UNIDAD ACADÉMICA:

DEPARTAMENTO DE INVESTIGACIÓN Y POSTGRADOS

TEMA:

DESARROLLO DE UN PROTOTIPO DE RED DEFINIDA POR SOFTWARE SDN PARA LA GESTIÓN MEDIANTE RECURSOS DE ESTÁNDAR ABIERTO

Proyecto de Investigación y Desarrollo de Grado previo a la obtención del título de

Magister en Gerencia Infomática

Línea de Investigación, Innovación y Desarrollo principal:

Redes y Aplicaciones

Caracterización técnica del trabajo:

Desarrollo

Autor:

Félix Mauricio Murillo Calderón

Director:

Ing. Balseca Manzano José Marcelo, Mg.

Ambato – Ecuador

Abril 2016

Desarrollo de un Prototipo de Red Definida por Software SDN para la Gestión mediante Recursos de Estándar Abierto

Informe de Trabajo de Titulación
presentado ante la
Pontificia Universidad Católica del Ecuador

Sede Ambato

por

Félix Mauricio Murillo Calderón

En cumplimiento parcial de
los requisitos para el Grado de
Magister en Gerencia
Informática

Departamento de Investigación y Postgrados
Abril 2016

Desarrollo de un Prototipo de Red Definida por Software SDN para la Gestión mediante Recursos de Estándar Abierto

Aprobado por:

Varna Hernández Junco, PhD
Presidente del Comité Calificador
Director DIP

Ing. Verónica Maribel Pailiacho Mena Mg.
Miembro Calificador

Ing. José Marcelo Balseca Manzano Mg.
Director de Proyecto

Dr. Hugo Altamirano Villaroel
Secretario General

Ing. Juan José Ramos Paredes Mg.
Miembro Calificador

Fecha de aprobación:
Abril, 2016

Ficha Técnica

Programa: Magister en Gerencia Informática

Tema: Desarrollo de un prototipo de red definida por software SDN para la gestión mediante recursos de estándar abierto.

Tipo de trabajo: Proyecto de Investigación y Desarrollo de Grado

Clasificación técnica del trabajo: Desarrollo

Autor: Félix Mauricio Murillo Calderón

Director: Ing. José Marcelo Balseca Manzano, Mg.

Líneas de Investigación, Innovación y Desarrollo

Principal: Ingeniería de Software y/o Plataformas Educativas

Secundaria: Redes y Aplicaciones

Resumen Ejecutivo

Este documento presenta el desarrollo del prototipo de red definida por software SDN utilizando para la gestión recursos de estándar abierto, en el que se ha desarrollado aplicaciones con controladores OpenFlow que se adhieran a cualquier tipo de infraestructura de red, buscando generar una mayor eficiencia y optimización de la capacidad de la red en el servicio que se brinda a los usuarios; para lo cual se ha utilizado los modelos ágiles XP en la que su principal herramienta es la disponibilidad de cambios a medida que sea necesario. El aplicativo de red SDN debe monitorear el tráfico de información dependiendo de las políticas establecidas por el administrador de red; adicionalmente la aplicación del prototipo de red presentó comportamientos con niveles de satisfacción aceptables en las pruebas iniciales de enrutamiento, monitoreo y de la reducción del número de paquetes que debe generar el servidor de manera que exista un flujo de información sin interrupciones. Esto permite que la infraestructura de red utilizando el protocolo OpenFlow en base a Floodlight se convierta en una red directamente programable, ágil, escalable, de gestión centralizada y con programación configurada mediante la utilización de Mininet para la creación de redes virtuales reales, lo que permite a los administradores optimizar recursos y mantener la simplicidad en los diseños, convirtiéndolas así en redes flexibles, seguras y eficientes.

Palabras claves: SDN, OpenFlow, Mininet, Floodlight, red, eficiente, ágil, infraestructura, administración de red.

Declaración de Originalidad y Responsabilidad

Yo, Félix Mauricio Murillo Calderón, portador de la cédula de ciudadanía y/o pasaporte No. 1802998409, declaro que los resultados obtenidos en el proyecto de titulación y presentados en el informe final, previo a la obtención del título de Magister en Gerencia Informática, son absolutamente originales y personales. En tal virtud, declaro que el contenido, las conclusiones y los efectos legales y académicos que se desprenden del trabajo propuesto, y luego de la redacción de este documento, son y serán de mi sola y exclusiva responsabilidad legal y académica.

Félix Mauricio Murillo Calderón
1802998409

Dedicatoria

El gran esfuerzo realizado para la obtención de mi maestría la dedico principalmente a mi esposa Gabriela por el empuje para conseguir las metas, a mis padres Félix y Alicia por sus enseñanzas diarias de optimismo y coraje ya que han velado por mi bienestar y educación siendo mi apoyo en todo momento.

Los amo con mi entera razón de ser.

Reconocimientos

En primer lugar a Dios por haberme guiado por el camino de la felicidad hasta ahora; en segundo lugar a cada uno de los que son parte de mi familia a mi PADRE, mi MADRE, mi Esposa, mi hija Emilie Zoé, a mi hermana y cuñado; por siempre haberme dado su fuerza y apoyo incondicional que me han ayudado y llevado hasta donde estoy ahora. Seguidamente a mis docentes revisores, el Ing. Juan José Ramos por sus sabias palabras y gentileza, una eterna gratitud a la Ing. Verónica Pailiacho por su paciencia demostrada la cual admiro y aprendí a serlo y demostrar amor por el trabajo.

Por último a mi Tutor del Proyecto de Investigación y Desarrollo de Grado porque en esta armonía grupal lo hemos logrado y quién me ayudo en todo momento, Ing. Marcelo Balseca.

Resumen

Este documento presenta el desarrollo del prototipo de red definida por software SDN utilizando para la gestión recursos de estándar abierto, en el que se ha desarrollado aplicaciones con controladores OpenFlow que se adhieran a cualquier tipo de infraestructura de red, buscando generar una mayor eficiencia y optimización de la capacidad de la red en el servicio que se brinda a los usuarios; para lo cual se ha utilizado los modelos ágiles XP en la que su principal herramienta es la disponibilidad de cambios a medida que sea necesario. El aplicativo de red SDN debe monitorear el tráfico de información dependiendo de las políticas establecidas por el administrador de red; adicionalmente la aplicación del prototipo de red presentó comportamientos con niveles de satisfacción aceptables en las pruebas iniciales de enrutamiento, monitoreo y de la reducción del número de paquetes que debe generar el servidor de manera que exista un flujo de información sin interrupciones. Esto permite que la infraestructura de red utilizando el protocolo OpenFlow en base a Floodlight se convierta en una red directamente programable, ágil, escalable, de gestión centralizada y con programación configurada mediante la utilización de Mininet para la creación de redes virtuales reales, lo que permite a los administradores optimizar recursos y mantener la simplicidad en los diseños, convirtiéndolas así en redes flexibles, seguras y eficientes.

Palabras claves: SDN, OpenFlow, Mininet, Floodlight, red, eficiente, ágil, infraestructura, administración de red

Abstract

This document presents the development of a software defined networking (SDN) prototype to manage open standards for witch, OpenFlow controllers applications were developed to support any type of networking infrastructure in order to achieve a higher efficiency while optimizing the capacity of the network and the services offered to the users; and for this reason, the XP extreme programming agile model was used, having as its main tool its ability to make changes so far as is necessary. The application of SDN networking must monitor network traffic depending on the policies set by the network administrator; In addition, the use of a SDN networking prototype showed network behaviors with acceptable levels of satisfaction in the initial tests of routing, monitoring and reducing the number of packets generated by the server to conserve the flow of information without interruption. This allows the networking infrastructure to use the OpenFlow protocol based on FloodLight to become a direct programmable network that is agile, scalable, centrally managed and configured using Mininet to create virtual reality networks, allowing the network administrator to optimize resources and to maintain simplicity in design, making them that are flexible, secure and efficient.

Keywords: SDN, OpenFlow, Mininet, Floodlight, networking, efficient, network, open standars, agile, infrastructure, network management.

Tabla de Contenidos

Lista de Tablas

Lista de Figuras

Capítulo 1

Introducción

1.1 Presentación del trabajo

La investigación es un proceso complejo que requiere de una dedicación armónica de conocimiento científico y contextual, cuyo resultado es un documento que conjuga etapas sucesivas que facilitan entender la aplicación de la Tecnología OpenFlow en la Empresa Avícola Agoyán, reduciendo los problemas en la transmisión de información y optimizando los recursos económicos, tecnológicos y de talento humano como un aporte de la ciencia en la empresa industrial, por su fácil adaptación a diferentes contextos.

La búsqueda de información, lleva al análisis de contenidos técnicos con principios propios de la tecnología OpenFlow que permiten una transmisión de información confiable, oportuna y eficiente que lleve a la empresa a alcanzar sus objetivos como un subsistema de información, cambiando la realidad de la empresa que presenta falencias en esta área; identificadas en la investigación de campo con técnicas de investigación confiables.

La investigación es el fruto del trabajo dedicado liderado por el autor de la obra, en el afán de aplicar los conocimientos adquiridos a lo largo del programa de maestría en Gerencia Informática y supervisados por el Director del Proyecto de Investigación y Desarrollo de Grado, quien ha demostrado un sólido conocimiento del tema, así como cualidades que permiten encontrar la aplicación práctica de los contenidos en el ámbito empresarial. Además hay que destacar la predisposición y colaboración del personal administrativo, comercial, de finanzas y operativo en la recolección de información.

El presente estudio es una inversión de tiempo y dinero que permite obtener una eficiencia de recursos en el ámbito empresarial. El costo total de la investigación es de alrededor de 500 dólares cubiertos por el investigador. Además, la ejecución de la propuesta le costaría a la empresa un total de 2000 dólares, recuperables en 12 meses de tiempo, generando un VAN de 200 dólares en los años de vida útil del proyecto. Los beneficios de orden financiero, serán además beneficios de orden administrativo por facilitar el proceso de control de los procesos planificados. Un beneficio del prototipo de red SDN en el ámbito comercial, es el de ayudar a los miembros del equipo de ventas a obtener información en tiempo real, facilitando la comercialización de productos en los diferentes

mercados que participa la empresa con productos frescos de calidad y con precios cómodos para un consumidor exigente.

El proceso de producción obtiene un apoyo significativo en la ejecución de sus actividades al obtener una retroalimentación oportuna de los demás niveles de la organización, adecuando los procesos a las exigencias del mercado.

1.2 Descripción del documento

La presente investigación es una propuesta técnica de Desarrollo, encausada en la línea de investigación principal Ingeniería de Software y/o Plataformas Educativas, y línea secundaria Redes y Aplicaciones. El mismo que inicia con la identificación del problema detectado en la Empresa Avícola Agoyán, esta mantiene un flujo de información importante entre su planta de producción y su departamento administrativo, por lo que no se detecta la creación de un diseño SDN en la infraestructura de red, además se pueda generar tablas de control de flujo que reflejen la conectividad entre dispositivos. En el capítulo 2, se aborda la descripción del problema y los objetivos indispensables que servirán como punto de partida para el desarrollo del prototipo de red.

En el Capítulo 3 contendrá conceptos sobre redes SDN, según [1] [2] los cuales concuerdan que una arquitectura de red definida por software SDN que permita separar el plano del control, del plano de datos llevando a redes programables, automatizables y flexibles, analizando los requisitos de hardware que lleven al diseño de una red distribuida entre los diferentes dispositivos, para una construcción de la arquitectura de red, que será probada previo a la ejecución definitiva. SDN es funcional ya que se utilizará el protocolo OpenFlow de la arquitectura SDN que establecen comunicación para acceder y manipular el plano de re direccionamiento de dispositivos de red como conmutadores y enrutadores, ya sean físicos o virtuales, llenado a un control descentralizado con políticas de control de flujo de datos.

La sección 4 menciona la Metodología Ágil XP, que se utilizó en la que se resalta una serie de valores y principios en el desarrollo de este proyecto; llevan a entender la comunicación, la simplicidad y la retroalimentación como elementos para alcanzar una propuesta técnica. La planeación, diseño, codificación, pruebas, son los pasos indicados para presentar una propuesta viable, que aplica los contenidos técnicos que permiten estandarizar una propuesta adaptada a las necesidades de la empresa investigada que se presenta como un eje de desarrollo económico y social para la región en la que ejerce influencia.

El Capítulo 5 incluye la presentación del Proyecto Final, donde se demuestra los diferentes comandos utilizados por el Prototipo la presentación por pantallas de uso de SDN ergonómico en la cual se cumple con las expectativas para el control por parte de los administradores de red y finalmente la sección 6 contendrá las conclusiones y recomendaciones que se pueden ejecutar para el desarrollo de este Prototipo en el cual se dejó como resultado que la aplicación de estas redes SDN son de difícil manejo pero que se podrá trabajar a futuro con muchos grupos de desarrollo.

Capítulo 2

Planteamiento de la Propuesta de Trabajo

2.1 Información Técnica Básica

Tema: Desarrollo de un prototipo de red definida por software SDN para la gestión mediante recursos de estándar abierto

Tipo de trabajo: Proyecto de Investigación y Desarrollo de Grado

Clasificación técnica del trabajo: Desarrollo

Líneas de Investigación, Innovación y Desarrollo

Principal: Ingeniería de Software y/o Plataformas Educativas

Secundaria: Redes y Aplicaciones

2.2 Descripción del Problema

Los avances actuales en relación a las infraestructuras de redes no tradicionales en el desarrollo ha innovado su concepción de buscar nuevos modelos de administración con la utilización sistemas eficientes o reglas de conexiones; esto ha promovido a la búsqueda de una arquitectura flexible de manejo altamente técnico que contribuya a mejorar la gestión de la misma con la posibilidad de utilizar y sentar un precedente en el uso de SDN que tiene como característica principal la maniobrabilidad y ajuste de las reglas de conexión.

El desarrollo de este proyecto es de suma importancia ya que no se detecta la creación de un diseño SDN en la infraestructura de red de la Empresa Avícola Agoyán los mismos que se han construido con un largo número de dispositivos de red tales como: Switches, Routers, Firewalls, utilizando muchos protocolos complejos, los cuales son implementados y concentrados en estos equipos.

Esto hace que los administradores de red adquieran la responsabilidad en la configuración para brindar políticas con respuesta inmediata ante eventos de red y aplicaciones [3] incluso en el que

se puedan procesar datos y manejar el tráfico de información con sólo recibir órdenes desde un controlador externo; funcionando como un Switch lógico.

En la actualidad a nivel de gestión de los administradores de red, es un desarrollo excepcional ya que se puede programar esta nueva infraestructura de red sin necesidad de aplicar las configuraciones de los dispositivos a mano y en cada uno de los mismos que conforman la red de datos de la Empresa Avícola Agoyán; primordialmente es de suma importancia el mantener la seguridad, flexibilidad, confiabilidad, disponibilidad de la información corporativa.

Económicamente el presente trabajo se justifica ya que al tener un control centralizado es posible alterar, en tiempo real, el proceder de la red y a su vez desplegar servicios y aplicaciones nuevas en cuestión de horas o días en vez de meses lo que genera gastos innecesarios. Adicionalmente esto puede generar una espera de que los dispositivos nuevos incluyan aplicaciones necesarias.

Los servicios se volverán comunes con un control granular de la red en el enrutamiento, multicast, seguridad de información, control de acceso, administración de Ancho de banda, y QoS, etc; consiguiendo mejorar la experiencia del usuario, a su vez ayudará a incrementar la velocidad de innovación con la mejora de la escalabilidad, flexibilidad, administración y agilidad de la red.

En el Desarrollo del mismo se logrará implementar una red definida por software la cual proveerá una mayor flexibilidad, programable, libre de proveedores, costos eficientes y una arquitectura innovadora de red [3], y obtener una eficiencia energética agradable a los costos de la Empresa, la implementación de este prototipo permitirá construir laboratorios SDN usando OpenFlow como una interface abierta para el control remoto de los dispositivos de red separando los planos de control de los planos de datos, así como permitirá utilizar OpenFlow para cloud computing networking. [3]

2.3 Preguntas Básicas

- **¿Por qué se origina?**

Se origina por el crecimiento de las transacciones comerciales y el manejo acelerado de información que exigen una respuesta inmediata, haciendo que la tecnología agilite estas operaciones. Las SDN surgen como alternativa de las redes actuales, debido a la necesidad de tener una mayor escalabilidad y manejo de tráfico más personalizable y eficiente, ya que la capacidad y eficiencia de las redes actuales ha sido sobrepasada por la cantidad y tipo de tráfico que se maneja.

