional.
9. MiniNet está bajo desarrollo activo. Por lo tanto, si se aspira, no tiene sentido, o no funciona por alguna razón, por favor sepamos en mininet-discuss y el usuario y el desarrollador de la comunidad MiniNet puede tratar de explicar el problema, solucionarlo, o ayudar a solucionarlo). Si encuentras bugs, te animamos a enviar parches para solucionarlos, o al menos para enviar un problema en github incluyendo un caso de prueba reproducible. [27] (Alcalde, 2015)

3.3.7.1.2 Cuáles son las limitaciones de Mininet?

Aunque creemos que Mininet es grande, tiene algunas limitaciones. Por ejemplo,

- Ejecutar en un solo sistema es conveniente, pero impone límites de recursos: si su servidor tiene 3 GHz de CPU y puede cambiar alrededor de 10 Gbps de tráfico simulado, esos recursos tendrán que ser equilibrados y compartidos entre sus hosts virtuales y switches.
- Mininet usa un único kernel de Linux para todos los hosts virtuales; Esto significa que no puede ejecutar software que dependa de BSD, Windows u otros kernels del sistema operativo.(Aunque puede conectar VMs a Mininet.)
- Mininet no escribirá su controlador OpenFlow para usted; Si necesita un enrutamiento personalizado o un comportamiento de conmutación, tendrá

que encontrar o desarrollar un controlador con las características que necesita.

- Por defecto su red de Mininet está aislada de su LAN y del Internet - esto es generalmente una buena cosa! Sin embargo, es posible utilizar el NAT objeto y / o la --nat opción de conectar su red MiniNet a la red LAN a través de la traducción de direcciones de red. También puede adjuntar una interfaz de hardware real (o virtual) a su red MiniNet (verexamples/hwintf.py para más detalles)
- De forma predeterminada, todos los hosts de Mininet comparten el sistema de archivos host y el espacio PID; Esto significa que puede tener que tener cuidado si está ejecutando demonios que requieren configuración en / etc., y debe tener cuidado de no matar los procesos erróneos por error. (Tenga en cuenta la bind.py ejemplo demuestra cómo hacer que los directorios privados por el huésped).
- A diferencia de un simulador, Mininet no tiene una noción fuerte de tiempo virtual; Esto significa que las mediciones de tiempo se basarán en tiempo real y que los resultados más rápidos que en tiempo real (por ejemplo redes de 100 Gbps) no pueden emularse con facilidad.

Un aspecto aparte en el rendimiento: Lo principal que debe tener en cuenta para los experimentos con limitaciones de red es que probablemente necesitará usar enlaces más lentos, por ejemplo 10 o 100 Mb / seg en lugar de 10 Gb / s, debido al hecho de que Los paquetes son reenviados por una colección de conmutadores de software (por ejemplo, Open vSwitch) que comparten recursos de CPU y memoria y normalmente tienen un rendimiento inferior al hardware de conmutación dedicado.

Para los experimentos con CPU limitada, también necesitará asegurarse de que limita cuidadosamente el ancho de banda de la CPU de los hosts de Mininet. Si se

preocupa principalmente por la corrección funcional, puede ejecutar Mininet sin límites de ancho de banda específicos. Esta es la forma rápida y sencilla de ejecutar Mininet y también proporciona el rendimiento más alto a expensas de la precisión de sincronización bajo carga.

Con algunas excepciones, muchas limitaciones que usted puede encontrar no serán intrínsecas a Mininet; Eliminarlos puede ser simplemente una cuestión de código, y se le anima a contribuir con las mejoras que pueda desarrollar. [28] (Alcalde, 2015)

3.3.8 Cbench

Cbench (controlador benchmarker) es un programa para probar los controladores OpenFlow mediante la generación de paquetes en los eventos para los nuevos flujos. Cbench emula un montón de interruptores que se conectan a un controlador, enviar paquetes de mensajes, y esté pendiente de flujo-mods para quedarse relegados hacia abajo. Si usted cree que un cambio que está haciendo puede tener un impacto en el rendimiento, puede ser útil usar cbench para medirlo. [28] (Atlassian, 2012)

CAPÍTULO IV

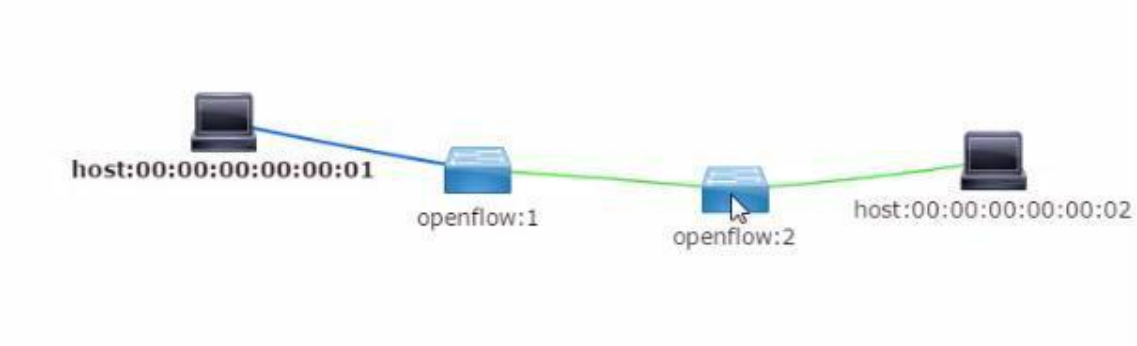
4. DESARROLLO DE LA PROPUESTA

4.1 Escenario y Pruebas iniciales

4.1.1 SDN con OpenDaylight

Lo primero que se realizó son las pruebas iniciales para esto se utilizó, las siguientes herramientas Virtual Box 5.0.2 con la distribución SDNHUB con Ubuntu 14.04 la versión de Mininet 2.2.1 donde se pueden crear y ejecutar ejemplos de topologías OPENDAYLIGHT como se muestra en la siguiente figura 5.

Figura 5: Topología en OPENDAYLIGHT



Para esto ejecutamos el primer comando donde nos dirigimos al directorio del SDN tutorial que es la distribución con la que estamos trabajando.

ubuntu@sdnhubvm:~$ cd SDNHub_Opendaylight_Tutorial

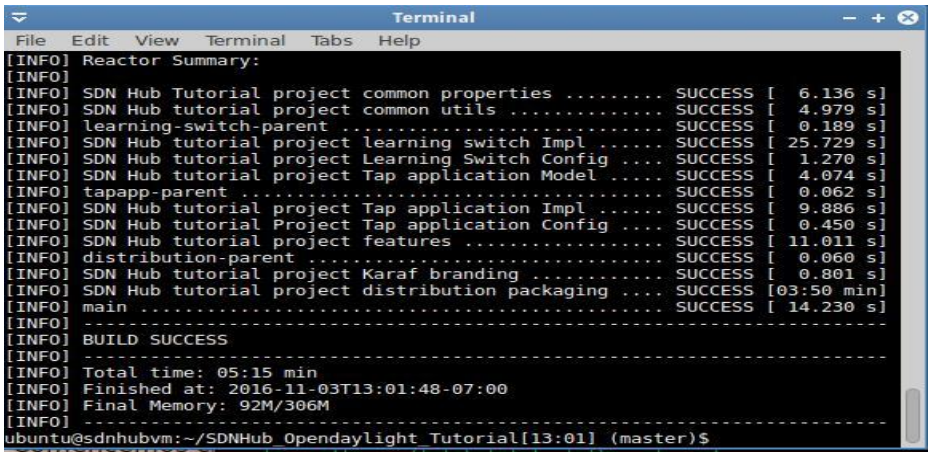
ubuntu@sdnhubvm:~$ git pull –rebase

Construcción del proyecto Maven

Algunos conocimientos básicos de maven son esenciales para trabajar con OpenDaylight que es la otra herramienta que vamos a trabajar.

ubuntu@sdnhubvm:~/SDNHub_Opendaylight_Tutorial$ mvn install –nsu

Figura 6: Construcción del proyecto OpenDaylight.



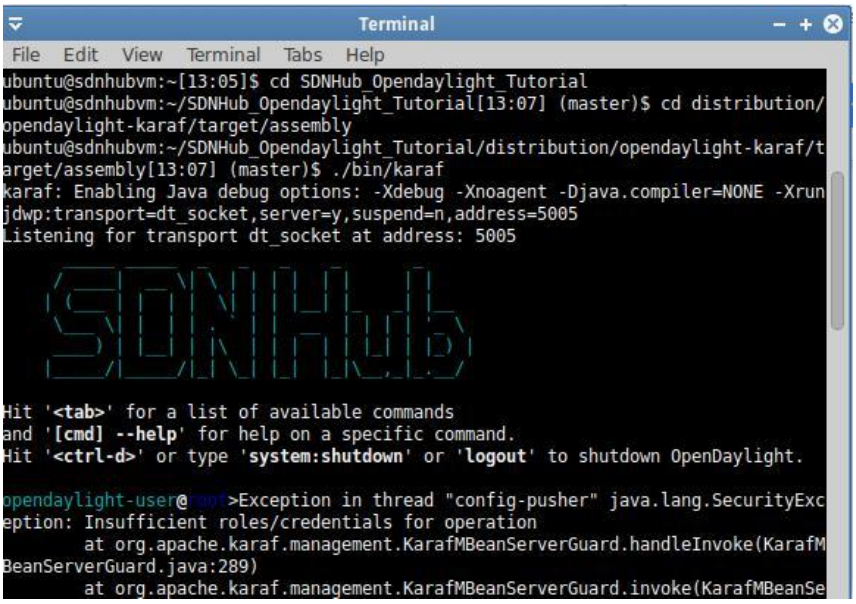
Creación de Funciones en Karaf

Ahora que ha compilado el proyecto de ejemplo, vamos a ejecutar el controlador en sí; Preferiblemente en un terminal diferente con las siguientes líneas tome en cuenta que se encuentre dentro del directorio de SDNHUB el resultado se muestra en la figura 7.

ubuntu@sdnhubvm:~/SDNHub_Opendaylight_Tutorial$ cd distribution/opendaylight-karaf/target/assembly

ubuntu@sdnhubvm:~/SDNHub_Opendaylight_Tutorial/distribution/opendaylight-karaf/target/assembly$ ./bin/karaf

Figura #7: Construcción del proyecto.