- **¿Dónde se origina?**

Se origina en la Empresa Avícola Agoyán ya que consta de diferentes áreas de producción como son: Avícola Agoyán: producción y comercialización de huevos para el consumo humano; División oriente: producción y comercialización de pollo en pie y faenado, Fabrica el Troje: Producción y comercialización de Alimento Balanceado; Reproductoras: Producción de huevos incubables para pollito bebe y Planta de Incubación: Producción de pollito bebe.

- **¿Dónde se detecta?**

Se detecta en el intercambio de información que se debe realizar entre las diferentes áreas que posee la Empresa Avícola Agoyán en el cual se desea separar las decisiones lógicas de ruteo y la administración de protocolos para la interacción entre los elementos del Network; el incremento en los servicios prestados por la red de la empresa ha hecho que se busque soliviones inmediatas a los requerimientos actuales de la misma.

2.4 Formulación de Meta

- Desarrollar un prototipo de red definida por software SDN que facilite la gestión utilizando recursos de estándar abierto.

2.5 Objetivos

2.5.1 Objetivo General

- Construir una arquitectura de red definida por software SDN que permita separar el plano del control del plano de datos, para conseguir redes más programables, automatizables y flexibles.

2.5.2 Objetivos Específicos.

1. Analizar los requisitos necesarios de hardware en la evolución tecnológica y la demanda de los clientes.
2. Diseñar la red distribuida entre los diferentes dispositivos individuales a los centros de gestión.
3. Construir la arquitectura de red ajustando los planos de control a partir de la creación de algoritmos de funcionamiento.
4. Realizar las pruebas piloto y evaluaciones antes de comprometerse con una arquitectura específica.

2.6 Delimitación Funcional

El protocolo fundamental OpenFlow de la arquitectura SDN establecerá la comunicación la cual permitirá acceder directamente y manipular el plano de redireccionamiento de dispositivos de red como conmutadores y enrutadores, ya sean físicos o virtuales; por lo tanto, el servidor permitirá que exista mayor control descentralizado con políticas de control de flujo de datos.

OpenFlow permitirá el acceso a dispositivos de red en la cual la programación se verá simplificada mediante una interfaz estándar; esto garantizará que exista mayor disponibilidad, confiabilidad y seguridad en los ambientes.

A través de SDN se logrará el desacoplé de la capa lógica que decide el ruteo del tráfico de la red del sistema que reenvía el tráfico hacia y desde ubicaciones subyacentes. Con la implementación de SDN, la capa lógica se sustituye por un controlador virtual en el que se puede configurar de manera centralizada todos los equipos de la red. La arquitectura brinda más flexibilidad y eficiencia en la utilización de los dispositivos de la organización para que se puedan adaptar a las nuevas aplicaciones particulares en cualquier momento.

Mediante la implementación de SDN, la arquitectura beneficiará en el rápido aprovisionamiento de recursos de la red; la empresa tendrá una migración hacia sistemas de gestión de modernas redes; SDN reducirá costos de implementación para la empresa, se ajustará a las necesidad inmediatas, en la cual se podrá sugerir al administrador de red que utilice uno o dos computadores trabajando como servidores.

Mientras que SDN eventualmente altera la arquitectura de la red, es posible en un periodo corto de tiempo el implementar una tecnología de virtualización de red en un ambiente híbrido en el cual lo tradicional y la tecnología virtual se podrán complementar, se podrá usar switches híbridos en el cual se enviará el tráfico de información de manera tradicional utilizando instrucciones SDN.

Por un lado se usará Switches Multi Layer, para configurar la base de datos, ruteo y VLAN, permitiendo maximizar recursos, además se podrá habilitar IP Routing para poder tener el protocolo OpenFlow para tener seguridad y monitoreo en las aplicaciones en la cual se podrá adicionar más switches programables. [4]

Capítulo 3

Marco Teórico

3.1 Definiciones y conceptos

3.1.1 Software Defined Networking

Es un nuevo enfoque al networking ya que la mayoría de las redes de datos actuales tienen límites para la innovación, SDN[1] desacopla en tres partes a los dispositivos tales como:

Plano de Datos o Forwarding Plane: es el que recibe la información que ingresa por una interface y decide que hacer utilizando las tablas de Forwarding;
Plano de Control o Control Plane: es la parte que aprende la Topología[2] es decir los protocolos de enrutamiento o direcciones de red.
Plano de Administración o Managment Plane: se encarga de revisar el funcionamiento correcto tanto del plano de control como el de datos.

Esta nueva arquitectura es emergente en los nuevos cambios en la industria; aparece en el momento de mayor demanda debido a la diversidad de servicios y a lo complejo que se vuelven las infraestructuras tradicionales; como indica en [5] estas redes superaron la capacidad de los operadores para activar y desactivar servicios rápidamente; lo que las vuelve dinámicas, manejables, rentables y adaptables, haciéndolos ideales para la naturaleza dinámica de banda ancha de las aplicaciones actuales. El resultado de este desacople es dinámico, administrable o gestionable de bajo costo y arquitectura adaptable en la que los administradores no tienen precedentes de programación, automatización y control como lo indica en [6].

Esta arquitectura desvincula las funciones de control y reenvío de la red permitiendo al control de la red hacerse programable directamente quedando abstraídos la infraestructura subyacente para las aplicaciones y los servicios de red. [7]

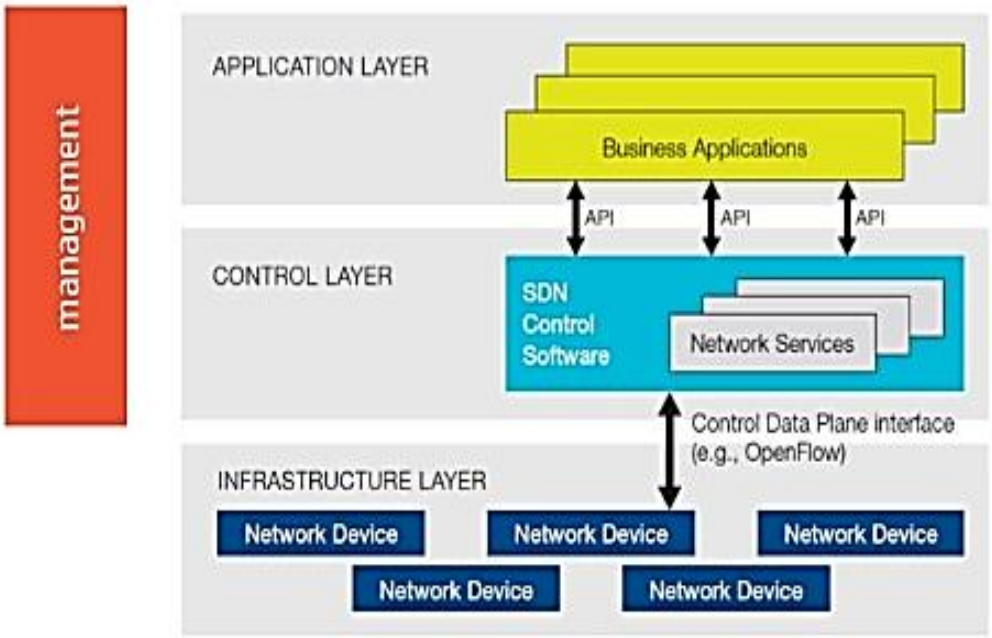
Las características de la arquitectura SDN según [8] son:

[1] SDN: Sofware Defined Networking
[2] Topología: Disposición física de los equipos

- Directamente Programable: Ya que el plano de control de la red se programa directamente debido a que las funciones de forwarding fueron desacopladas.
- Ágil: Al tener monitoreo constante sobre la red de telecomunicaciones, permite al administrador de red o a las aplicaciones hacer cambios dinámicos en los flujos de red según los requerimientos.
- Gestión centralizada: La inteligencia de la red está centralizada en un software llamado controlador SDN, que mantienen una visión global de la red, haciendo parecer una gran red de switches como un único solo switch autómata.
- Programación ordenada: esta arquitectura SDN brinda la posibilidad a los administradores de la red el administrar, configurar y optimizar todos las herramientas de red con lo cual generará ágilmente por medio de los programas dinámicos de SDN los mismos que pueden ser descritos por ellos mismo ya que por una de las características principales no se fundamentan de ningún software propietario o de pago.
- Basados en estándares abiertos y de proveedor neutral: Como se implementa a través de estándares abiertos, SDN simplifica el diseño de la red y la operación porque integra un solo protocolo para todos los fabricantes.

La composición de la arquitectura de SDN se la puede observar en la siguiente figura:

Figura Nº 1 Arquitectura de SDN



Fuente: http://www.muycomputerpro.com/wp-content/uploads/2013/02/sdn_hp1.jpg

3.1.1.1 Arquitectura de las redes SDN

1. Capa de aplicación

Es la de más alto nivel desde la cual se controla el comportamiento de la red. Para ello se utiliza la API[3] del controlador, que puede ser utilizada por usuarios o por otros Software de orquestación o aplicaciones de infraestructura o corporativas [9], para tener una visión global de las condiciones actuales de la red, con el fin de mejorar la transmisión de datos, mejorar la toma de decisiones locales por nodos de redes individuales acerca de cómo tratar los flujos de tráfico de una aplicación en particular y cómo se deben compartir con las plataformas de control de aplicaciones. [8]

Al programar la red se pueden conseguir la implementación ágil con el resto de componentes IT, por ejemplo, haciendo que la red se adapte automáticamente a los movimientos de máquinas virtuales entre Data Centers, permitiendo grandes mejoras frente a los diseños tradicionales.

La capa de aplicación debe mantener racionalizadas las operaciones de los centros de datos para ayudar a ofrecer una mejor automatización, y paralelamente garantizar la seguridad en la red en tiempo real.

Listado de aplicaciones que se encuentran bajo la arquitectura OpenFlow son:

1. FlowVisor (Permite ver el flujo de datos)
2. Aster'X (Topología de red dedicada principalmente a voz IP)
3. Using all wireless Network Around me (La implementación se realiza bajo una aplicación streaming utilizando una red WiFi[4])
4. ElasticTree (Caracterización de un data center, y evalúa el consumo de energía)
5. Open Pipes (Plataforma para la construcción de sistemas de hardware)

2. Capa de control

Se encuentran los controladores, los mismos que mantienen la inteligencia de la red, estos se comunican con los controladores de Hardware de la capa de infraestructura con el protocolo OpenFlow que gestiona el flujo de datos en esta capa y brinda una API abierta para que puedan ser programados desde la capa de aplicación [8] [9]; por medio de esta capa, se crea una vista centralizada de la topología de la red de datos en la cual se programa flujos y se retroalimenta a la aplicación, con información de la topología de red o el tráfico de paquetes. [9].

[3] Application Programming Interface
[4] Wireless Fidelity

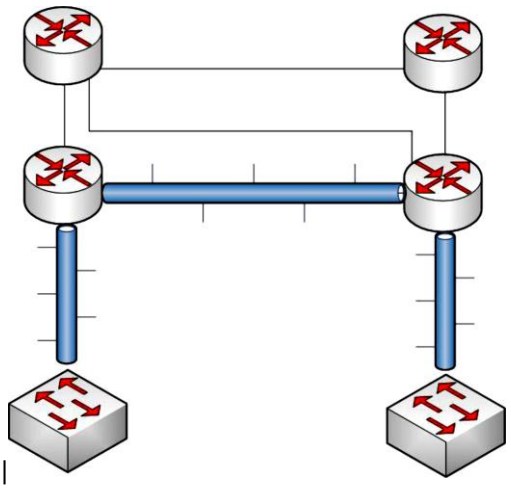
3. Capa de infraestructura

Está formada por dispositivos de red, los mismos que son gestionados por medio del protocolo OpenFlow, permitiendo simplificar la configuración al administrador de red, en el cual únicamente se tiene en cuenta el Hardware que lo componen, no los protocolos que contengan ni ningún tipo de inteligencia, por lo que pueden ser equipos de muy bajo coste y no tendrían que ser de un único fabricante, tan solo deberán contar con el soporte del método que les conecta con el controlador, o lo que es lo mismo, el protocolo OpenFlow.

3.1.1.2 Arquitectura de Red Actual y la Arquitectura SDN

El envío de paquetes y las decisiones de enrutamiento de alto nivel acontecen dentro del mismo dispositivo esto acaece en el networking tradicional, mientras que en las redes definas por software estas dos funcionalidades se separan permitiendo tener un control centralizado. En la siguiente figura se muestra la diferencia de estas dos arquitecturas. [10]

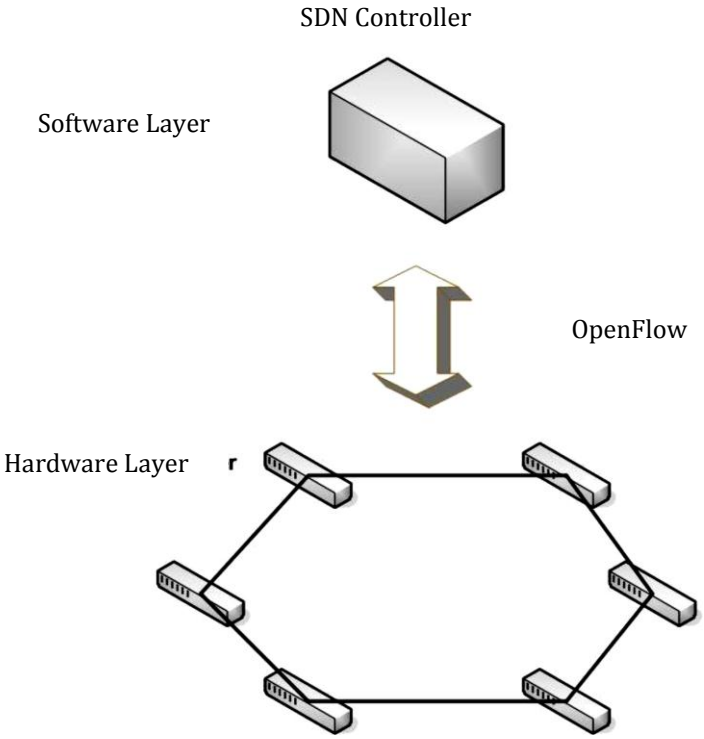
Figura Nº 2 Arquitectura de Red Tradicionales



Fuente: Software Defined Networking [10]

En una red SDN hay un hecho clave: la separación del plano de control del plano de datos. En la que se observa como un único conjunto de medios de transmisión interconectados por medio de varios elementos de conmutación en la capa física, como se puede ver en la arquitectura tradicional, adicionalmente una capa de control o de software por encima de la misma. Esta capa de software comprende el Controlador (*Controller*), que es quien llevará la gestión de la red [10].

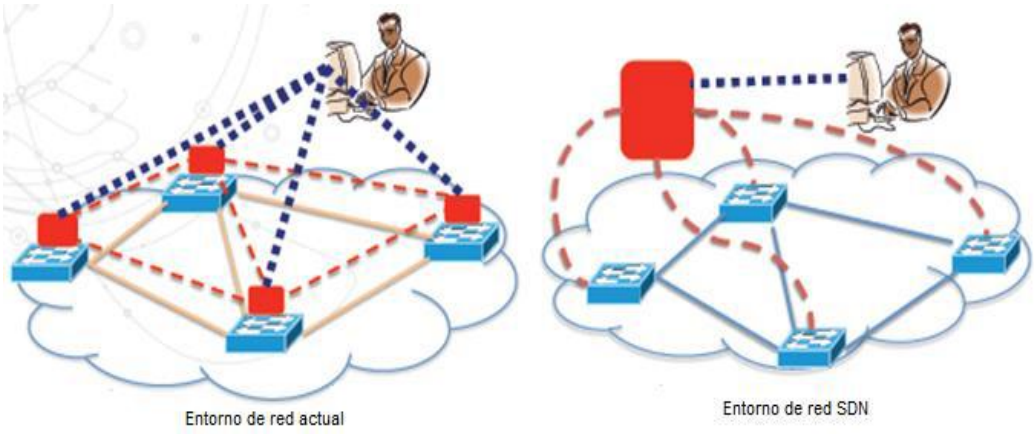
Figura Nº 3 Arquitectura SDN



Fuente: Software Defined Networking [10]

El administrador de red puede moldear el intercambio de información desde la consola centralizada de control sin afectar a los dispositivos conmutadores conectados individualmente [11], mismo que cambia las reglas de los conmutadores de la red siempre y cuando sea necesario agregando o eliminando prioridades, bloqueando o permitiendo el tráfico de paquetes específicos con un control detallado algo que no ocurre en un entorno de red tradicional.

Figura Nº 4 Gestión individual vs. Gestión centralizada



Fuente: http://openlab.web.cern.ch/publications/presentations/software-defined-networking-technology-details-and-openlab-research

Tabla Nº 1 Arquitectura de red tradicional y la Arquitectura SDN

ARQUITECTURA DE RED ACTUAL	ARQUITECTURA DE RED SDN
Envió de paquetes basados en coincidencias de la tabla. Las tablas de flujo están cerradas en los dispositivos. Protocolos totalmente distribuidos Interfaces propietarias Configuración individual de los dispositivos Automatización posible pero tediosa	Tablas de enrutamiento abiertas. Formato y acciones de las tablas claramente especificadas. APIs bien definidas. Control lógicamente centralizado en el Software controlador. APIs abiertas para el acceso y manipulación del plano de Datos o Ej: OPENFLOW. Control central Una solo interfaz (API) para todos los dispositivos. [12]

Fuente: Elaboración Propia

Los Beneficios de las redes definidas por software son los siguientes:

1. **Control de entornos de múltiples proveedores de manera centralizada:** El software de control SDN puede controlar cualquier dispositivo de red habilitado para OpenFlow [1] de cualquier proveedor, incluyendo switches, routers y switches virtuales [11], lo que da lugar a la gestión de conjunto de dispositivos de un solo proveedor.
2. **Automatización mediante la reducción compleja de la red:** SDN ofrece un entorno de automatización y gestión flexible de red [13], lo que hace posible el desarrollo de herramientas que automatizan muchas tareas de administración que se realizan de forma manual hoy en día [11].
3. **Mayor tasa de innovación:** la adopción SDN acelera la innovación empresarial permitiendo a los operadores de red, literalmente programar y reprogramar la red en tiempo real [11] con lo que se satisface especificaciones especiales de negocio.
4. **Seguridad y Fiabilidad de la arquitectura:** Esta arquitectura SDN permite redefinir comandaos de configuración y políticas de gran nivel, que luego son introducidas a la red mediante el protocolo OpenFlow. La misma que está relacionada la cual elimina la necesidad de realizar individualmente la configuración de todos los dispositivos presentes en la red cada vez que se requiera un cambio necesario en un punto específico de servicio, o aplicación, como

resultado esto hace que se reduzca la probabilidad de generar fallas en la nueva arquitectura debido a errores en la configuración.

5. **Mejor experiencia de usuario:** La centralización para el control de la red permite que el estado de la información esté presente inmediatamente para las aplicaciones de gran nivel, esta nueva infraestructura se puede adaptar de mejor manera a los requerimientos específicos del usuario final.

3.1.2 Protocolo OpenFlow

OpenFlow es el primer estándar de interface de comunicaciones definidas entre la Capa de Control y la Capa de Datos de la Arquitectura SDN, admite el directo acceso a la manipulación del plano de datos presente en los dispositivos por ejemplo switches, routers ya sean estos físicos e híbridos los cuales son mencionados por (*hypervisor-based*) [14].

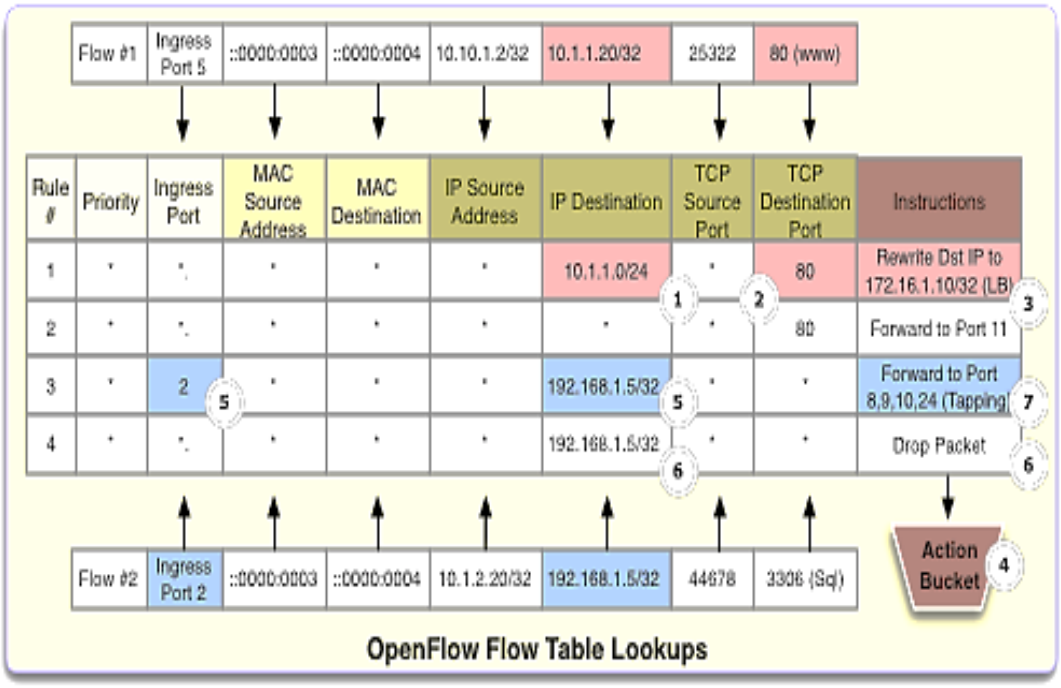
El protocolo se desarrolló en Stanford, y la v1.0 fue publicada a finales del 2009 y la v1.1 a principios del 2011. En marzo del 2011 se creó la ONF y los derechos de propiedad intelectual de OpenFlow fueron transferidos a ella. Parte de la tarea de la ONF es controlar y comercializar el protocolo. [14]

El protocolo OpenFlow se implementa entre la infraestructura de los dispositivos de red y el software de control de SDN. OpenFlow utiliza el concepto de flujos para identificar el tráfico de red basado en reglas de coincidencia predefinidas que pueden ser programadas de forma estática o dinámica por el un controlador SDN lo cual permite definir cómo el tráfico debe fluir a través de los dispositivos de red.

3.1.2.1 Funcionamiento de OpenFlow

OpenFlow aprovecha el hecho que la mayorías de los switches Ethernet contienen tablas de flujo (*Flow-Tables*), aunque cada una de estas son propias de los fabricantes se han identificado varias características en común las cuales son utilizadas por OpenFlow para programar dichas tablas [6] [11], un ejemplo calificado se indica en la siguiente Figura Nº 5:

Figura Nº 5 Open Flow Tabla de Flujos

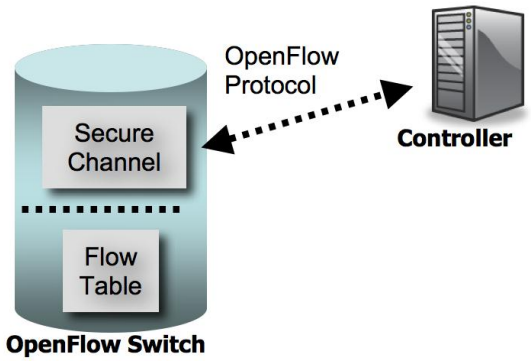


Fuente: http://etherealmind.com/wp-content/uploads/2013/03/sdn-firewall-migration-6.png

El primer paquete de un flujo, arriba al switch y este comprueba las tablas de flujo con el objetivo de hallar similitudes, si ninguna coincidencia fue encontrada cada paquete es re enviado al controlador y este se encargará de añadir la entrada de flujo en la tabla del conmutador switch, luego de insertar la entrada cada paquete que coincida con la entrada añadida nueva se envían directamente al destino sin la necesidad de enviarlo al controlador [15].

El switch OpenFlow se caracteriza por tener dos campos de acción como se indica en la siguiente figura Nº 6.

Figura Nº 6 Comunicación switch OpenFlow **[16] [17]**

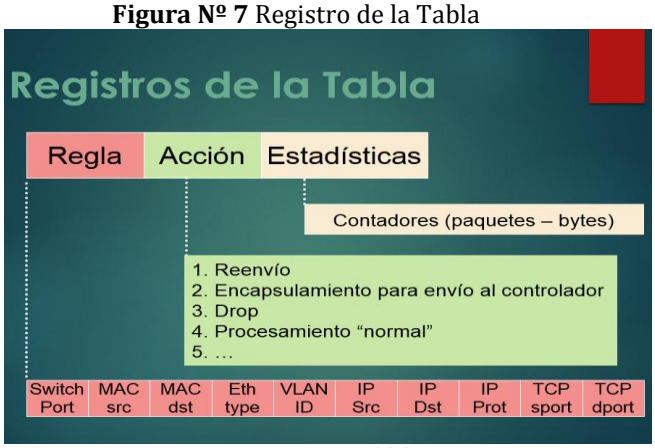


Fuente: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.0.0.pdf

3.1.2.2 Tabla de flujos OpenFlow

Las tablas de Flujo contienen un conjunto de entradas de Flujo como son: cabeceras del paquete para poder comparar los paquetes entrantes [17], un contador de actividad, y un conjunto acciones para emplear en paquetes similares. Todos los paquetes son procesados por el switch y comparados con la tabla de flujos. Si se encuentra una entrada coincidente, alguna acción para esa entrada es realizada (por ejemplo, la acción podría ser la de enviar un paquete fuera de un puerto especifico).

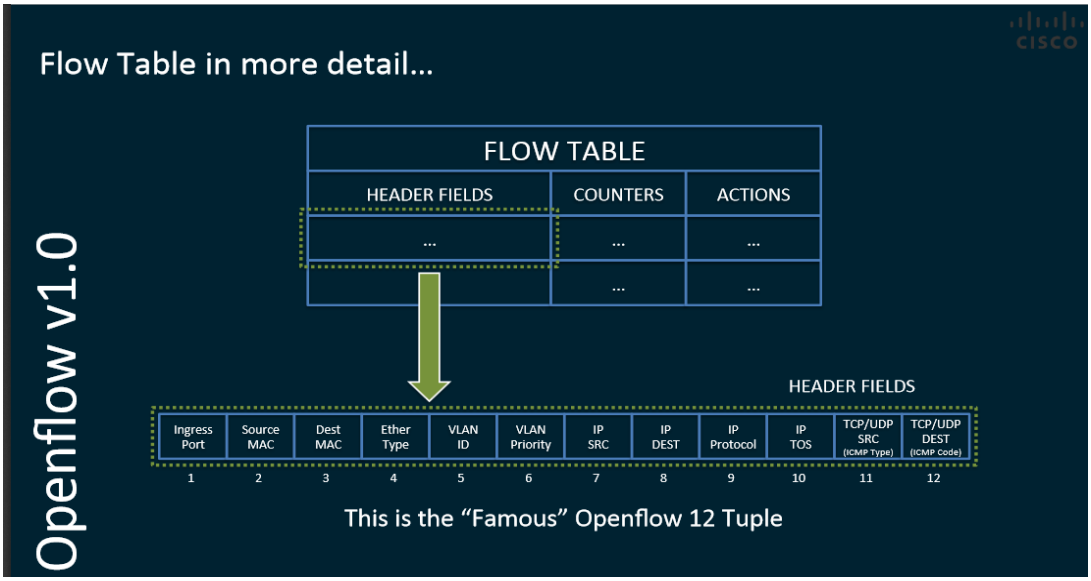
Figura Nº 7 Registro de la Tabla



Fuente: Iván Bernal [2]

Los **Campos de Cabecera** llamados Header Fields, contienen información de la cabecera del paquete y se compara con los paquetes entrantes; en la siguiente figura N° 8 se observa la tabla de Flujos.

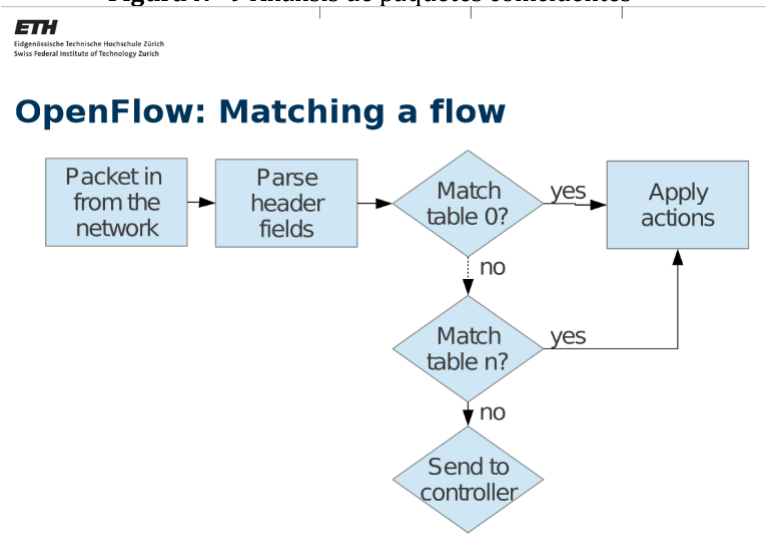
Figura Nº 8 Header Fields – Flow Table



Fuente: Josef Ungerman, Pág. 20 [18]

En la siguiente Figura se muestra como el switch analiza los campos de cabecera para analizar las coincidencias.

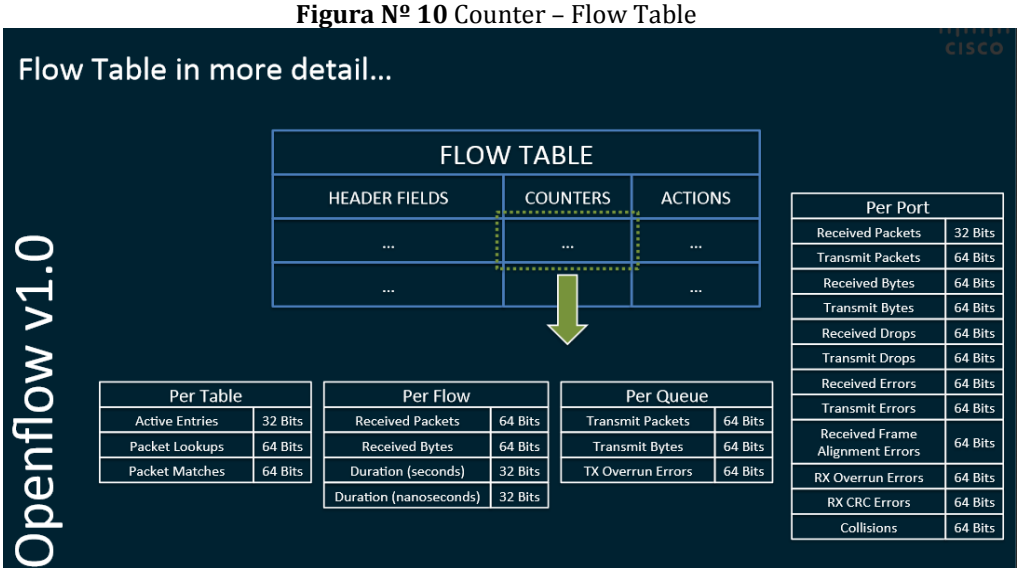
Figura Nº 9 Análisis de paquetes coincidentes



Fuente: Adrian Gämperli - Load balancing using Software Defined Networking [19]

En el caso de **Los Contadores**, estos son utilizados para acumular estadísticas del flujo particular, así como el número de paquetes recibidos, número de bytes, y el tiempo de vida del flujo. [11]

Figura Nº 10 Counter – Flow Table



Fuente: Josef Ungerman, Pág. 20 [18]

Las entradas de Flujo se enlazan mediante acciones con las cuales trata el switch a los paquetes similares. El switch puede rechazar esta entrada de flujo siempre y cuando no se procese la lista de las acciones en el orden predeterminado.