Ejecutar karaf inicia todos los paquetes Java instalados como archivos java en el entorno OSGi. Una vez iniciados todos los bundles, se llamará al método createInstance () en cada clase de implementación y el controlador se moverá a un estado de operación impulsado por eventos.

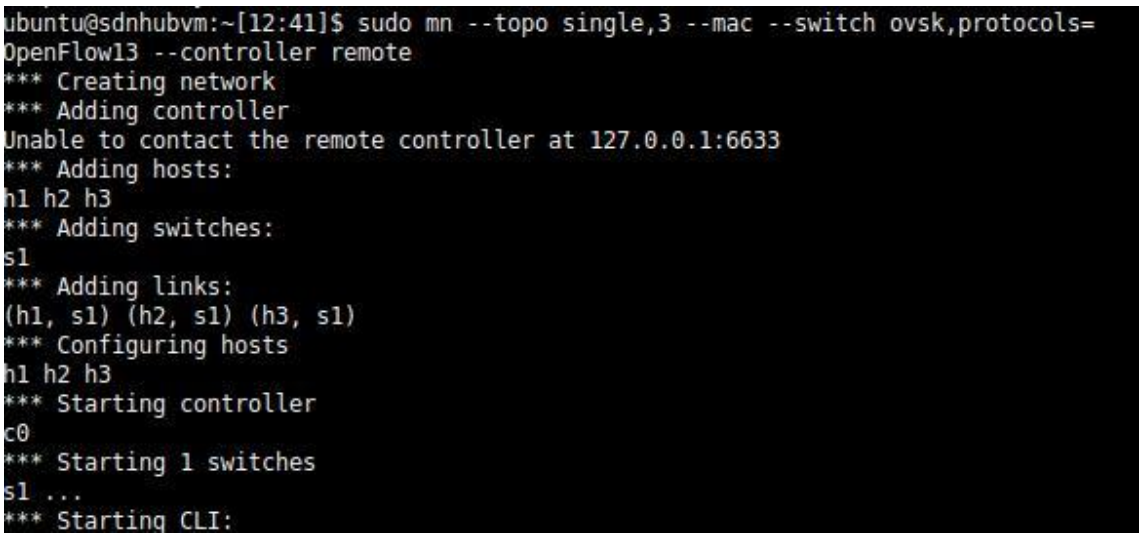
El karaf shell es el portal principal para administrar todas las aplicaciones y los paquetes Java.

Mininet

Es un software de emulación de red que funciona con Open vSwitch para crear un conjunto de conmutadores OpenFlow conectados a hosts virtuales e interconectados en diferentes topologías ejecutamos el siguiente comando

Usuario: mininet Clave: mininet ubuntu@sdnhubvm:~$ sudo mn --topo single,3 --mac --switch ovsk,protocols=OpenFlow13 --controller remote

Figura 8: Uso de Mininet

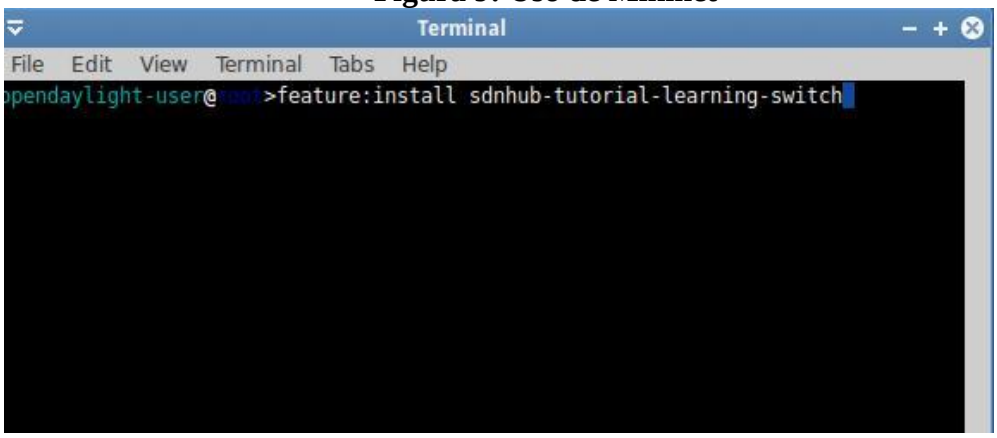


Con esta línea se inician 3 hosts y 1 switch. El conmutador también intenta conectarse a un controlador remoto. Desde que comenzamos OpenDaylight, también deberías ver varias líneas escritas en la consola karaf una vez que el switch se conecta a ella figura 9.

Puede iniciar un ping entre dos de los hosts. Sin embargo, el ping fallará porque no hay inteligencia en el conmutador para aprender las direcciones MAC de cada host y reenviar el tráfico a los puertos de conmutador correctos para estos debemos de instalar lo siguiente mininet> h1 ping h2

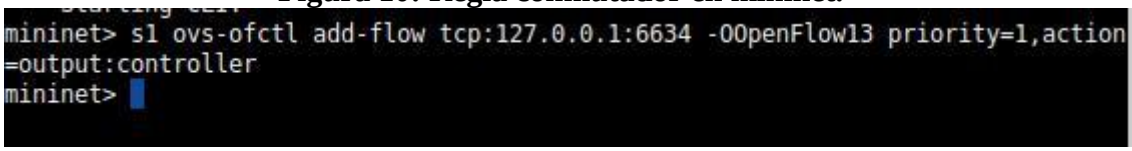
Agregamos esta inteligencia instalando la característica sdnhub-tutorial-learning-switch. Se dará cuenta de que después de hacer esto el ping comenzará a tener éxito.

Figura 9: Uso de Mininet



Luego de esto agrega una regla en el conmutador para enviar mensajes de paquete en el controlador como se muestra en la figura 10.

Figura 10: Regla conmutador en mininet.



mininet> s1 ovs-ofctl add-flow tcp:127.0.0.1:6634 -OOpenFlow13

priority=1,action=output:controller

mininet> h1 ping h2

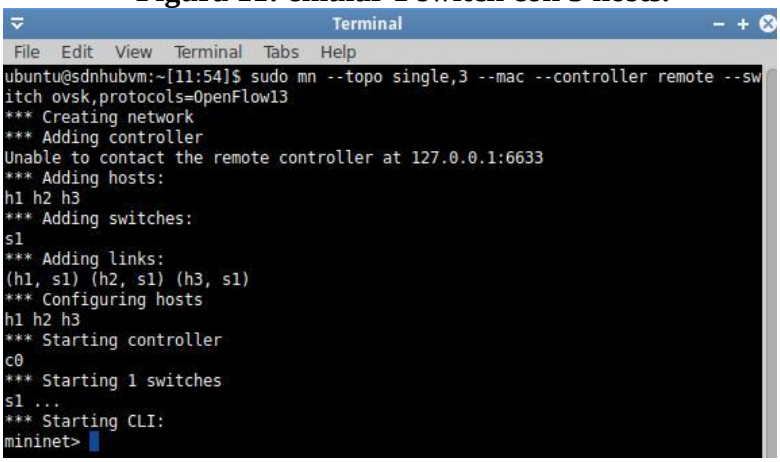
64 bytes from 10.0.0.2: icmp_req=1 ttl=64 time=94.1 ms 64 bytes from 10.0.0.2: icmp_req=2 ttl=64 time=21.2 ms 64 bytes from 10.0.0.2: icmp_req=3 ttl=64 time=21.9 ms.

4.2.2 OpenFlow y SDN

Para la siguiente práctica se utilizó Openflow v1.3 soporte para túneles, contadores de tráfico por flujo, puente de backbone de proveedor.

Para esto se ejecuta una ventana de terminal de mininet con el siguiente comando. Esto inicia un entorno de emulación de red para emular 1 switch con 3 hosts. sudo mn --topo single,3 --mac --controller remote --switch ovsk,protocols=OpenFlow13.

Figura 11: emular 1 switch con 3 hosts.



Este comando funciona en esta distribución SDNHUB con Ubuntu 14.04 para otras distribuciones se usa el siguiente comando.

ovs-vsctl set bridge s1 protocols=OpenFlow13

Luego de esto se inicializa el wireshark para analizar los mensajes OpenFlow sudo wireshark & figura 12.