A continuación se presenta las acciones que implementa OpenFlow:

a. **ALL:** Envía el paquete de salida a todas la interfaces y no incluye a las interfaces de entrada
b. **CONTROLLER (Controlador):** Encapsula y envía los paquetes hacia el controlador.
c. **LOCAL:** envía el paquete hacia la pila de red local de los switch
d. **TABLE (Tabla):** Realizar acciones en la tabla de fuljo, solamente para mensajes packet out.
e. **IN PORT:** Envía el paquete de salida hacia el puerto de entrada

Las acciones opcionales que el switch soporta para los siguientes puertos virtuales son las siguientes:

NORMAL: Procesa el paquete utilizando la ruta de envío tradicional apoyado por el switch es decir, VLAN; este switch puede comprobar el campo VLAN[5] para determinar si debe o no enviar el paquete a lo largo de la ruta de procesamiento normal.

FLOOD: Inundación de paquetes a lo largo de spanning tree [20], necesarios en muchos casos para garantizar la disponibilidad de las conexiones de red, pero no incluye la interfaz de entrada.

3.1.2.3 Canal seguro

El canal seguro es la interfaz encargada de conectar cada switch OpenFlow a un controlador, a través de esta interfaz, el controlador configura y gestiona al switch, recibiendo eventos desde el switch y enviando paquetes al mismo [11] [21] [1] .

3.1.3 Tipos de switch OpenFlow

OpenFlow cuenta con dos tipos de switches: 1. OpenFlow, y 2. OpenFlow-hybrid. El switch OpenFlow solo soporta operaciones del protocolo OpenFlow, en estos todos los paquetes son procesados por OpenFlow pipeline, y no pueden ser procesados de otra manera, los switches híbridos poseen la posibilidad de elegir una o más VLAN para trabajar de manera OpenFlow y el restante para hacerlo en modo normal.

Los switches OpenFlow-hybrid soportan ambas operaciones OpenFlow y operaciones de normales switching de Ethernet.

OpenFlow Pipeline de cada switch, está conformado por múltiples tablas de flujos que contiene múltiples entradas de flujo como vemos en la siguiente Figura N° 11.

[5] Red de área local virtual o LAN virtual

Figura Nº 11 Entradas de Flujo OpenFlow 1.3 Tomado de OpenFlow Switch Specification 1.3.0

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie

Fuente: ONF Open Networking Foundation [17]

a) Match Fields

Primera etapa en la que se establece los prerrequisitos para que a un paquete se le aplique varias series de instrucciones, las partes que se evalúan son los encabezados de estos paquetes, los puertos lógicos necesarios y puertos físicos.

b) Priority

Es el valor para que sea evaluado el flujo en el proceso.

c) Counters

Se retroalimenta cada tiempo en el que es procesado el paquete.

d) Instructions

Son una serie de acciones y son ejecutadas en el proceso de pipeline.

e) Timeouts

Tiempo fuera o tiempo máximo que debe permanecer la entrada de flujo, ya sea porque no se utiliza durante un espacio de tiempo o simplemente porque se creó con un tiempo determinado [8].

f) Cookies

Valor seleccionado por el controlador en el cual se filtra las modificaciones que se ejecuta en flujo.

3.1.3.1 SDN Switch puro

En un switch de la arquitectura SDN contiene la mayor parte de las funciones de control de un conmutador tradicional esto quiere decir que posee protocolos de enrutamiento que permiten crear y usar los mimos para las bases de datos de información de reenvío con las que se ejecutan en el controlador central.

3.1.3.2 Switch híbrido

En un Switch híbrido, las tecnologías SDN y los protocolos de conmutación tradicionales funcionan simultáneamente. Un responsable de red puede configurar el controlador SDN para descubrir y controlar ciertas corrientes de tráfico mientras que los protocolos de redes tradicionales y distribuidas continúan dirigiendo el resto del tráfico en la red.

3.1.4 Controladores SDN

Según [11] el controlador es el elemento principal de una red SDN y se considera como el sistema operativo de la misma, el controlador centraliza toda la comunicación que pasa por los dispositivos y provee una visión general de la red. En las SDN, es el controlador central el que dicta el comportamiento general de la red a partir de los requerimientos de las aplicaciones [22].

Las aplicaciones que se ejecutan en el controlador determinan la forma en que los distintos flujos se comportan en la red, el controlador se comunica con los dispositivos a través de OpenFlow y cada elemento de la red se comunica con el controlador pidiendo instrucciones cada vez que no sepa cómo actuar frente a un determinado flujo [23].

En la siguiente tabla se indica los diferentes tipos de Controladores SDN, desarrollados libremente y comercialmente.

Tabla Nº 2 Controladores SDN

CONTROLADOR	CARACTERÍSTICA	AMBIENTE DE DESARROLLO
NOX	Desarrollado paralelamente con OpenFlow Instalación y ejecución en el Controlador Define reglas para la administración de flujo en SDN. Posee una API Entrada y salida asíncrona	Python C++
POX	Similar a NOX Rápido desarrollo Reglas dinámicas para el flujo Comprueba la distribución de flujos Permite depurar y corregir errores.	Python
JAXON	Controlador en Java	Java
TREMA	Software Controlador	C Ruby
BEACON	Multiplataforma se ejecuta en máquinas virtuales Open Source licencia GPLv2 Funcionamiento rápido Interfaz web de manera opcional en Jetty Entrepise	Java
FLOODLIGHT	Corre sobre un gestor de máquinas virtuales Trabaja en dispositivos físicos y virtuales Es modular (se lo denomina Learning Switch) Instalación en Ubuntu Requiere Java Developement Kit JDK Conmuta tráfico entrante y de envío	Java - Python

Fuente: Elaboración Propia

3.1.4 Mininet

Mininet crea redes virtuales, en base a un kernel real, switches y códigos de aplicación sobre una máquina virtual (VM, cloud o nativa), en solo segundos se configura con tan solo un simple comando [24]. Este ejecuta mediante comandos un conjunto de hosts, switches, routers, enlaces y controladores sobre un simple kernel Linux. Utiliza técnicas de virtualización para simular una red completa sobre una máquina virtual [10].

El acceso para cada host se puede realizar de manera individual, se puede hacer una conexión mediante ssh; en la que se podrán enviar paquetes entre los diferentes hosts a través de lo que parecerán enlaces Ethernet. En Mininet los host, switches, links y controladores se comportan como dispositivos reales, la ventaja de mininet es que estos dispositivos son recreados usando software en lugar de hardware [25] [26].

3.1.4.1 Ventajas de Mininet

Este emulador de red presenta varias ventajas presentadas en la siguiente tabla:

Tabla Nº 3 Ventajas de Mininet

CARACTERÍSTICA	DESCRIPCIÓN
RÁPIDO	La puesta en marcha de una red simple tarda sólo unos segundos. Esto significa que el bucle de gestión edit-debug es muy rápido.
CREACIÓN DE TOPOLOGÍAS PERSONALIZADAS	Crea varias topologías diferentes a las existentes en mininet utilizando código basado en Python.
EJECUTAR PROGRAMAS REALES	Acciones que se ejecuten en Linux están presentes para que funcione desde los servidores de Internet hasta el protocolo TCP con esto se inicia el monitoreo de paquetes.
PERSONALIZA EL REENVÍO DE PAQUETES	Los switch en MiniNet son altamente programables con el uso de OpenFlow permitiendo que los diseños de red se pueden migrar de manera fácil a los switches OpenFlow
COMPARTIR Y REPLICAR LOS RESULTADOS OBTENIDOS	Instalado los paquetes específicos para su funcionamiento, es de fácil programación para los administradores

CREACIÓN Y EJECUCIÓN	Se crea scripts de Python para programar de manera flexible.

Fuente: Elaboración Propia

3.1.5 Api Rest

Un API REST (Interfaz de programación de aplicaciones de transferencia de estado representacional), es el mecanismo con el cual el programa puede intercambiar datos o información con una entidad externa a través de HTTP en tiempo de ejecución [27].

La premisa básica es que una aplicación abre un socket servidor de red y espera por peticiones HTTP. Los usuarios externos o incluso otros programas pueden tener acceso a esta API y enviar y recibir mensajes hacia y desde él. Dependiendo de la petición HTTP recibida, o más específicamente la del comando HTTP URI y, la aplicación puede realizar alguna tarea prescrita.

Floodlight implementa un API REST en el núcleo del controlador, pero también hay muchos módulos de aplicación implementados dentro de este. Por ejemplo, el módulo Static Flow Pusher el cual permite a un usuario enviar flujos a Floodlight, así como la eliminación y consulta de los flujos utilizando una petición HTTP para publicar urls. Otros módulos de Floodlight que implementan las API REST son: el módulo Device Manager, Firewall y el módulo Topology Manager. Usando estas APIs REST, el administrador puede realizar varias tareas tales como: consultar por los dispositivos conectados en la red, instalar una regla de firewall que bloquea un dispositivo, o para calcular una ruta a un dispositivo de otro respectivamente.

3.1.5.1 Creación de una Api Rest

La implementación de una API REST brinda la posibilidad de interactuar con el módulo mientras se está ejecutando mediante las API que defina. Puede parecer una tarea de enormes proporciones para crear una interfaz tan accesible en la web, pero el proceso se hace muy fácil a través del servicio de Floodlight IRestApiService.

Este servicio está respaldado por Restlet, que proporciona el servidor HTTP subyacente y la transformación y enrutamiento de peticiones HTTP a los manipuladores dentro de Floodlight.

En un alto nivel, hay cuatro pasos para implementar una API REST en Floodlight [27]:

- Obtener el IRestApiService
- Definir los URls que queremos para nuestro módulo

- Implementar los URls definidos en nuestro módulo
- Decir a Floodlight que nuestra nueva API REST existe

3.2 Estado del arte

En el contexto de largo plazo, la consolidación de territorios y ciudades digitales [28]; aumentará la productividad y la competitividad al activar en el sector productivo el uso de nuevas tecnologías y la generación de nuevas oportunidades de crecimiento e inclusión.

A pesar de que la ingeniería de hardware ha evolucionado la capacidad para hacer el mejor uso de la infraestructura, esta se encuentra limitada por las normas que inadecuadamente han hecho que las redes tradicionales y las arquitecturas del cliente final se encuentren mal adaptadas.

Un motor preponderante en la innovación y futuro de las redes de telecomunicaciones son las universidades y grupos líderes en investigación que están explorando nuevas estrategias para hacer las LAN[6] y WAN[7] del futuro más fáciles de manejar, más seguras y más potentes; capaces de operar sobre diferentes tecnologías bajo el concepto de convergencia[8] y buscando cambiar el modo de controlarlas [29] con esta visión se crea la alianza en Latino América llamada CLARA[9], misma que busca el desarrollo, administración, y dirección de la red regional de Internet RedCLARA [30], el cual su principal propósito es el apoyar el desarrollo de aplicaciones avanzadas en áreas que coexiste única y exclusivamente para las comunidades de educación e investigación siendo en este ámbito el promover la investigación y uno de sus más atractivos aportes es en relación a las arquitecturas SDN [3].

La virtualización es considerada uno de los últimos elementos de las redes tradicionales; es así que el origen del término SDN[10] es complejo; retoma la idea inicial de redes de telefonía que utilizaron la separación del control y planos de datos para simplificar la gestión de redes y el despliegue de los nuevos servicios y en la búsqueda de nuevas alternativas para reemplazar las redes tradicionales y satisfacer las necesidades de las empresas con los crecientes requerimientos en relación al QoS: voz, video, virtualización y cloud computing [3]; ONF impulso la introducción del Estándar OpenFlow el cual permite la programación remota del plano de control y es considerado el componente vital de la arquitectura SDN. [3]

La ONF[11] es una de las mayores industrias sin fines de lucro que demuestra los avances de SDN y la estandarización de los elementos de esta arquitectura, en el cual el protocolo OpenFlow, el cual

[6] Redes de área local
[7] Redes de área extensa
[8] Maximiza el aprovechamiento de la capacidad de las redes
[9] Cooperación Latino Americana de Redes Avanzadas
[10] SDN Software Defined Networking
[11] Open Networking Foundation

conforma estructuras de comunicación entre el control y el plano de datos que soportan los dispositivos de red; es el primer estándar de interface diseñado minuciosamente para SDN, demostrando un alto rendimiento, control de tráfico a través de los dispositivos de red de diferentes proveedores mismo que permite un acceso directo y manipulación de los planos de forwarding de los dispositivos de red como Routers, switches, firewall, WAC; etc., los que pueden ser controlados de manera física y virtual (mediante hypervisor).

OpenFlow necesariamente desacopla el plano de control de los Dispositivos de red hacia un camino centralizado y lógico basándose este en un control mediante software; de hecho, la implementación de SDN en el mundo actual del Networking se ha convertido en una de las mayores tendencias en los Data Center [31] ya que la carencia de escalabilidad y gestión de las redes tradicionales generan la aplicación de un ambiente basado en software. [3]

La innovación en el networking mejora la flexibilidad y la administración holística [32], en la cual habilita la experimentación de redes sin necesidad de impacto o sufrir consecuencias, el propósito de SDN será el de proveer un ambiente funcional considerando los principales componentes de red para generar un balance con la reducción del gasto global de operación rápido crecimiento escalar de la red y una visión de futuro potencialmente acceder a las redes NV (Virtual networking) para que exista comunicación entre red, servidor, y almacenamiento personal.

En el desarrollo de investigaciones a nivel local es considerado uno de los temas de gran auge, ya que por el momento se encuentran realizados estudios como indica en [33] en el Switch virtual se convierte en el centro o núcleo de la mayoría de las redes de los Data Center, esto debido a que la Virtualización como se relaciona en [34] se realiza por medio de ***switches de virtualización***, se divide el ancho de banda disponible en canales seguros, esto permite crear zonas seguras internas y consolidar la seguridad externa.

La virtualización de redes habilita la movilidad de aplicaciones y datos como lo dice en [35], no solo a través de los servidores sino también en las redes y en los Datacenters, también se debe tomar en cuenta la optimización de dicha virtualización para garantizar el alto rendimiento del mismo, es decir, configurar de manera óptima los administradores de máquinas virtuales para maximizar la utilización de recursos. [34]

Al finalizar el estudio de [33] este clarifico que los administradores de red pueden controlar de forma adecuada el flujo de tráfico de la red según las necesidades específicas de la misma; también indica en [33] que creando reglas adecuadas permitirá tener una adecuada gestión de control de acceso la cual se podrá implementar en cualquier tipo de red SDN.

Capítulo 4

Metodología

4.1 Diagnóstico

Las necesidades actuales sobre las redes tradicional en cuanto a la infraestructura y calidad de prestación de servicios, han generado muchos estudios sobre SDN, en la cual se ha visto la necesidad de buscar situaciones puntuales como flexibilidad, protocolos de utilización, y lo más importante es la innovación de las mismas hacia tendencias como el Cloud Computing en el cual el tiempo de administración debe disminuir con menos errores o equivocaciones en el costado de acceso llamado reglas de control de acceso para que no exista perdida de QoS.

Para el desarrollo del prototipo se ha tomado en cuenta la información obtenida de los equipos que soportan SDN, en la Empresa Avícola Agoyán situada en las Provincias de Pastaza con sus plantas de producción y en la Provincia de Tungurahua con sus oficinas administrativas, se encuentra en franca expansión y crecimiento gracias a la calidad del producto y el servicio personalizado que brinda a sus clientes.

Este crecimiento ha llevado a manejar diferentes sistemas de información y comunicación que han facilitado el trabajo en las dos provincias, pero que han quedado obsoletos en pocos meses por las condiciones específicas de sus operaciones.

La adquisición de arquitecturas tradicionales es demasiado costoso, así como el uso de herramientas Web 2.0 ha sido limitado, ya que el flujo de información es masivo el cual limita la transmisión de datos de manera eficiente, a esto se lo llamó el Tiempo Fuera en el cual se degrada el servicio generando retrasos en la conectividad lo que ha llevado en muchas ocasiones a perder información valiosa que significa reprocesos en los trabajos realizados por los administradores de red.

En la búsqueda de mejoras en la calidad de servicio se ha motivado a recurrir a infraestructuras con menos probabilidades a errores, lo cual se debe ajustar a las necesidades específicas de los clientes; estas se relacionan con la definición de políticas para determinado tráfico de información; cambios producidos en el networking pueden ser realizado con SDN de manera rápida y en tiempo Real.

La necesidad de desarrollar un prototipo que sea compatible con los conmutadores de red y que permita al administrador manejar diferentes tipos de acciones o reglas en la empresa, hace que se

pueda tomar decisiones a nivel de gerencia, permitiendo que se desarrolle el mismo con el cual los administradores de red manejarán herramientas puntuales para la gestión de la misma.

Inicialmente, se analizó la viabilidad para proceder a la realización de la encuesta en la que se podrá aclarar interrogantes en relación a la gestión de red las mismas que se generan en los pensamientos de los usuarios que intervienen en el manejo de los dispositivos y equipos de cómputo con sus aplicaciones que permiten la obtención de datos sobre el tráfico y el cumplimiento de las políticas de la red las mismas que se las realizaron a 10 usuarios recurrentes de la red.

La motivación para desarrollar este estudio es de carácter netamente operativo volviendo flexibles a las redes. Además en relación a lo económico estas son baratas en cuanto a la infraestructura, optimizará costos, en circunstancias posteriores se obtendrá un ahorro de energía en la que se mantiene un gasto energético equitativo en los equipos al momento de su uso.

Después de realizar la encuesta (ver **Apéndice A)** a las 10 personas que tienen relación directa con la administración de la red, se obtuvieron los siguientes resultados los mismos que se mencionan en el análisis de la encuesta. Cabe indicar que se relacionó tanto el nivel de conocimiento de información con las preguntas que no contienen un ambiente de información técnica, sino que están dirigidas para poder ser contestadas de manera que las mantengan un grupo de trabajo en relación a la administración y gestión de la infraestructura de red.

Los resultados que se mencionaran son los siguientes:

Evaluación de encuesta.

Resultados Encuesta

1. Indique con una escala de 1 a 5 siendo :

1	2	3	4	5
Muy en desacuerdo	En desacuerdo	Indeciso	De acuerdo	Muy de acuerdo

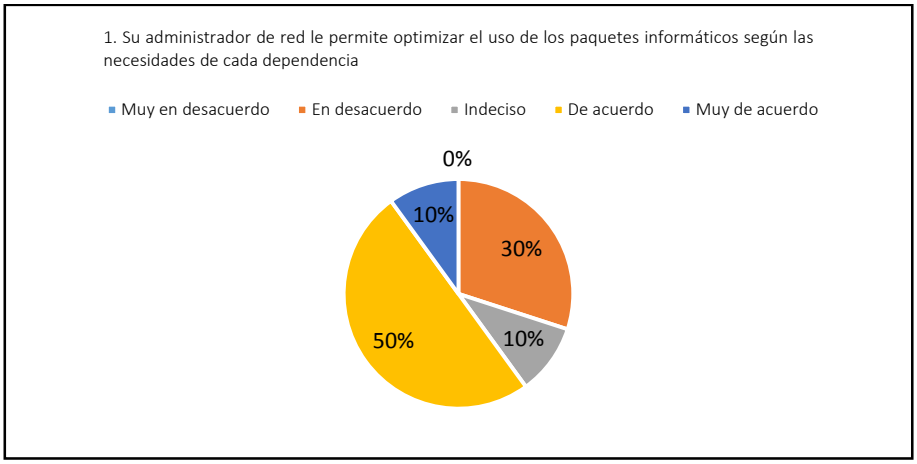
¿Su administrador de red le permite optimizar el uso de los paquetes informáticos según las necesidades de cada dependencia?.

Tabla Nº 4 Optimizar el uso de los paquetes informáticos

	Personas	Porcentaje %
Muy en desacuerdo	0	0%
En desacuerdo	3	30%
Indeciso	1	10%
De acuerdo	5	50%
Muy de acuerdo	1	10%
Total	10	100%

Fuente: Elaboración Propia

Figura Nº 12 Optimizar el uso de los paquetes informáticos



Fuente: Elaboración Propia

Análisis e interpretación:

Los usuarios directos de la red de le Empresa Avícola Agoyán consideran que el administrador de redes les permite optimizar el trabajo; sin embargo, existe un porcentaje considerable que está indeciso o en desacuerdo en esta pregunta. Al concentrar las operaciones en la ciudad de Ambato, ellos consideran que el administrador de redes le permite optimizar, mientras que quienes trabajan en otros lugares, miran de otra manera el uso de este recurso. Las necesidades de cada dependencia son diferentes, por lo que en ciertas dependencias de la institución no se puede conseguir una optimización de diferentes paquetes propicios para un trabajo organizado y pertinente.

2. Indique con una escala de 1 a 5 siendo :

1	2	3	4	5
Muy en desacuerdo	En desacuerdo	Indeciso	De acuerdo	Muy de acuerdo

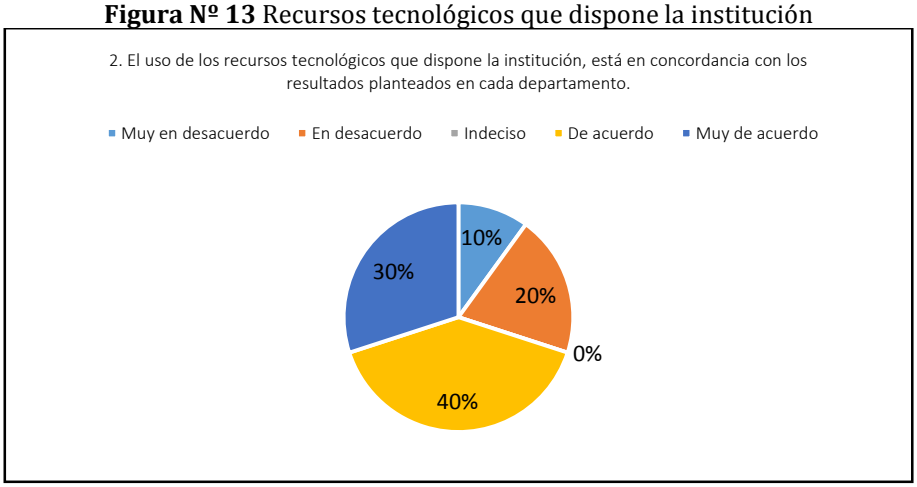
¿El uso de los recursos tecnológicos que dispone la institución, está en concordancia con los resultados planteados en cada departamento?.

Tabla Nº 5 Recursos tecnológicos que dispone la institución

	Personas	Porcentaje %
Muy en desacuerdo	1	10%
En desacuerdo	2	20%
Indeciso	0	0%
De acuerdo	4	40%
Muy de acuerdo	3	30%
Total	10	100%

Fuente: Elaboración Propia

Figura Nº 13 Recursos tecnológicos que dispone la institución



Fuente: Elaboración Propia

Análisis e interpretación:

El 40% de la población investigada piensan que los recursos tecnológicos son pertinentes para las operaciones; sin embargo, la tercera parte restante considera todo lo contrario. Los técnicos especialistas, encargados de los sistemas informáticos y de las redes de comunicación, han encontrado serias dificultades para ejecutar sus actividades especializadas con los recursos que dispone la institución, por cuanto los recursos tecnológicos están especializados en la ejecución de actividades de oficina.

3. Indique con una escala de 1 a 5 siendo :

1	2	3	4	5
Muy en desacuerdo	En desacuerdo	Indeciso	De acuerdo	Muy de acuerdo

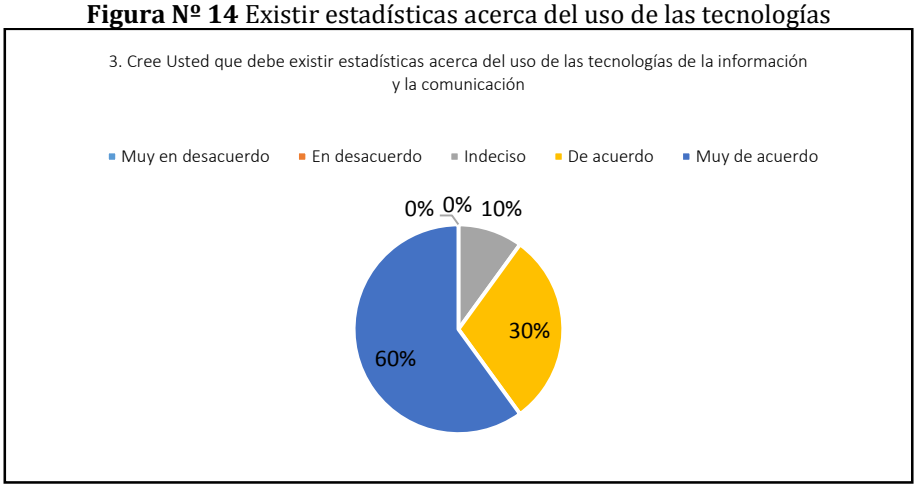
¿Cree Usted que deben existir estadísticas acerca del uso de las tecnologías de la información y la comunicación?

Tabla Nº 6 Existir estadísticas acerca del uso de las tecnologías

	Personas	Porcentaje %
Muy en desacuerdo	0	0%
En desacuerdo	0	0%
Indeciso	1	10%
De acuerdo	3	30%
Muy de acuerdo	6	60%
Total	10	100%

Fuente: Elaboración Propia

Figura Nº 14 Existir estadísticas acerca del uso de las tecnologías



Fuente: Elaboración Propia

Análisis e interpretación:

En base a los resultados obtenidos la mayoría de las personas encuestas está de acuerdo en que se lleven estadísticas acerca del uso de las tecnologías, mientras que uno de cada diez investigados desconoce acerca de la temática por lo que se muestra indeciso. Los encuestados conocen que el tráfico de la información hace que se genere lentitud e ineficiencia, y comprenden que estos retrasos afectan el desempeño del trabajo en equipo, la optimización de la información y el uso de los recursos tecnológicos.

4. Indique con una escala de 1 a 5 siendo :

1	2	3	4	5
Muy en desacuerdo	En desacuerdo	Indeciso	De acuerdo	Muy de acuerdo

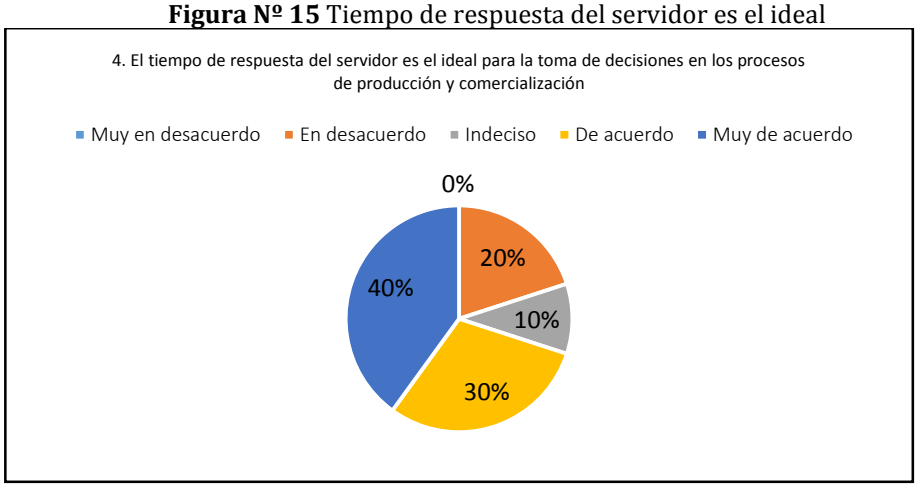
¿El tiempo de respuesta del servidor es el ideal para la toma de decisiones en los procesos de producción y comercialización?

Tabla Nº 7 Tiempo de respuesta del servidor es el ideal

	Personas	Porcentaje %
Muy en desacuerdo	0	0%
En desacuerdo	2	20%
Indeciso	1	10%
De acuerdo	3	30%
Muy de acuerdo	4	40%
Total	10	100%

Fuente: Elaboración Propia

Figura Nº 15 Tiempo de respuesta del servidor es el ideal



Fuente: Elaboración Propia

Análisis e interpretación:

Se considera que el tiempo de respuesta es el ideal en cuanto al manejo de información, mientras que el 10% restante se muestra indeciso o en desacuerdo en relación al tiempo de respuesta. Al comparar con la pertinencia de la tecnología, se nota que se manejan porcentajes similares, por lo que es evidente que dependerá mucho de las necesidades especializadas, por cuanto los administradores de datos se verán imposibilitados de optimizar los recursos acorde con la tecnología proporcionada.

5. Indique con una escala de 1 a 5 siendo :

1	2	3	4	5
Muy en desacuerdo	En desacuerdo	Indeciso	De acuerdo	Muy de acuerdo

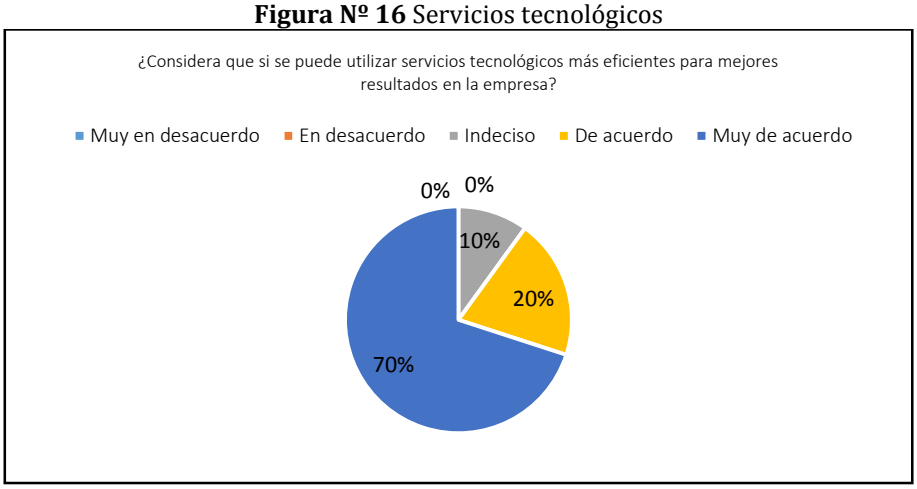
¿Considera que si se puede utilizar servicios tecnológicos más eficientes para mejores resultados en la empresa?

Tabla Nº 8 Servicios tecnológicos

	Personas	Porcentaje %
Muy en desacuerdo	0	0%
En desacuerdo	0	0%
Indeciso	1	10%
De acuerdo	2	20%
Muy de acuerdo	7	70%
Total	10	100%

Fuente: Elaboración Propia

Figura Nº 16 Servicios tecnológicos



Fuente: Elaboración Propia

Análisis e interpretación:

Un 70% de encuestados considera que se puede obtener mejores servicios de tecnología, mientras que una décima parte se muestra indeciso. A pesar de que la empresa se encuentra laborando normalmente con este servidor; sin embargo, existen momentos en los que colapsa la información, siendo imposible ejecutar las operaciones con eficiencia, por lo que el análisis de nuevas posibilidades de proveedores de este servicio es válida.

4.2 Método aplicado

Para la aplicación se ha utilizado la Metodología Ágil XP, ya que el mismo resalta una serie de valores y principios que deben tomarse en cuenta durante el desarrollo del proyecto; la comunicación, la simplicidad, la retroalimentación como parte fundamental de un sistema de lazo cerrado es considerado como actuadores y sirven de pilar fundamental en el desarrollo, la programación se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad[12].

Las características que se observarán en el desarrollo del prototipo son las siguientes:

- Metodología liviana de desarrollo.
- Conjunto de prácticas y reglas empleadas para desarrollar el prototipo
- Basada en diferentes ideas acerca de cómo enfrentar ambientes muy cambiantes
- En vez de planificar, analizar y diseñar para el futuro distante, se realizará todo esto un poco cada vez, a través de todo el proceso de desarrollo

4.2.1 Fases de la Metodología XP

El desarrollo de esta metodología se inicia con la obtención de conocimientos previos por parte de equipo de trabajo cabe indicar que la comunicación con el cliente es válida para encontrar la satisfacción o no del prototipo. En estas fases, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto como se menciona en [36]. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en este proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo.