Figura 12: Ejecución de Wireshark

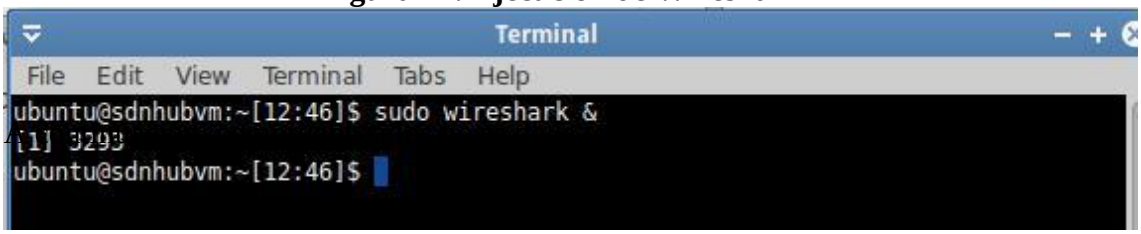
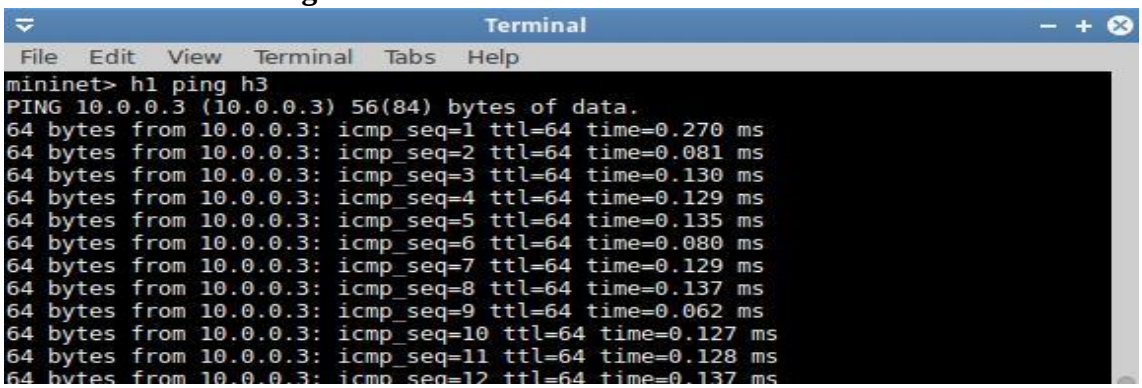
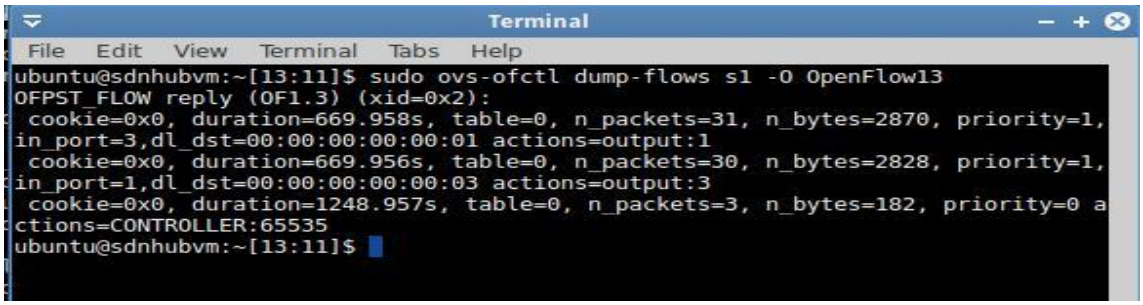


Figura 13: Prueba de conectividad mininet



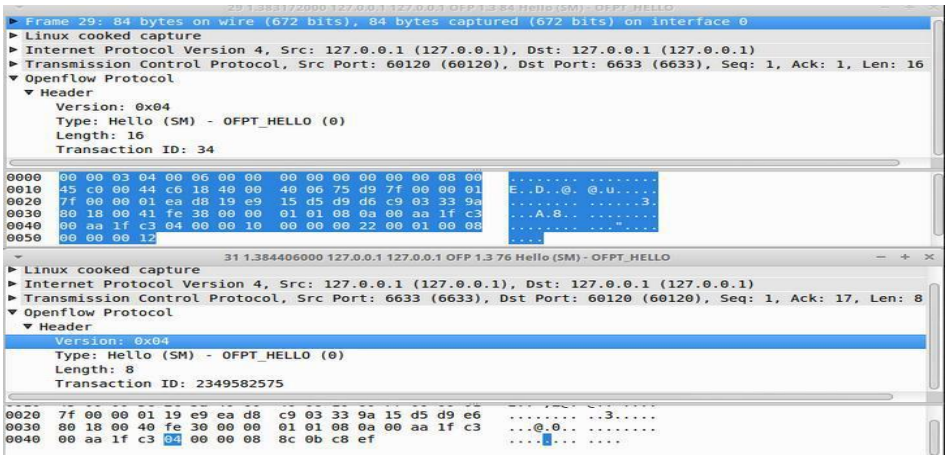
Ahora puede listar los flujos en curso utilizando el siguiente comando específico de OpenFlow 1.3:

Figura 14: Listado de flujos de openflow



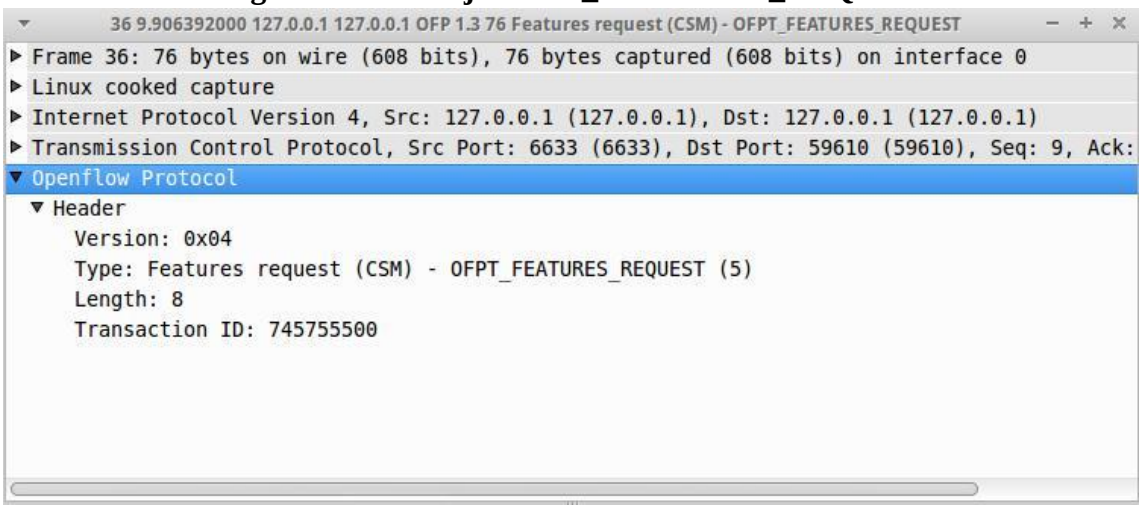
El conmutador inicia una conexión TCP estándar (o TLS) al controlador. Cuando se establece una conexión OpenFlow, cada entidad debe enviar un mensaje OFPT_HELLO con la versión de protocolo establecida en la versión de protocolo OpenFlow más alta admitida por el remitente. En la siguiente figura, podemos ver que OpenFlow versión 1.3 ha sido negociado. El conmutador inicia una conexión TCP estándar (o TLS) al controlador. Cuando se establece una conexión OpenFlow, cada entidad debe enviar un mensaje OFPT_HELLO con la versión de protocolo establecida en la versión de protocolo OpenFlow más alta admitida por el remitente. En la siguiente figura, se observa que el OpenFlow versión 1.3 ha sido negociado como se muestra en la figura 15.

Figura 15: Negociado del protocolo OPENFLOW.



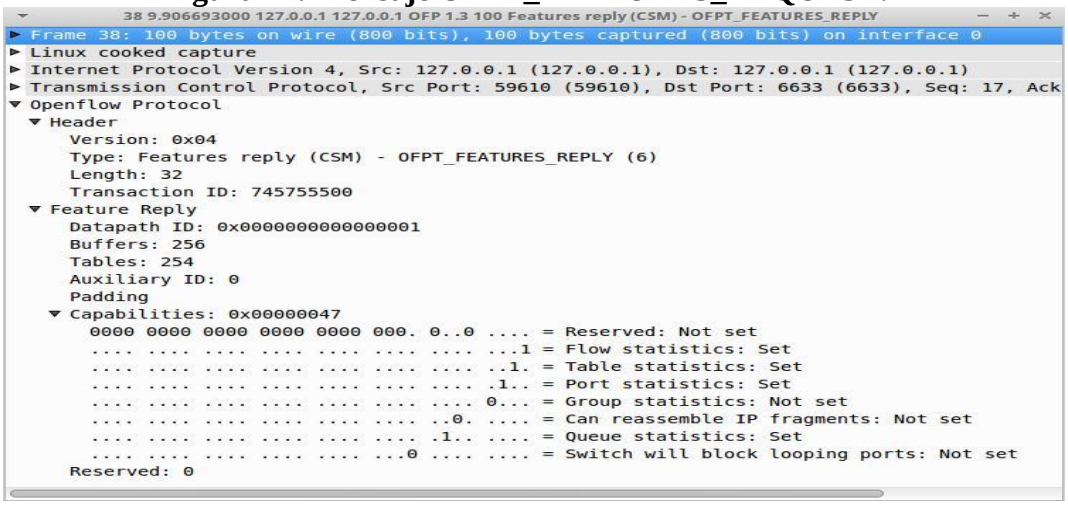
Después de establecer satisfactoriamente una sesión, el controlador envía un mensaje OFPT_FEATURES_REQUEST. Este mensaje sólo contiene un encabezado de OpenFlow y no contiene un cuerpo.

Figura 16: Mensaje OFPT_FEATURES_REQUEST.



El modificador responde con un mensaje OFPT_FEATURES_REPLY. Observe el Datapath ID y las capacidades de conmutador enviadas como parte del mensaje de respuesta de Feature.