Las fases para el desarrollo del prototipo son las siguientes:

- Fase: Planeación.
- Fase: Diseño.
- Fase: Codificación.
- Fase: Pruebas.

4.2.1.1 Planeación

La etapa inicial que es la **Planeación** de la metodología aplicada se interactuó mediante un conversatorio con el administrador de la red el cual sostuvo e indico los requerimientos que los usuarios de la infraestructura demandan al departamento de TI, en esta se sienta un precedente

[12] Conocimiento de algo con anticipación por medio de ciertas señales o indicios:

para garantizar tiempos cortos de comunicación, historias de usuario, velocidad de información, entregas pequeñas, reuniones, etc.

La empresa maneja una arquitectura de red con equipos de baja gama y no administrables lo que permite una limitada sincronización de la información de acuerdo con las necesidades de la administración. La topología de la red actual está montada en estrella y cuenta con un modem Motorola SB5100 SURFboard, otorgado por la Empresa Tv Cable S.A el cual permite tener 5 direcciones IP públicas, los cuales están en conexión con un Router TP LINK de serie TL – WR841N, convirtiendo a la misma en una arquitectura no ampliable, poco flexible la cual no se ajusta a las necesidades de los usuarios de la empresa ya que los mismos manejan flujo de trabajo demasiado costoso. Los requerimientos presentados y analizados en su lenguaje natural de los servicios que se espera que el prototipo cumpla o que el sistema provea y de las restricciones bajo las cuales debe operar se han considerado realizando detalles intrínsecos que la red puede brindar y son los siguientes:

Tabla Nº 9 Requerimiento de la Arquitectura

Id	Solicitud - Usuario - Administrador	Descripción	Requerimiento Funcional	Requerimiento de Arquitectura
S1	Disponer de acceso remoto las 24 horas del día.	Controlar todas las acciones de la Pc desde una computadora ubicada en otro lugar.		X
S2	La red debe estar separada en segmentos de red diferentes a través de VLAN.	Dentro del diseño la red que preste servicio a los usuarios debe ser un segmento de red diferente al de los administradores de red		X
S3	Acceso vía web	Los administradores podrán validar los flujos de comunicación desde cualquier lugar con acceso a internet.	X	X
S4	Las conexiones remotas se pueden realizar con protocolos de comunicación seguros.	Las conexiones de acceso remoto deben manejarse a través de un canal seguro a través de la red de manera cifrada		X
S5	Manejo de concurrencia	Que el sistema ofrezca la posibilidad de que varios usuarios trabajen de forma simultánea.	X	X
S6	Un sistema de monitorio y supervisión	Se debe permitir al administrador de red supervisar y monitorear en línea y de forma a remota el trabajo de los usuarios tales como: tiempos de conexión, caída de información.	X	X

S7	Visualización remota de trafico de información	Creación de un entorno de trabajo que permita la visualización en tiempo real	X	X

Fuente: Elaboración Propia

4.2.1.2 Diseño

En base a la información brindada sobre los requerimientos de los usuarios al administrador de red el diseño de la misma debe ser sencillo para lo cual se elaborará diagramas útiles que pueden convertirse en la columna vertebral del desarrollo del prototipo, en este lapso se elegirán los controladores específicos SDN en base a la tabla Nº 2 y se analizará cuál de ellos es el más apropiado para construir la nueva arquitectura de red; es importante resaltar que esta tarea es permanente durante el ciclo de vida del proyecto partiendo del punto inicial el cual va siendo corregido y mejorado en el transcurso del mismo.

Para la elección del controlador a ser utilizado y de acuerdo a las características se determinó que Floodlight es un Open SDN Controller de mayor capacidad para el desarrollo del prototipo según las siguientes características:

- Dispone de un sistema de módulo de loading que hacen que sea fácil de extender y mejorar la infraestructura.
- Fácil de configurar.
- Compatible con una amplia gama de switch virtuales y físicos que soporten OpenFlow
- Diseñado para ser de alto rendimiento el cual es multiproceso desde su inicio

El controlador se encarga de las siguientes operaciones según [2]:

- Software centralizado (lógicamente)
- Interactúa con todos los dispositivos de conectividad.
- Dispone de API abiertas.
- Actúa como un sistema operativo de red
- Las aplicaciones que se ejecutan en el controlador determinarán cómo se comportan los flujos.

La construcción y configuración de Floodlight se realiza siguiendo los siguientes pasos como se indica en la Figura N° 17:

Figura Nº 17 Configuración de Floodlight

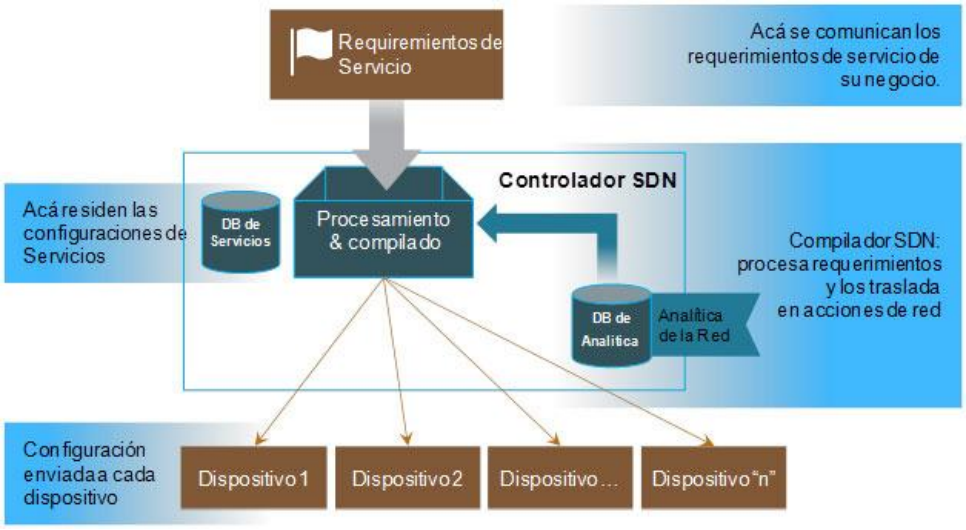
```
$ git clone git://github.com/floodlight/floodlight.git
$ cd floodlight
$ ant
$ sudo mkdir /var/lib/floodlight
$ sudo chmod 777 /var/lib/floodlight
```

Fuente: http://www.projectfloodlight.org/getting-started/

Para el correcto proceso del desarrollo del proyecto una de las principales partes es empezar por el diseño de la arquitectura, en la que se permita a los administradores de red ejecutar e investigar de manera remota sobre ambientes físicos reales los cuales podrán programar prácticas e interactuar con los dispositivos.

En la etapa de diseño de la arquitectura de la red es de manera fundamental identificar la estructura de los eventos esto implica describir mediante un diagrama como estarán distribuidos los usuarios en la red como se lo indica en la figura N° 18.

Figura Nº 18 Estructura de la red

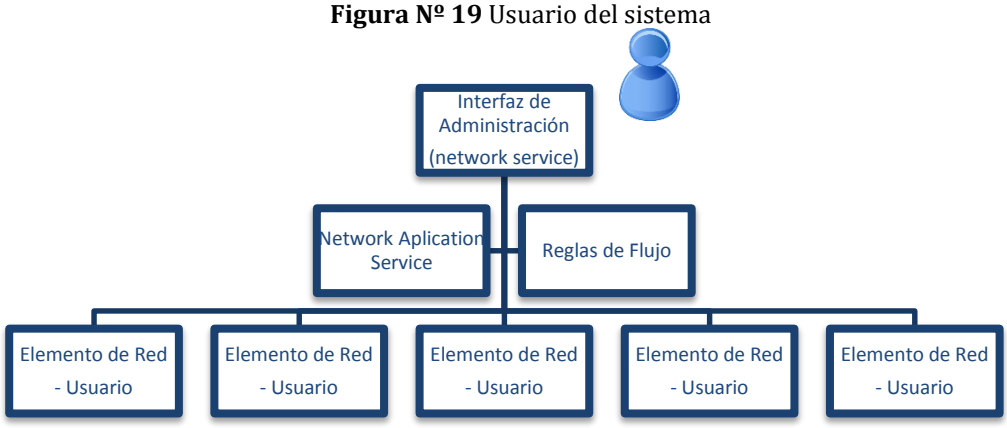


Fuente: Requerimientos de red [36]

4.2.1.2.1 Identificación de Usuarios del Sistema

La identificación del usuario se encuentra en la capa más baja de la Figura Nº 19, la que se encuentra dividida en un sentido jerárquico comenzando con la interfaz del administrador que en el desarrollo es la página web seguidamente la capa del controlador SDN que es FloodLight y los usuarios que se consideran en la capa de interfaces asociadas al controlador SDN.

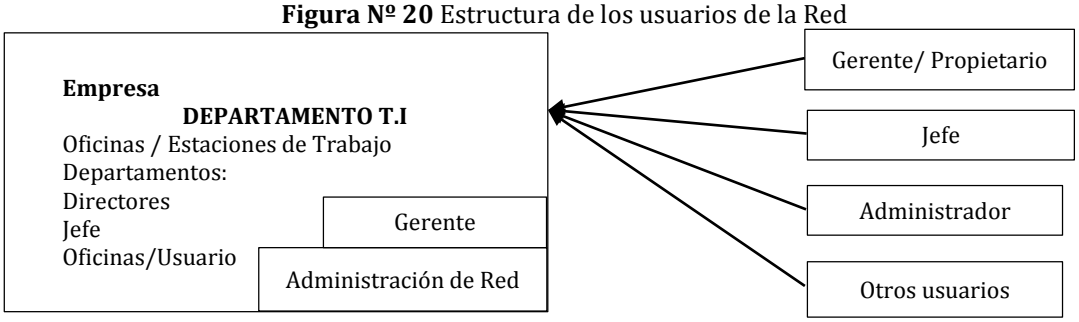
Figura Nº 19 Usuario del sistema



Fuente: Elaboración Propia

La palabra usuario significa principalmente los usuarios finales del sistema, se puede incluir a todos los involucrados en el sistema por ejemplo: administradores de contenidos, administradores de red y sistemas de gestión como se indica en la Figura N° 20.

Figura Nº 20 Estructura de los usuarios de la Red



Fuente: Elaboración Propia

En el diseño de este prototipo se contemplan tres subredes que son las siguientes:

1. **Gerencial**
    a. Gerente General
    b. Subgerente

2. **Departamentos**
    a. Jefe de Administración
    b. Jefe de TI
    c. Jefe de Ventas
    d. Jefe de Producción
    e. Jefe de Finanzas
    f. Jefe de Logística

3. **Subordinados**
    a. Empleados en general

4.2.1.2.2 Diseño de la Arquitectura de Red SDN

Para este desarrollo de SDN hay que tomar en cuenta que los enrutadores, switches deben soportar el protocolo OpenFlow, seguidamente que el respaldo de back up sea provisto por dos diferentes empresas de comunicaciones para poder garantizar la alta disponibilidad de la red a todo instante. Mientras que en la capa de acceso se encuentra el acceso a las tres subredes como también se encuentra la conexión de usuarios finales, redes inalámbricas y servicios.

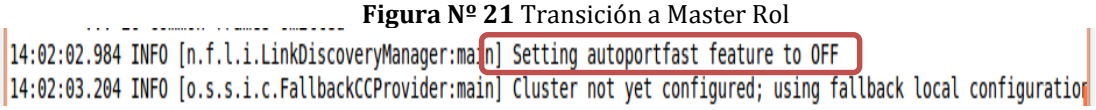
La arquitectura se utilizará un modelo jerárquico, dado que la misma se administra y expande con mayor facilidad ya que consiste en una colección de segmentos (registros) que se conectan entre sí por medio de enlaces. El diseño de este tipo de red está constituido por tres capas independientes las cuales cumplen una función específica.

La separación en capas hace que el diseño de la red se vuelva modular según [37] es un modelo jerárquico típico consta de tres capas a saber: capa de acceso, capa de distribución y capa de núcleo.

En la **capa de acceso** se encuentra disponible los dispositivos del cliente final, tales como computadoras de escritorio o portátiles, impresoras de red; teléfonos IP, lo cual proporciona acceso a la red y está también incluye los router OpenFlow, switches Openflow, y los puntos de acceso inalámbricos AP.

En el desarrollo del prototipo esta capa está formada por la primera línea de dispositivos de red en la que cada host está conectado a un dispositivo de red e intrínsecamente la función de acople cumple el OpenFlow Hypervisor Switch o a su vez el OpenFlow Physical Switch al momento de la primera conexión el ambiente por defecto de Floodlight las tablas de flujo se encuentran libres como se indica en la Figura N° 21.

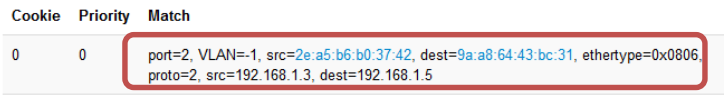
Figura Nº 21 Transición a Master Rol



Fuente: Elaboración Propia

Para la **capa de distribución** se añade los datos o información recibida de los switches de la capa de acceso; esto se hace antes de que se transmitan a la capa núcleo para el enrutamiento hacia su destino final, en esta capa se controla el flujo de tráfico de la red con la implementación de políticas y se encargará de trazar los dominios de broadcast al momento de realizar el enrutamiento de los destinos entre las LAN virtuales (VLAN) (Virtual Local Área Network, Redes Virtuales de Área Local), todo esta información se puede encontrar en la Figura N° 22.

Figura Nº 22 Información de las políticas de tráfico



Cookie	Priority	Match
0	0	port=2, VLAN=-1, src=2e:a5:b6:b0:37:42, dest=9a:a8:64:43:bc:31, ethertype=0x0806, proto=2, src=192.168.1.3, dest=192.168.1.5

Fuente: Elaboración Propia

En la **capa de núcleo** se considera el backbone de alta velocidad de la red, es esencial para la interconectividad e interconexión de los dispositivos de la capa de distribución; ya que es importante que el núcleo sea sumamente disponible, redundante y se puede conectar a los distintos

puertos de la arquitectura. En la Figura N° 23 se considera la acción que realiza y el tiempo de respuesta de la conexión.

Figura Nº 23 Acciones de la regla de flujo

Cookie	Priority	Match	Actio	Packets	Bytes	Age	Timeout
0	0	port=2, VLAN=-1, src=2e:a5:b6:b0:37:42, dest=9a:a8:64:43:bc:31, ethertype=0x0806, proto=2, src=192.168.1.3, dest=192.168.1.5	outpu 4	1	42	4 s	15 s
0	-1	port=4, VLAN=-1, src=9a:a8:64:43:bc:31, dest=2e:a5:b6:b0:37:42, ethertype=0x0806, proto=1, src=192.168.1.5, dest=192.168.1.3	outpu 2	1	42	4 s	15 s

Fuente: Elaboración Propia

En este punto se aclara que el núcleo añade el tráfico de todos los dispositivos de la capa de distribución, con lo que se considera a la red jerárquica en la que existen beneficios en cuanto a la escalabilidad, al rendimiento de la red, seguridad, facilidad de administración y facilidad de mantenimiento.

4.2.1.2.3 Direccionamiento IP

El principal propósito es el evitar el desperdicio amesurado de las direcciones IP en la red, para eso se utilizará en la configuración de la Red el método VLSM (Variable Length Subnet Masking). Según la distribución se requiere de 5 redes LAN, y 2 redes WAN para crear la infraestructura necesaria para este desarrollo; así creamos las siguientes subredes:

El número total de hosts necesarios en la red es de 57 ya que este número es menor a 254 direcciones IP el direccionamiento de esta red se lo realizara en Clase C; como es de conocimiento teórico se organiza las mismas desde la red más grande a la más pequeña, para lo cual la clave es empezar a asignar el direccionamiento desde la red de tamaño más grande.

Luego de haber determinado anteriormente la cantidad de hosts necesarios para la red, presentamos la siguiente tabla con las direcciones encontradas:

Tabla Nº 10 Dirección de subneteo para la Red

NÚMERO DE SUBRED	DIRECCIÓN RED	DIRECCIÓN PRIMER HOST	DIRECCIÓN ÚLTIMO HOST	DIRECCIÓN BROADCAST	MÁSCARA DE SUBRED
0	192.168.1.0	192.168.1.1	192.168.1.30	192.168.1.31	255.255.255.192
1	192.168.1.32	192.168.1.33	192.168.1.46	192.168.1.47	255.255.255.240
2	192.168.1.48	192.168.1.49	192.168.1.62	192.168.1.63	255.255.255.240
3	192.168.1.64	192.168.1.65	192.168.1.70	192.168.1.71	255.255.255.248
4	192.168.1.72	192.168.1.73	192.168.1.78	192.168.1.79	255.255.255.248
5	192.168.1.80	192.168.1.81	192.168.1.82	192.168.1.83	255.255.255.252

6	192.168.1.84	192.168.1.85	192.168.1.86	192.168.1.87	255.255.255.252
7	192.168.1.88	192.168.1.89	192.168.1.90	192.168.1.91	255.255.255.252
8	192.168.1.92	192.168.1.93	192.168.1.94	192.168.1.95	255.255.255.252
9	192.168.1.96	192.168.1.97	192.168.1.98	192.168.1.99	255.255.255.252

Fuente: Elaboración Propia

En la correspondiente asignación de direcciones IP para los dispositivos de red de ruteo se presenta la siguiente tabla:

Tabla Nº 11 Asignación de direcciones IP

DISPOSITIVO	INTERFACE	DIRECCIÓN IP	DEPARTAMENTOS
Router 0	S/Q/3/0	192.168.1.98	
Router 1	F/0/0	192.168.1.30	Gerencial
	S/0/0/0	192.168.1.93	
	S/0/0/1	192.168.1.81	
Router 2	F/0/0	192.168.1.46	Departamentos
	S0/1/0	192.168.1.85	
	S0/0/1	192.168.1.82	
Router 3	F/0/0	192.168.1.46	
	S0/1/0	192.168.1.85	
	S0/0/1	192.168.1.82	
Router 4	F/0/0	192.168.1.78	Subordinados
	F/0/1	192.168.1.70	
	S/0/0/1	192.168.1.89	
	S/0/0/0	192.168.1.94	

Fuente: Elaboración Propia

Cabe indicar que las dirección pueden variar dependiendo de las condiciones especificadas por el administrador de red esto quiere decir que las mismas varían ya que están configuradas en su mayoría por DCHP[13].

4.2.1.2.4 Análisis de los dispositivos de bajo costo para OpenFlow

Tabla Nº 12 Lista de Switch OpenFlow

DISPOSITIVO	DESCRIPCIÓN	PRECIO DE REFERENCIA
Router TP-Link TL-WR1043ND versión 1.11	Router que consta de un procesador potente (procesador Atheros AR9132@400Mhz) compatible con la versión Attitude Adjustment 12.09 de Open Wrt.	TP-LINK TL-WR1043ND V2 Wireless N300 Gigabit Router, 300Mbps, USB port for Storage, 3 Detachable Antennas, Speed Boost up to 450Mbps, WPS Button $500,00
Linksys WRT54GL – OpenWRT de Cisco	Un puerto Internet RJ-45 10/100 Mbps Cuatro puertos Ethernet RJ-45 10/100 Mbps	Linksys WRT54GL Wi-Fi Wireless-G Broadband Router $550,00

[13] Protocolo de configuración dinámica de host

	Todos los puertos soportan intercambio de pines por software MDI/MDI-X CPU Broadcom BCM5352 @200MHz Memoria RAM de 16 Mb Memoria Flash de 4 Mb	

Fuente: http://archive.openflow.org/wk/index.php/OpenFlow_1.0_for_OpenWRT

4.2.1.3 Codificar

Debido a la gran cantidad de información obtenida en los anteriores pasos es necesario que se trabaje en pareja con el administrador de red ya que existirán áreas de trabajo afectadas por el gran flujo de información, en este punto se integrará todos los requerimientos para acceder al uso de la nueva arquitectura.

De manera inicial esto representa una reducción del 50% en productividad de trafico de información, sin embargo, según esta metodología XP no es considerada como pérdida, cuando se trabajó con el administrador se pudo obtener un diseño de mejor calidad con un código más organizado y con menores errores que si se hubiera trabajado de manera autónoma; además según SDN en [38] es una ventaja representativa contar con un desarrollador que ayudará a solucionar inconvenientes en tiempo de codificación, los cuales se presentan con mucha frecuencia.

4.2.1.3.1 Gestor de Máquinas Virtuales

Para la codificación del desarrollo del prototipo se realizará en base a los elementos necesarios que actuarán sobre MININET que son los siguientes:

1. VirtualBox
2. Cliente de telnet/SSH

Estas aplicaciones o herramientas son de acceso libre, de manera fácil al descargar de internet, por lo que en un tiempo escaso se podrá tener un ambiente de trabajo sobre **MININET**.

El gestor de máquinas virtuales es un aplicativo que es capaz de permitir que múltiples sistemas operativos se integren en uno solo sin tener que alterar o modificar la información presente en la partición del sistema operativo raíz o nativo. Para determinar que Gestor es el adecuado para el desarrollo de este Prototipo se indica en la siguiente Figura N° 24.

Figura Nº 24 Tabla comparativa de Gestores de Máquinas Virtuales

Interfaz y uso de la aplicación	VirtualBox	Xen	Virtual Machine Manager
Interfaz de la aplicación sencilla	SÍ	SÍ	SÍ
Barra de herramientas con accesos directos	SÍ	SÍ	NO
Herramientas de accesibilidad	NO	NO	NO
Aplicación multiplataforma	SÍ	NO	NO
Aplicación disponible en español	SÍ	SÍ	NO
Interfaz disponible en varios idiomas	SÍ	SÍ	NO
Autoejecutables	SÍ	NO	NO

Características	VirtualBox	Xen	Virtual Machine Manager
Puede ser utilizado desde medios externos.	SÍ	NO	NO
Alberga todo tipo de S. Operativos	SÍ	SÍ	SÍ
Alta velocidad de ejecución.	SÍ	SÍ	SÍ

Fuente. http://www.bilib.es/fileadmin/Comparativa_gestor_maquinas_virtuales.pdf

Para la virtualización de la red se utilizará VirtualBox ya que una de las principales características es que permite la ejecución simultánea de varias instancias de máquinas virtuales albergando diferentes y/o similares sistemas operativos en cada una de ellas, lo que hace que sea la mejor herramienta de virtualización [44] como lo indica en la Figura Nº 24.

Para la configuración de VirtualBOx se hace lo siguiente:

1. Memoria RAM de 1 Gb.
2. Se configura con un solo NAT para acceder al internet.

En este caso las interfaces están correctamente configuradas: tanto eth0, eth1, eth2, eth3 disponen de dirección IP; si alguna de ellas careciera de IP ejecutar el siguiente comando:

$ sudo dhclient ethX

Ejemplo:

$ sudo dhclient eth0

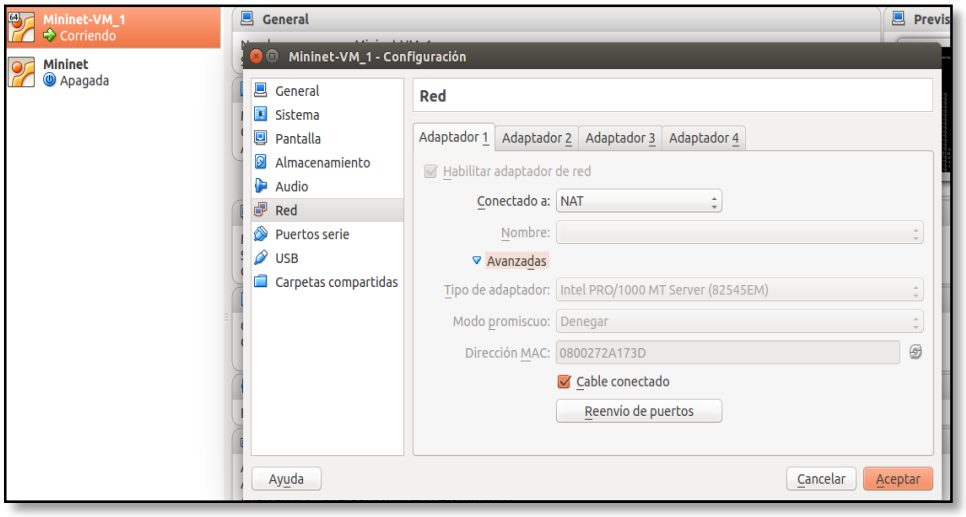
$ sudo dhclient eth1

$ sudo dhclient eth2

$ sudo dhclient eth3

La interfaz caída se puede identificar mediante la Dirección MAC en las configuraciones Virtual Box; habitualmente las direcciones empiezan con la IP 10.0.0.x que corresponden a la Configuración NAT como se indica en la Figura Nº 25.

Figura Nº 25 Configuración de NAT



Fuente: Configuración Virtual Box

El otro tipo host-only-interface, a fin de comunicarse con la máquina host. Cuando se establezcan sesiones ssh, es con la IP de esta última interfaz con la que se establecerá comunicación [10].

4.2.1.3.2 Configuración de Phtyon

Este es un lenguaje de multipropósito o propósito general, el cual se lo puede utilizar de manera interactiva en el que se permite experimentar en una ventana mediante los programas en los cuales se desarrollan y prueban a lo largo de que son construidos. Las siguientes son características especiales de este lenguaje:

1. Lenguaje de alto nivel
2. Propósito general
3. Multiplataforma (*Open Source*)
4. Sintaxis clara

Sigue los siguientes pasos para empezar con el proyecto del Desarrollo de SDN los cuales son:

1. Acceso individual a cada host
2. Conexión SSH[14] con cada host
3. Individualmente el host ejecutará aplicaciones que se hayan instalado en el sistema Operativo Linux.
4. Los paquetes se enviaran entre los distintos hosts mediante enlaces de Ethernet

[14] SSH: Secure Shell (SSH) protocolo de seguridad para comunicaciones de Network

4.2.1.3.3 Configuración de MININET

Topologías personalizadas se crea con Python API que es el siguiente:

-mininet> es el prompt que presenta la CLI de mininet.

Login: mininet

Password: mininet

La topología basado en mininet/topo.py:SingleSwitchTopo, se considera la creación de host específicos llamados h1 hasta hN conectados a un simple switch s1.

A continuación en la Figura N° 26 se presenta parte del código de configuración en MININET

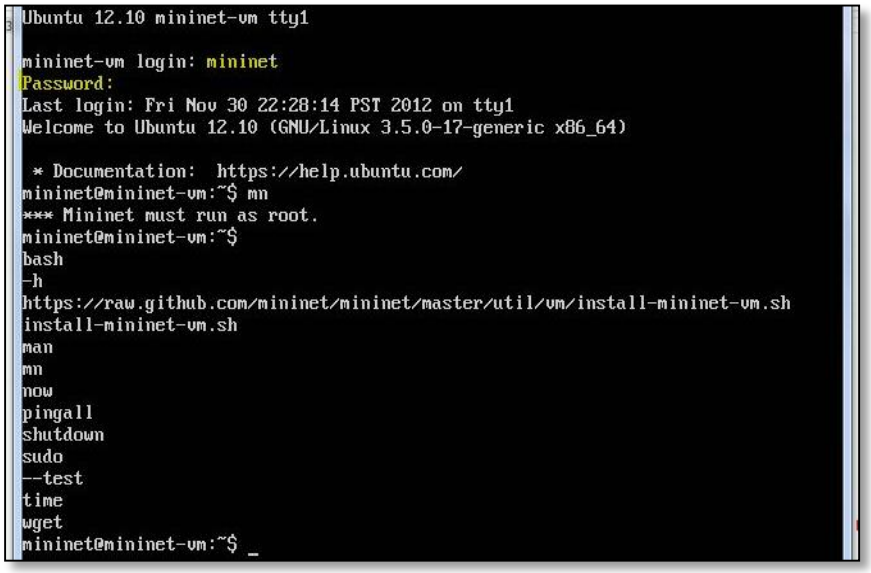
Figura Nº 26 Script de Python

```
#!/usr/bin/python
  from mininet. Topo import Topo
  from mininet.net import Mininet
  from mininet.util import dumpNodeConnections
  from mininet.log import setLogLevel

  class SingleSwitchTopo(Topo):
    "SIWCTH SIMPLE CONECTADO A N HOSTS."
    def build(self, n=2):
      switch = self.addSwitch('s1')
      # Python's range(N) generates 0..N-1
      for h in range(n):
        host = self.addHost('h%s' % (h + 1))
        self.addLink(host, switch)

  def simpleTest():
    "CREACIÓN Y TEST DE LA RED SIMPLE"
    topo = SingleSwitchTopo(n=4)
    net = Mininet(topo)
    net.start()
    print "CONEXIÓN DE HOSTS"
    dumpNodeConnections(net.hosts)
    print "TEST DE CONECTIVIDAD"
    net.pingAll()
    net.stop()

  if __name__ == '__main__':
    # Tell mininet to print useful information
    setLogLevel('info')
    simpleTest()
```

Fuente: Script Python

El acceso a mininet como antes fue mencionada se ingresa con el nombre y usuario y contraseña.

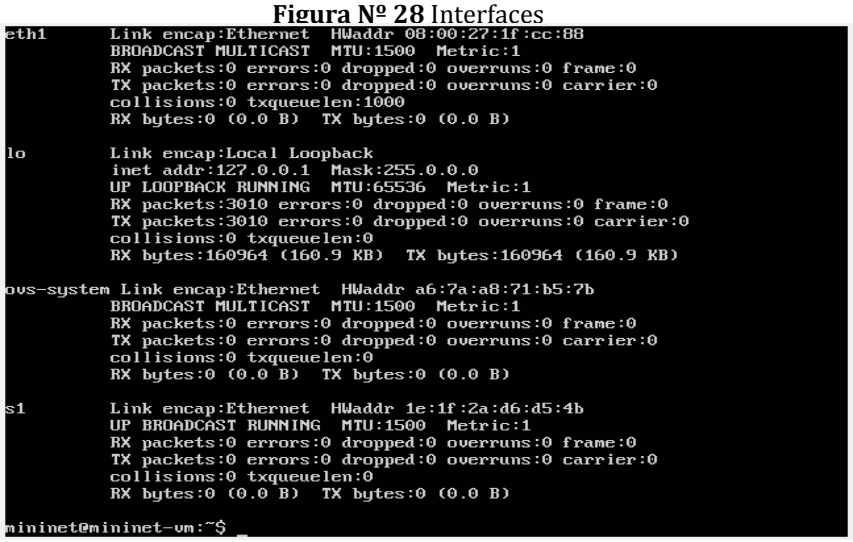
Figura Nº 27 Mininet



Fuente: Elaboración Propia

Verificación de las interfaces de la interfaces las cuales deben estar correctamente configuradas.

Figura Nº 28 Interfaces



Fuente: Elaboración Propia

4.2.1.3.4 Configuración e Instalación de Floodlight

Prerrequisitos:

1. Ubuntu 14.04 o superior
2. Instalar JAVA, Ant
3. Instalar eclipse.
4. El proceso de instalación y configuración de Floodlight es muy intuitivo por lo cual se describirá solamente las partes más importantes.