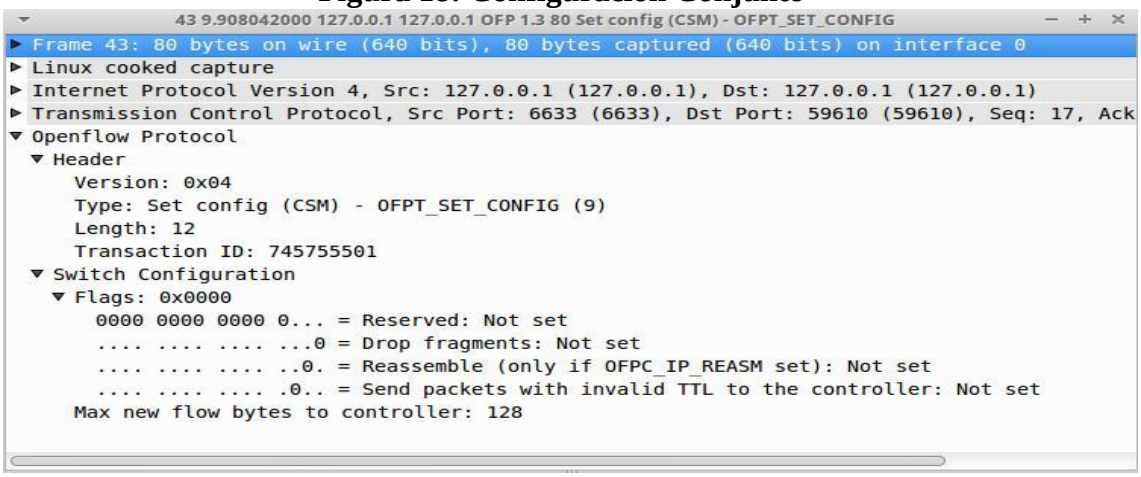
Figura 17: Mensaje OFPT_FEATURES_REQUEST.



Configuración del conjunto

A continuación, el controlador envía el mensaje OFPT_SET_CONFIG al conmutador. Esto incluye el conjunto de banderas y Max bytes de paquete que debe enviar el controlador de datos al controlador.

Figura 18: Configuración Conjunto

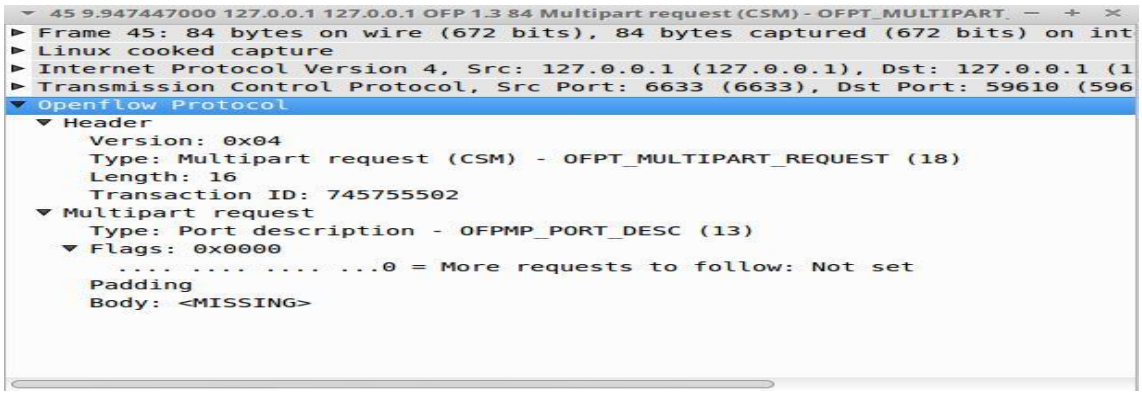


Solicitud de varias partes – Respuestas

El controlador puede solicitar el estado desde el datapath utilizando el mensaje OFPT_MULTIPART_REQUEST. Los tipos de mensaje manejados por este mensaje incluyen varias estadísticas (FLOW / TABLE / PORT / QUEUE / METER, etc.) o características de descripción (METER_CONFIG / TABLE_FEATURES /

PORT_DESC etc). En nuestro simple_switch_13.py, RYU envía internamente un MULTIPART_REQUEST para solicitar la descripción del puerto.

Figura 19 Solicitud de varias respuestas

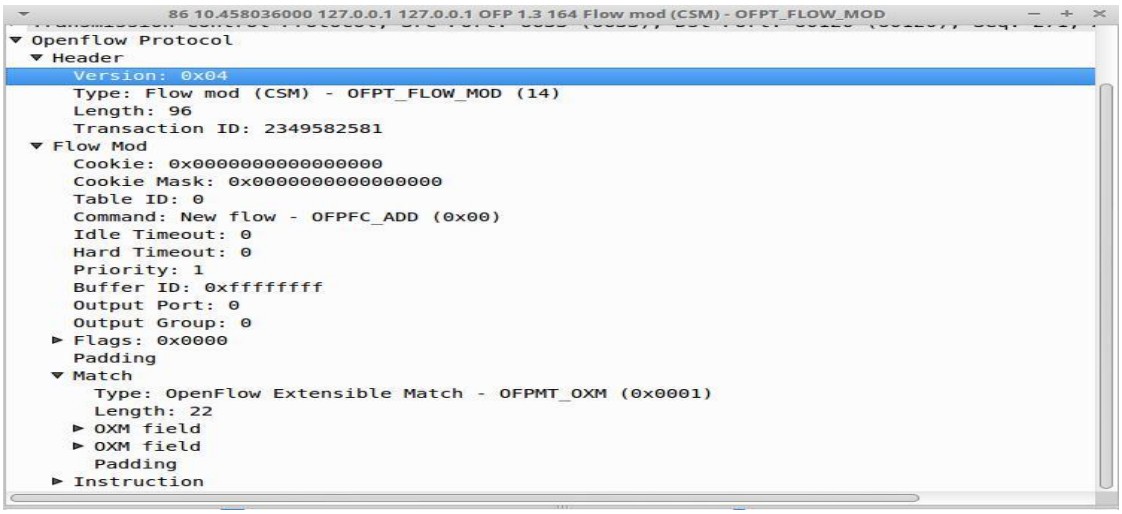


Modos flujo

Los flujos pueden ser proactivos (por ejemplo, flujos de preinstalación como TableMissFlow) o reactivamente (por ejemplo, reaccionar para mensajes packet_in) enviados desde el controlador. Los mensajes de modificación de la tabla de flujo pueden tener los siguientes tipos: OFPFC_ADD, OFPFC_DELETE, OFPFC_DELETE_STRICT, OFPFC_MODIFY, OFPFC_MODIFY_STRICT.

En el caso siguiente, el controlador instala un nuevo flujo, lo que muestra que aparte del conjunto de parámetros de OF 1.0 como prioridad, idle_timeout, etc., la estructura de coincidencia e instrucción refleja los nuevos parámetros especificados en 1.3.

Figura 20: Modos de flujos



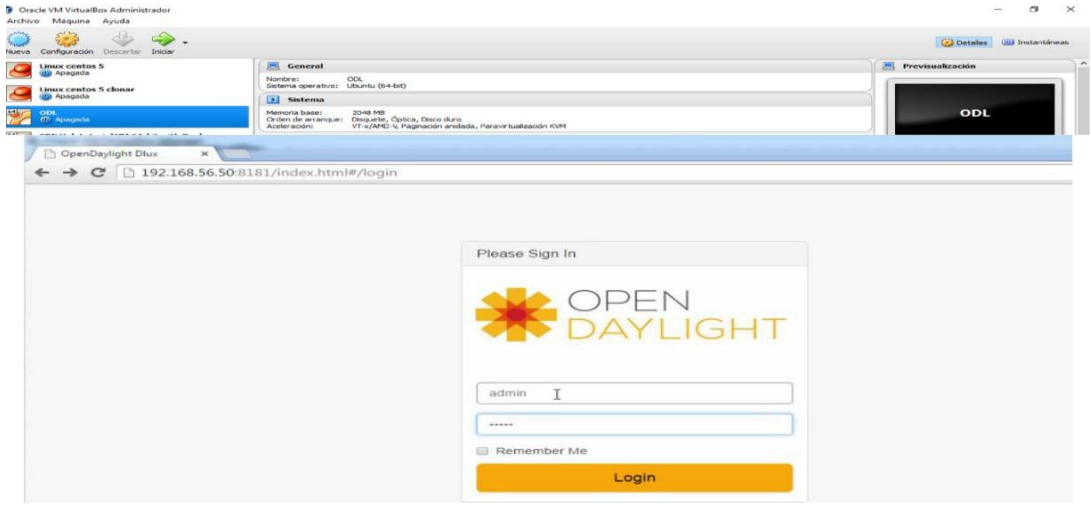
4.3 Escenario practico de OPENDAYLIGHT con el protocolo Openflow

4.3.1 Escenario practico de OPENDAYLIGHT diagrama autoridades

Lo primero que se realizo fue descargar OPENDAYLIGHT se instaló en la máquina virtual como se muestra en la siguiente figura 21 el usuario y clave por defecto son:

Usuario: admin Clave: admin.

Figura 21: Acceso a OPENDAYLIGHT



Para esta práctica se realizó con 4 switch y 4 host para esto se utilizó la siguiente línea en mininet tomando en cuenta que en la Facultad de Ciencias Informáticas contamos 4 switch para esta vamos a usar 4 host cada uno en uno de los switch.

$ sudo mn –controller=remote, ip=192.168.56.50 –topo=linear,4 –switch=ovsk, protocols=Openflow13 –mac

Figura 22: Creación de 4 switch

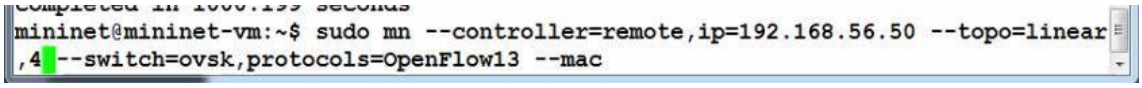
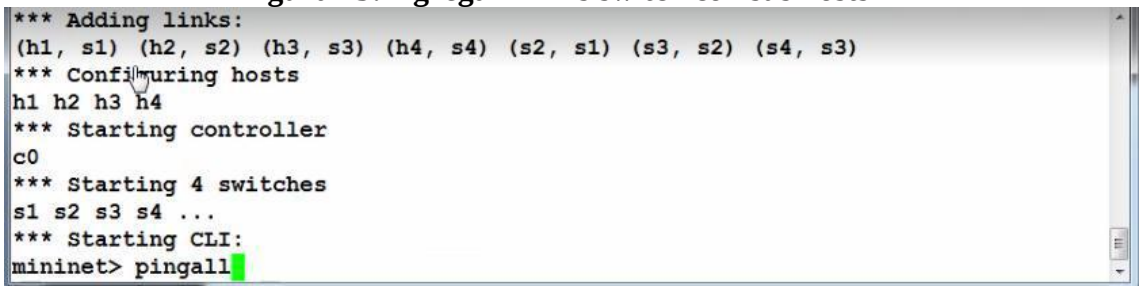


Figura 23: Agregar Links switch con sus hosts



Para para revisar que la conexión sea exitosa solamente entre los host y los switch que se ha establecido se realizan un Pingall el cual permite realizar todas las formas de conectividad entre los host tal como muestra en la siguiente figura 24.

Figura 24: Grafico de la red con 4 switch en OPENDAYLIGHT

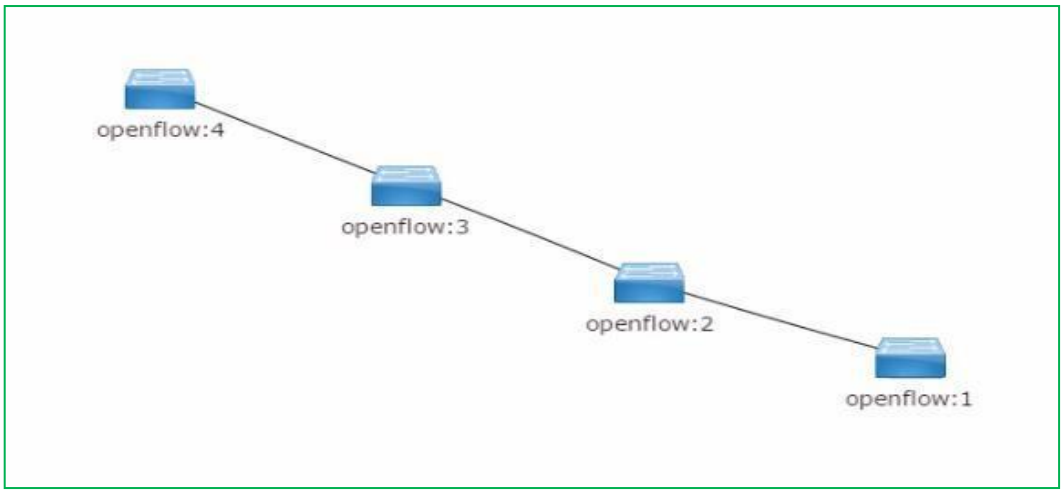
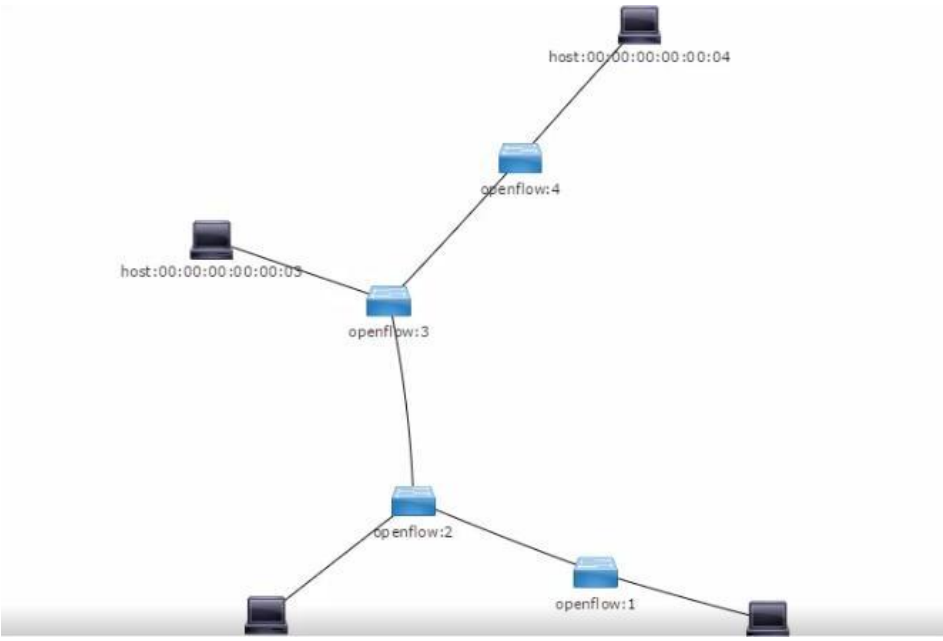


Figura 25: Verificar conexión con Pingall

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> X h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 8% dropped (11/12 received)
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
mininet>
```

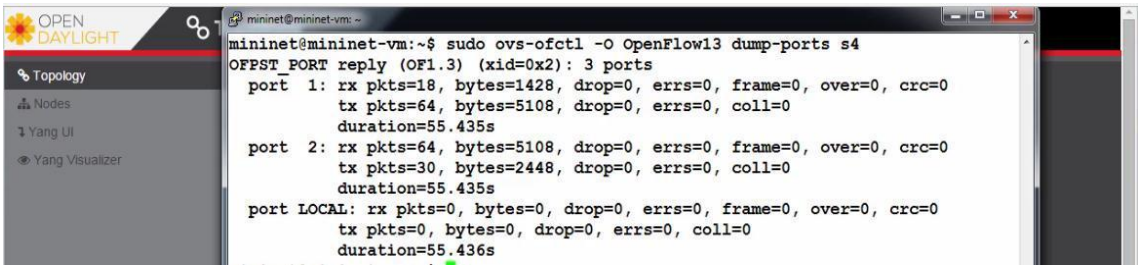
```
mininet@mininet-vm:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s4
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x2b0000000000000d, duration=67.579s, table=0, n_packets=18, n_bytes=142
8, priority=2,in_port=1 actions=output:2,CONTROLLER:65535
 cookie=0x2b0000000000000c, duration=67.579s, table=0, n_packets=52, n_bytes=408
8, priority=2,in_port=2 actions=output:1
 cookie=0x2b00000000000009, duration=73.525s, table=0, n_packets=16, n_bytes=136
0, priority=100,dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x2b00000000000009, duration=73.525s, table=0, n_packets=0, n_bytes=0, p
riority=0 actions=drop
mininet@mininet-vm:~$
```

Figura 26: Grafico final con OPENDAYLIGHT



Para mostrar la tabla de flujos sudo ovs-ofctl –o OpenFlow13 dump-ports s4

Figura 27: Tabla de flujos en el switch4



Luego de esto se realizó otra topología con 10 hosts usando uno de los switch antes expuesto tomando en cuenta el switch que tiene las autoridades de la facultad el cual cuenta con 10 computadoras, se ejecuta la línea de comandos pingall y reload al grafico como se muestra en las siguiente figuras 28 y 29.

Figura 28: Creación 10 host topología single.

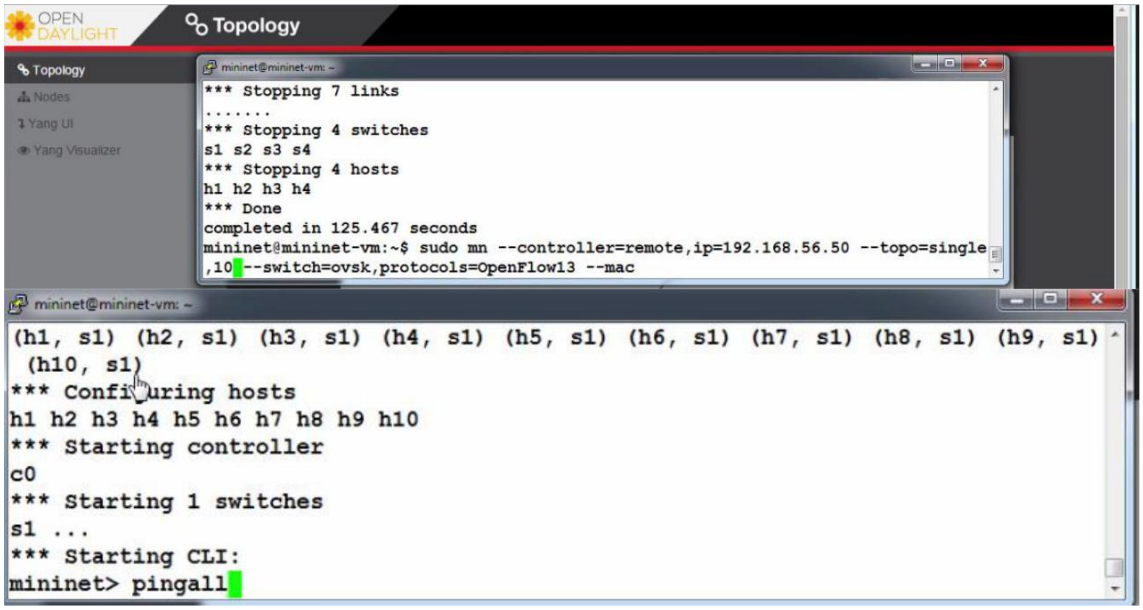
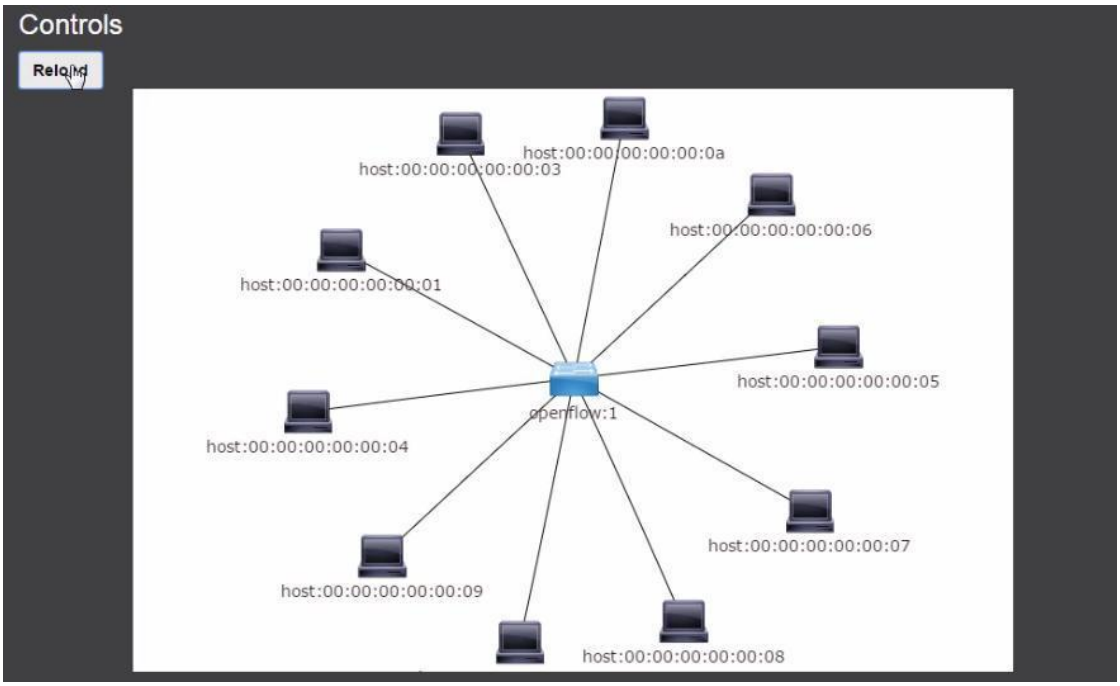


Figura 29: Topología de red de Autoridades FCI



Se ejecuta el comando para mostrar la salida de las interfaces como se muestra en la siguiente figura 30. Sudo ovs-vsctl show

Figura 30: Mostramos la salida de las interfaces

```
mininet@mininet-vm:~$ sudo ovs-vsctl show
0b8ed0aa-67ac-4405-af13-70249a7e8a96
    Bridge "s1"
        Controller "ptcp:6634"
        Controller "tcp:192.168.56.50:6633"
            is_connected: true
        fail_mode: secure
        Port "s1-eth6"
            Interface "s1-eth6"
        Port "s1-eth1"
            Interface "s1-eth1"
        Port "s1"
            Interface "s1"
                type: internal
        Port "s1-eth4"
            Interface "s1-eth4"
        Port "s1-eth2"
            Interface "s1-eth2"
        Port "s1-eth7"
            Interface "s1-eth7"
        Port "s1-eth9"
            Interface "s1-eth9"
        Port "s1-eth5"
            Interface "s1-eth5"
        Port "s1-eth3"
            Interface "s1-eth3"
```

4.3.2 Escenario practico de OPENDAYLIGHT con topologías

Topología linear en mininet

Un switch conectado a N hosts. Se crea con el siguiente comando: Sintaxis

$sudo mn --topo single,N

N es el número de hosts deseados.

Para esto se realiza lo siguientes: Se crearán 4 switch cada uno con hosts conectado. En primer lugar se ejecutará el siguiente comando

$ sudo mn –controller=remote, 192.168.56.50 --mac --topo=linear,4

Con esta línea se agregará controller, hosts, switches y se probará la conectividad en los nodos con pingall figura 31.

Figura 31: Creamos la topología linear

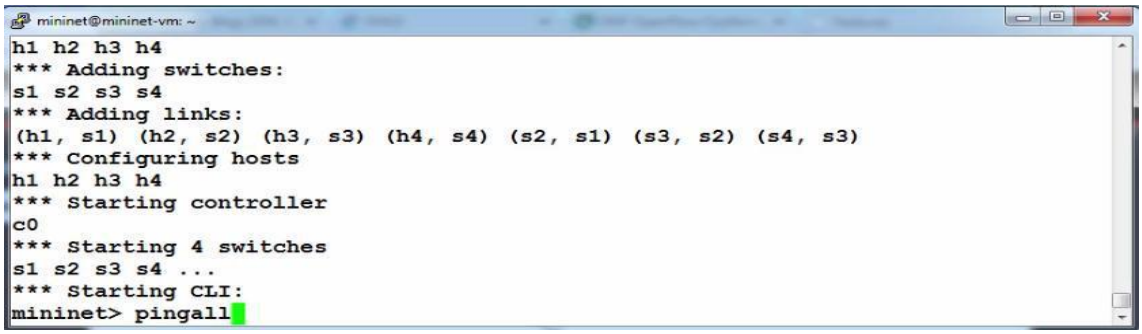
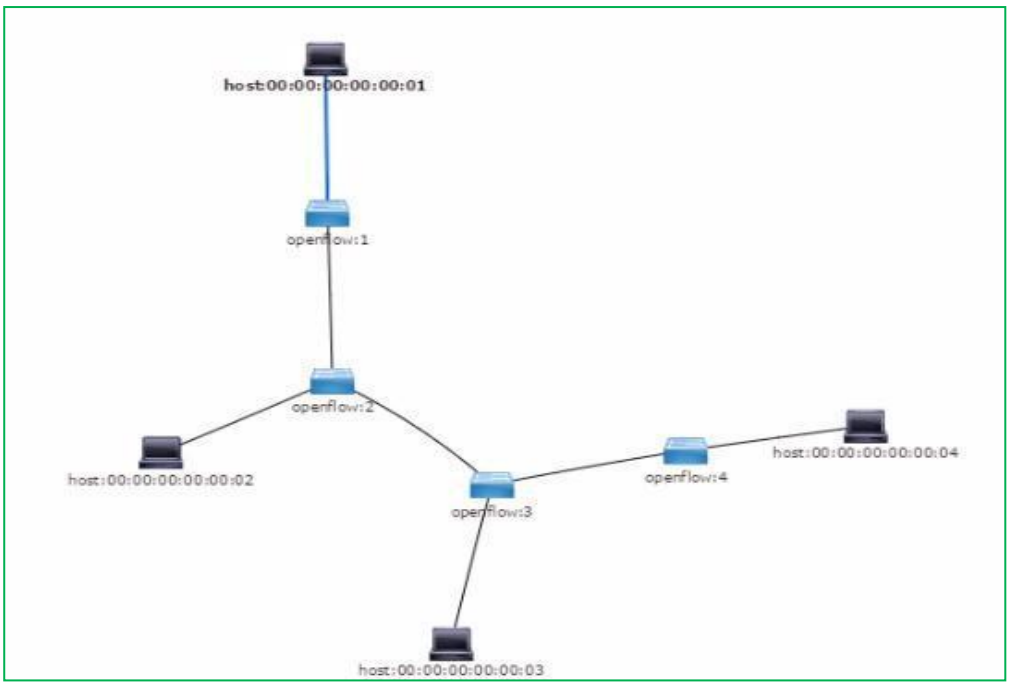


Figura 32: Topología linear opendaylight mininet



Topología Tree en mininet

Tree Crea una topología de árbol con profundidad N y anchura M. Sintaxis

$sudo mn --topo tree,depth=n, fanout=m

Para esta práctica se ejecuta la siguiente línea de comando con profundidad de 2 y anchura de 3 el resultado figura 33.

sudo mn –controller=remote, ip=192.168.56.50 –mac –topo=tree, depth=2, fanout=3

Después de ejecutar esta línea ejecutamos pingall para verificar la conectividad.

Figura 33: Topología Tree opendaylight mininet

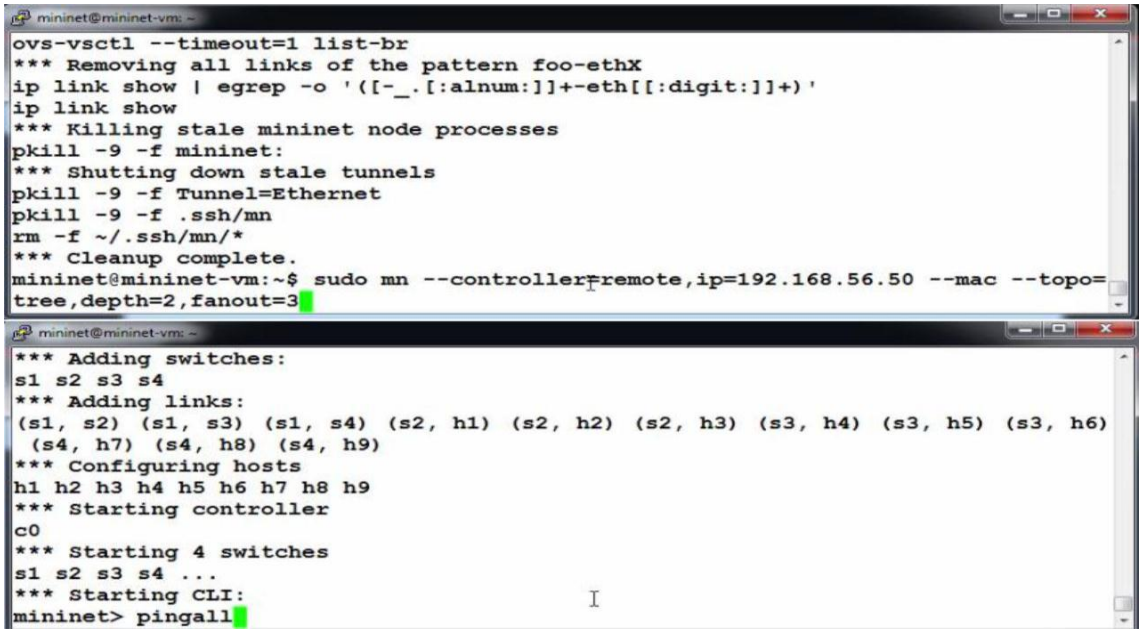
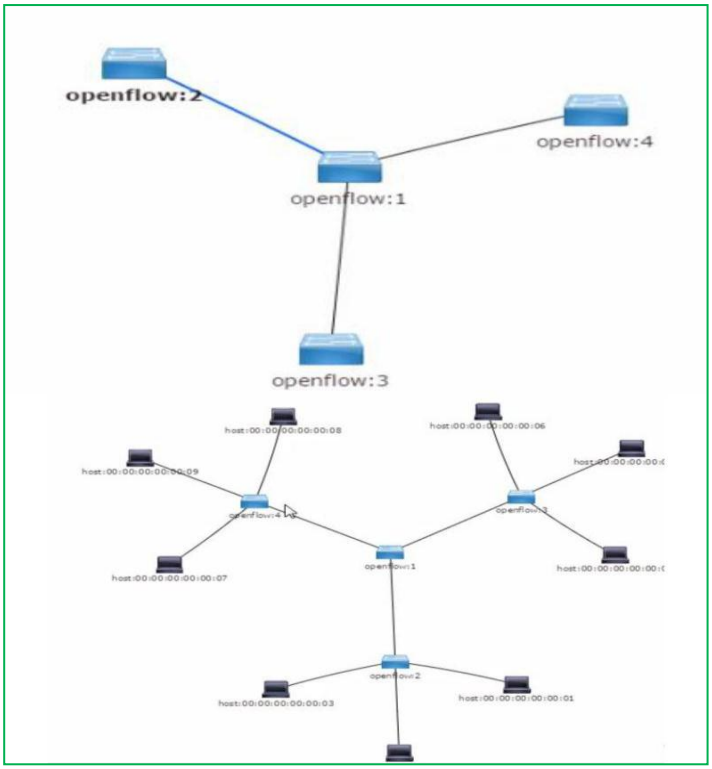


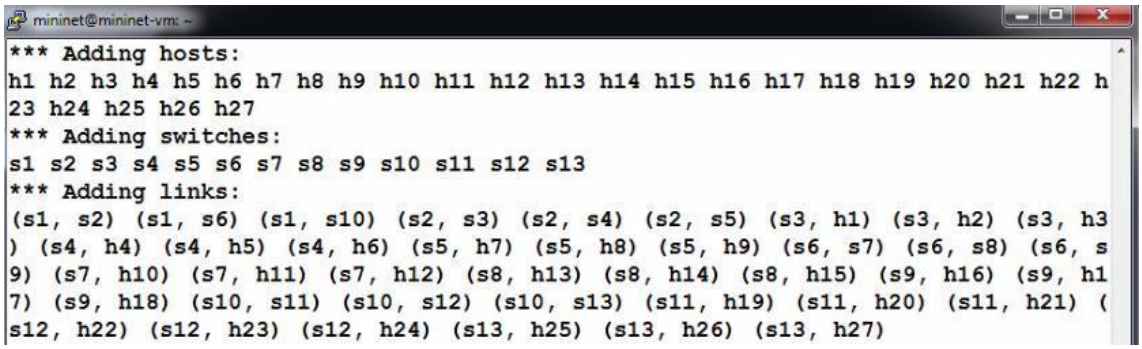
Figura 34: Diagrama de Topología Tree opendaylight mininet



Se realizó un cambio en la práctica y se ejecutó la siguiente línea de comando con profundidad de 3 y anchura de 3 el resultado sudo mn –controller=remote, ip=192.168.56.50 –mac –topo=tree, depth=3, fanout=3 Donde agregamos 13 switches y 27 hosts.

Después de ejecutar esta línea ejecutamos pingall para verificar la conectividad.

Figura 35: Topología Tree opendaylight mininet



Una vez ejecutado los comandos se observa que el diagrama queda de la siguiente manera: Se posee 13 switchs y 27 hosts repartidos 3 por cada switchs.

Figura 36: Diagrama de Topología Tree opendaylight con switches

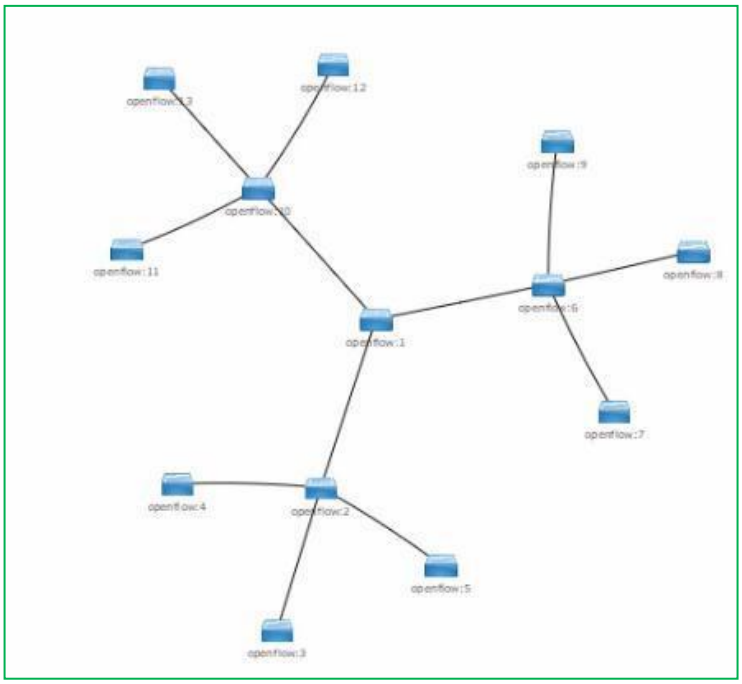
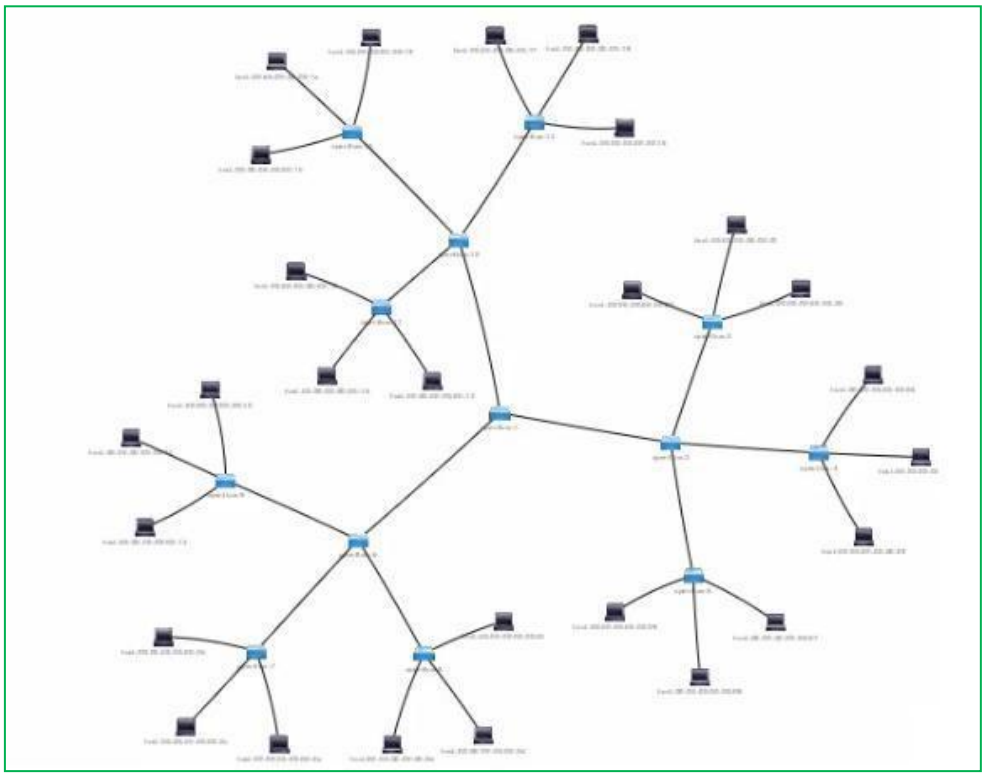


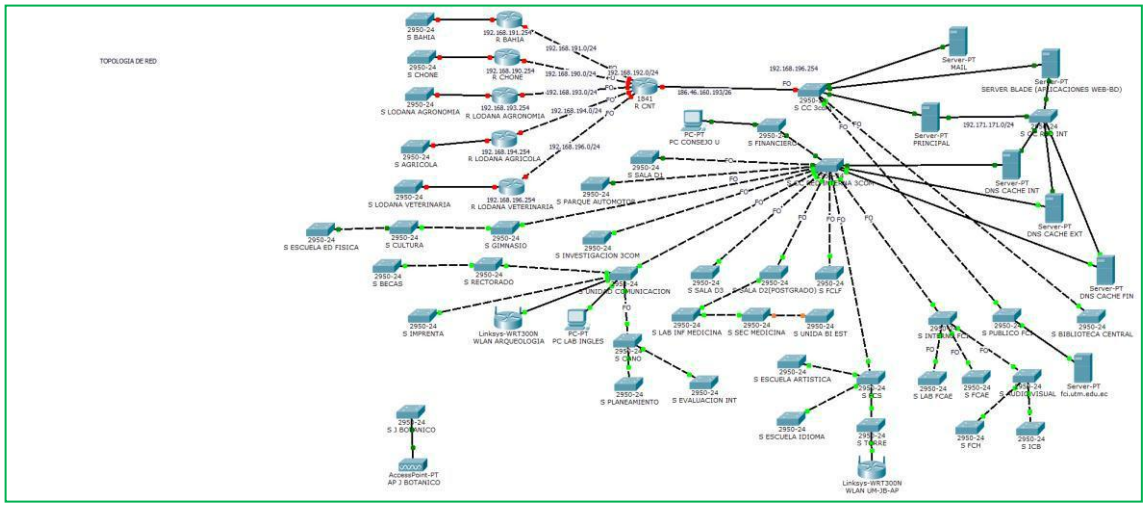
Figura 37: Diagrama de Topología Tree opendaylight con hosts



4.5 Diagrama de redes de comunicaciones de la UTM

De acuerdo al siguiente diagrama se observa que las topologías aplicada single, tree se pueden aplicar a el diseño de la red, el problema que se encuentra es que los equipos de comunicación no soporta SDN y Openflow pero el siguiente punto de esta tesis se describe los equipos que se pueden adquirir para que se logre implementar esta tecnología en la UTM.

Figura 38: Diagrama de redes de comunicaciones UTM



CAPÍTULO V

5. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

Una vez finalizado el trabajo se concluye:

Luego de una exhaustiva investigación se logró conocer a cabalidad el protocolo Openflow, comprobando los resultados de la aplicación en la herramienta SDN, con la utilización de dicha arquitectura permitirá el mejoramiento de las redes en la comunidad universitaria, asegurando así una comunicación eficaz entre las facultades y departamentos administrativos.

La realización del estudio efectuado con la arquitectura del SDN permitió que los dispositivos conectados mediante la red, controlan remotamente las tablas de reenvió en una red de switches y routers, permitiendo así automatizar y controlar las redes altamente escalables y flexibles, que se adaptan rápidamente a los requerimientos de la institución.

SDN y Openflow representa una revolución en el mundo de las redes de comunicaciones, como vemos en esta investigaciones las empresas que han implementado Google, Cisco, Huawei, Hp, empiezan a confiar cada día más en estas nuevas tecnologías porque son el futuro de las redes al permitir aprovechar mejor los recursos de las tecnologías actuales.

Finalmente se ha probado el software de emulación Opendaylight y Mininet siendo una herramienta orientada a demostrar de manera sencilla y educativa las

principales características de OpenFlow y el modo en el que interactúan sus diferentes elementos de red, sobre todo al controlador en este estudio, se realizaron tres escenarios similares a la redes de comunicaciones de la universidad, con lo que se demostró que estas herramientas pueden facilitar una mejor administración de nuestra red.

5.2 Recomendaciones

De acuerdo a las conclusiones, se recomienda:

Empezar usando las herramientas Mininet y Opendaylight para estudiar las redes definidas por software, porque estas herramientas permiten realizar la programación de la red accediendo a programar los equipos de acuerdo a las necesidades, familiarizándose con conceptos y procedimientos propios del protocolo OpenFlow y de SDN, contando con una visualización gráfica.

Dentro de las lecciones aprendidas que ha dejado este estudio, sería conveniente una investigación más detallada del simulador virtual utilizado el SDN que es Mininet, para utilizar de mejor manera sus componentes y su aplicación dentro del centro de cómputo.

Que la Universidad Técnica de Manabí adquiera equipos que soporten Redes Definidas por Software con el protocolo Openflow, esto permitirá poder realizar una pequeña implementación eligiendo una área de red menos crítica en el campo universitario y con los resultados obtenidos y las lecciones aprendidas alcanzar la madurez necesaria para poder implementarla en toda la UTM.

El personal de TICS de la Universidad Técnica de Manabí se les debe capacitar en el uso de Redes Definida por Software con las herramientas de Mininet y

Opendaylight lo que permitirá que la utilización de estas tecnologías tenga el éxito esperado.

BIBLIOGRAFÍA

[01] Cuella, J. C. (2014). Implementación de Open Flow sobre NetEPGA. http://www.researchgate.net/publication/270588431_Implementacin_de_Open _ Flow_sobre_NetFPGA

[02] Barona López, L. (2013). Propuesta de escenarios virtuales con la herramienta VNX para pruebas del protocolo Openflow. España, Madrid, España. http://www.dit.upm.es/~posgrado/doc/TFM/TFMs2012-2013/TFM_Lorena_Barona_2013.pdf.

[03] Blandón Gómez, D. (2013). Openflow: El protocolo del futuro. p.61. http://biblioteca.ucp.edu.co/ojs/index.php/paginas/article/view/255

[04] Moreno Galué, V. A. (2012). Estudio del estandar Openflow mediante caso práctico utilización con la herramienta de VNX. España. http://www.dit.upm.es/~posgrado/doc/TFM/TFMs2011-2012/TFM_Vanessa_Moreno_2012.pdf.

[05] Millán Tejedor, R. J. (2014). SDN: el futuro de las redes inteligentes. Recuperado el 20 de 07 de 2015, de http://www.ramonmillan.com/tutoriales/sdnredesinteligentes.php

[06] Figuerola, Norberto. (2013). SDN – Redes definidas por Software https://articulosit.files.wordpress.com/2013/10/sdn.pdf

[07] Andrés, M. (14 de 04 de 2014). Universidad Sur Colombiana. Recuperado el 30 – 12 de 2015, de http://journalusco.edu.co/index.php/IngenieriayRegion/article/view/384/1383

[08] Fundación Telefónica. (2015). TELOS 100: La era digital: balance y tendencias Madrid.

[09] Velásquez Vargas, W. (2015). http://www.academia.edu. academia.edu, 7.

[10] Pantoja, L. (04 de 12 de 2013). www.gluc.unicauca.edu.com. Recuperado el 04 de 01 de 2016, de

http://gluc.unicauca.edu.co/index.php?title=Maquinas_Virtuales_y_la_Virtualizacion&action=edit

[11] OPENFLOW. (2011). www.openflow.org. Recuperado el 04 de 01 de 2016,

[12] Plaza, Crisanto. (2015). Ensayo sobre la regulación tecnológica: La era digital en Europa. Taurus. España

[13] Openvswitch. (16 de 11 de 2015). Openvswitch. Recuperado el 30 de 12 de 2015, de http://openvswitch.org/

[14] Serrano Carrera, David Andrés. (2015). Trabajo fin de grado. Redes Definidas por Software (SDN): OpenFlow. Universitat politécnica de Valencia. https://riunet.upv.es/bitstream/handle/10251/62801/SERRANO%20-%20Redes%20Definidas%20por%20Software%20(SDN)%3A%20OpenFlow.pdf?sequence=3

[15] Escobar Ordoñez, Jaime Andrés. (2015). Control de flujo de una red definida por software usando sensores térmicos. Tesis de grado. Universidad Técnica de Ambato. repo.uta.edu.ec/bitstream/123456789/15083/1/Tesis_t1065ec.pdf

[16] Fernández Romero, Y., & García Pom, K. (03 de 11 de 2011). revista telemática. Recuperado el 04 de 01 de 2016, de http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/33

[17] Virtualbox. (2015). Obtenido de https://www.virtualbox.org/

[18] Vmware. (2015). vmware. Recuperado el 30 de 12 de 2015, de https://www.vmware.com/products/workstation

[19] Red Hat Enterprise, L. (2005). Red Hat Enterprise Linux 4: Manual de referencia. Recuperado el 30 de 12 de 2015, de http://www.gb.nrao.edu/pubcomputing/redhatELWS4/RH-DOCS/rhel-rg-es-4/ch-ssh.html

[20] Thomas E., D. (2014). http://invisible-island.net. Recuperado el 03 de 01 de 2016, de http://invisible-island.net/xterm/

[21] Github. (2013). Dpctl Documentation. Recuperado el 03 de 01 de 2016, de https://github.com/CPqD/ofsoftswitch13/wiki/Dpctl-Documentation

[22] Wireshark. (2014). wireshark. Recuperado el 03 de 01 de 2016, de https://www.wireshark.org/

[23] Mininet. (2015). mininet. Recuperado el 03 de 01 de 2016, de http://mininet.org/

[24] Atlassian. (01 de 2012). atlassian. Recuperado el 03 de 01 de 2015, de https://floodlight.atlassian.net/wiki/pages/viewpage.action?pageId=1343657 [30]

[25] Serrano, C. (2016). Redes Definidas por Software (SDN): OpenFlow. España.

[26] Alcalde, L. (2015). https://translate.googleusercontent.com. Recuperado el 20 de 10 de 2016, de https://translate.googleusercontent.com

[27] Nokia. (2016). http://inpublic.globenewswire.com. Recuperado el 08 de 12 de 2016, de http://inpublic.globenewswire.com/releaseDetails.faces?rId=2048126

[28] Intel América. (2015). http://mundocontact.com. Recuperado el 08 de 12 de 2016, de http://mundocontact.com/tag/sdn/

[29] Cisco. (2015). http://www.cisco.com. Recuperado el 09 de 12 de 2016, de http://www.cisco.com/c/en/us/products/collateral/cloud-systems-management/open-sdn-controller/datasheet-c78-733458.html

[30] ONF, O. (2016). https://www.opennetworking.org. Recuperado el 9 de 12 de 2016, de https://www.opennetworking.org/sdn-openflow-products?start=20

[31] BROCADE. (2016). http://www.brocade.com/content/html/en/configuration-guide/nos-601-sdnguide/GUID-3CBDC5D1-C379-4CE2-9B5A 36DFDE4B57A4.html