Descargar los archivos del controlador Floodlight

Para descargar los archivos del controlador Floodlight se utilizaron los siguientes comandos:

$ git clone git://github.com/floodlight/floodlight.git

$ cd floodlight

$ ant eclipse

Crear el FloodlightLaunch para la ejecución de Floodlight

Para ejecutar Floodlight es necesario seguir los siguientes pasos:

a. Click Run

b. Run Configurations

c. Click Derecho Java Application/New

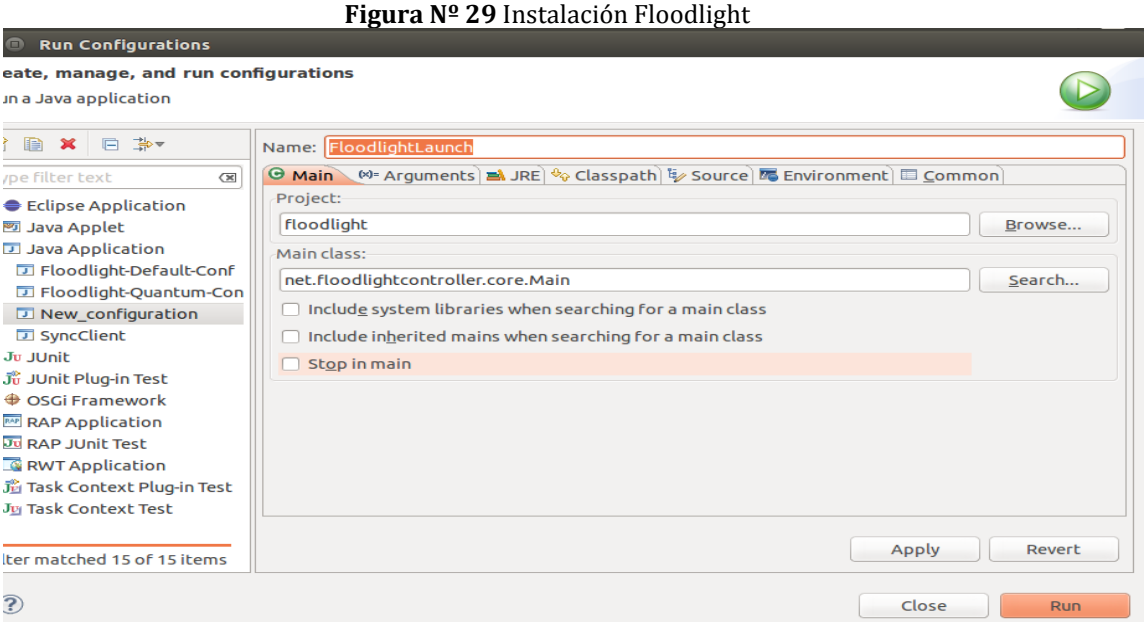
d. Establecer el nombre del nuevo archivo como: **FloodlightLaunch**

e. Establecer el nombre del proyecto como: **floodlight**

f. Establecer el Main como: net.floodlightcontroller.core.Main

g. Click Apply. [11]

Figura Nº 29 Instalación Floodlight



Fuente: Elaboración Propia

4.2.1.4 Pruebas

Las pruebas de aceptación o pruebas funcionales fueron supervisadas por el administrador basándose en los requerimientos tomados de las historias de usuario. En todas las interacciones, cada una de las historias de usuario seleccionadas por el administrador se tuvo una o más pruebas de aceptación, de las cuales se determinó los casos de prueba y la identificación de los errores que serán corregidos. Los errores que se apreciará a menudo son en la ejecución de la página web ya

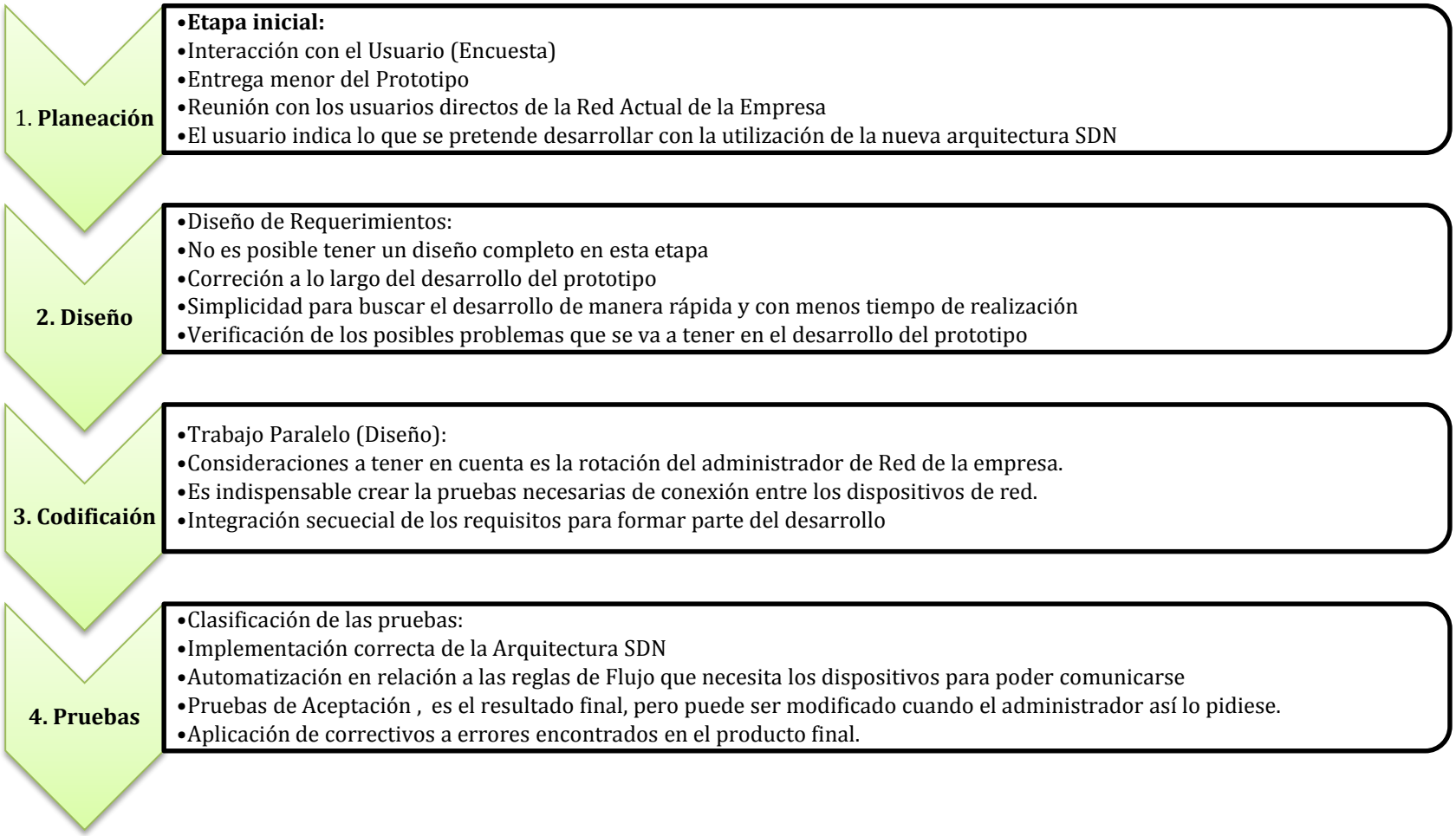
que esta entrará en funcionamiento siempre y cuando el controlador sea ejecutado con antelación, así se evitarán errores en las líneas de comando de la misma.

Es importante resaltar la diferencia entre las pruebas de aceptación que tuvo la página web como un API Rest de OpenFlow y las unitarias como la conexión individual de host (h1 - hN) en lo que al papel del usuario se refiere. Mientras que en las pruebas de aceptación juega un papel muy importante la selección de los casos particulares de configuración de la página web ya que en cada instancia se debe declarar la dirección IP del controlador para que esta entre en funcionamiento y no genere los errores antes mencionados.

Teniendo estos aspectos ya desarrollados se mantuvo la infraestructura en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones con la implementación de SDN; brindando a la red un recurso de fácil manejo y flexible mejorando así la gestión por parte del administrador de la red.

Luego de haber descrito de manera coherente las fases de la metodología aplicada se presenta una visión más clara o resumen de la metodología XP aplicada en este desarrollo que se puede verificar en la Figura N° 30.

Figura Nº 30 Proceso de Desarrollo de la Metodología XP



Fuente: Elaboración Propia

Capítulo 5

Resultados

5.1 Evaluación preliminar

En [39], La empresa Avícola Agoyán cumplió 51 años generando productos de calidad, gracias al aliento y esfuerzo de sus promotores, manejando principalmente la producción de carne y la comercialización de huevos.

La empresa Avícola Agoyán, por su diversidad de actividades tales como la Producción de huevos, pollos y balanceado, desarrolla sus operaciones en la provincia de Tungurahua cantón Ambato, mientras que las actividades administrativas y comerciales se concentran en la provincia de Pastaza en la que se ejecutan los trabajos productivos.

En el Cantón Ambato, existen 9 oficinas en el área administrativa, con 29 computadoras personales instaladas en red; de las cuales 10 computadoras portátiles tienen acceso a internet constantemente para el manejo del departamento de ventas.

La planta de producción en el cantón Puyo, está dividida a su vez en seis plantas de producción de diferentes productos que tienen procesos individuales, las cuales poseen dos oficinas cada una, equipadas con una computadora personal en cada oficina con acceso a internet.

5.1.1 Debilidades de la estructura de redes

Según [39], no se ha hecho un inventario tecnológico con las verdaderas necesidades específicas para cada puesto en la que permita optimizar el uso y manejo de licencias de paquetes informáticos específicos que eviten el uso inadecuado de los recursos de la institución que en muchos de los casos son utilizados para tareas personales.

Cada departamento reporta sus actividades de manera individual, tomando en cuenta sus necesidades individuales que precisan datos concretos de gestión y administración de sus recursos conforme lo establecido en el área administrativa.

El departamento de redes maneja estadísticas en cuanto al empleo de la tecnología que permite entregar informes pormenorizados del uso y manejo de la tecnología, lo que ayuda a respaldar los datos según las condiciones cambiantes del entorno.

La arquitectura actualmente no ha permitido que se emplee las máquinas de manera eficiente, por cuanto se desconoce de las condiciones de uso y manejo de las diferentes herramientas

suministradas a los empleados, como herramienta institucional para el desempeño en cada puesto de trabajo.

El desconocimiento de sistemas que optimizan el uso y manejo de información, hace que se posea temor en relación a la implementación de propuestas eficientes que sean autofinanciables, mediante la redistribución de recursos en condiciones de actualización constante.

5.2 Indicadores de gestión

El administrador de red maneja el tráfico de información, evaluando la necesidad de actualización y mantenimiento del sistema, para ejecutar las actividades aprovechando la tecnología que se emplea. Para este desarrollo el administrador podrá generar nuevos indicadores de gestión de la red como por ejemplo:

a. Tiempo de Login a la nueva arquitectura
b. Velocidad de transmisión
c. Puerto de entrada y salida
d. Tasa de transmisión fallida

5.2.1 Gestión de la red

La gestión de redes abarca muchos aspectos, que pueden sintetizarse en tareas no tan simples como: despliegue, integración y coordinación del hardware, software y los elementos humanos para monitorizar, probar, configurar, analizar, evaluar y controlar los recursos de la arquitectura de red para conseguir niveles de trabajo y de servicio adecuados a los objetivos de la nueva arquitectura desarrollada.

5.2.1.1 Herramientas de gestión de red.

El administrador de red tiene la capacidad de presentar de manera eficiente los informes que sean necesarios para indicar los niveles de tráfico de información presentes en la nueva arquitectura de red con la ayuda de ciertos sistemas de monitorización de servicios. Los cuales son los siguientes:

a. Nagios
b. Wireshark

5.2.1.1.1 Nagios

Este sistema de monitoreo permite al administrador de la red SDN habilitar, identificar y resolver problemas de infraestructura de TI, antes de que afecte los procesos críticos de la empresa. El diseño fue pensado para ser escalable (facilidad de crecer); y que permite extender su funcionalidad con la utilización/creación de extensiones, debe correr en sistemas Linux.

Nagios comprobará que se han especificado todos los objetos correctamente y en el orden siguiente según [41]:

1. Verificar que todos los contactos son grupos de al menos un grupo de contacto.
2. Verificar que todos los miembros de un grupo de contacto son contactos válidos.
3. Verificar que todos los equipos son miembros de al menos un grupo de equipos.
4. Verificar que todos los equipos especificados en un grupo de equipos son equipos válidos.
5. Verificar que todos los equipos tienen al menos un servicio asociado a ellos.
6. Verificar que todos los comandos usados en los servicios y equipos, son válidos.
7. Verificar que todos los comandos usados en los manejadores de eventos de servicios y equipos, son válidos.
8. Verificar que todos los comandos usados para notificaciones de contactos, equipos y servicios, son válidos.
9. Verificar que todos los periodos de tiempos usados para servicios, equipos y contactos, son válidos.
10. Verificar que todos los periodos de tiempos para comprobación de servicios, son válidos

5.2.1.1.2 Wireshark

El administrador utilizará el analizador de protocolos con código libre diseñado y disponible para plataformas Windows y Unix con el objetivo principal de realizar el análisis de tráfico de información presente y brindará soluciones a los problemas de la arquitectura de red.

5.3 Validación del Desarrollo

Luego de haber desarrollado el proyecto se debe presentar el producto final el cual permite tener una arquitectura SDN para la gestión con recursos de estándar abierto; dado este precedente, se tiene un emulador de red llamado Mininet el cual crea redes virtual realistas ejecutados sobre kernel de linux, en la cual se puede simular redes completas sobre una máquina virtual (VM, cloud nativa),

Por razones de carácter económico es de suma importancia realizar este proyecto en Emuladores ya que los equipos que soportan SDN con controlador OpenFlow son demasiado costosos, los mismos que son de muy poco acceso a las configuraciones ya que al momento de este desarrollo se encontró en la Empresa la poca predisposición para obtener Router y Switches configurables para el mejor funcionamiento de esta arquitectura emergente.

5.3.1 Desarrollo mediante Mininet

Este software ayuda profundamente a la investigación y desarrollo de prototipos, pruebas y depuración. La ventaja principal es la de tener una red experimental completa en un solo computador portátil o de escritorio.

Este software es compatible con cualquier tipo de topología de red en la que se incluye un conjunto básico de una topología con parámetros estandarizados, otra de las ventajas principales es la de proporcionar una API de Python que se considera sencillo y escalable para la creación de sistemas de redes y experimentación.

5.3.2 Características de Mininet

- Brinda un almacenamiento de pruebas de red sencillo, barato para desarrollar aplicaciones con el protocolo OpenFlow.
- Trabajo de forma concurrente sobre la misma topología de red para los desarrolladores.
- Respaldo hacia las pruebas de retorno las cuales son de retroalimentación con fácil empaquetado en el sistema.
- Incluye un Interfaz (CLI[15]) para depurar y ejecutar pruebas en toda la infraestructura de red

5.3.3 Infraestructura de red

En la topología de esta red se consigue gracias al código desarrollado en Phyton V 3.4.3 en la cual se debe cumplir los siguientes objetivos:

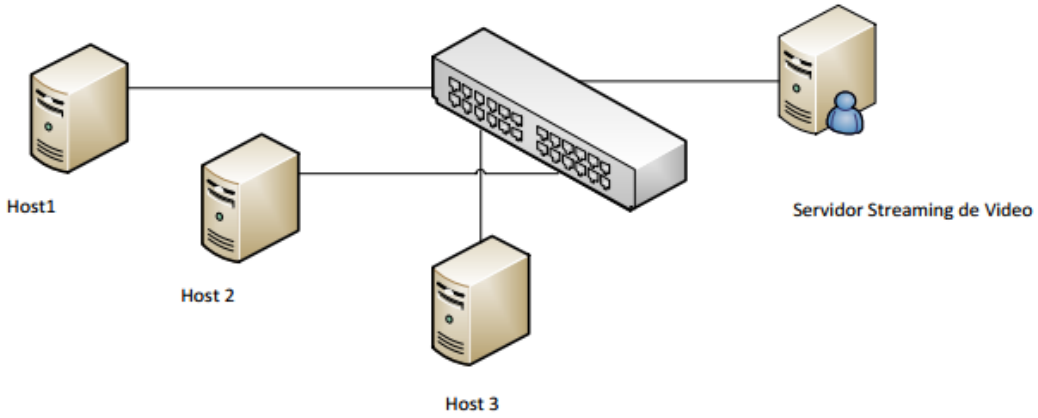
- Programar el prototipo de la red en Phyton
- Realizar pruebas de conectividad entre los dispositivos de red utilizando el protocolo ICMP (Protocolo de Control de Mensajes en internet) con un ping.
- Listar la tabla ARP del switch
- Utilizar la interfaz de línea de comando, para validar información del prototipo creado.

[15] Línea de Comando

5.4 Diseño de la Red

El diseño de la red estará conformado por un servidor como indica la Figura Nº 31.

Figura Nº 31 Diseño de la red SDN



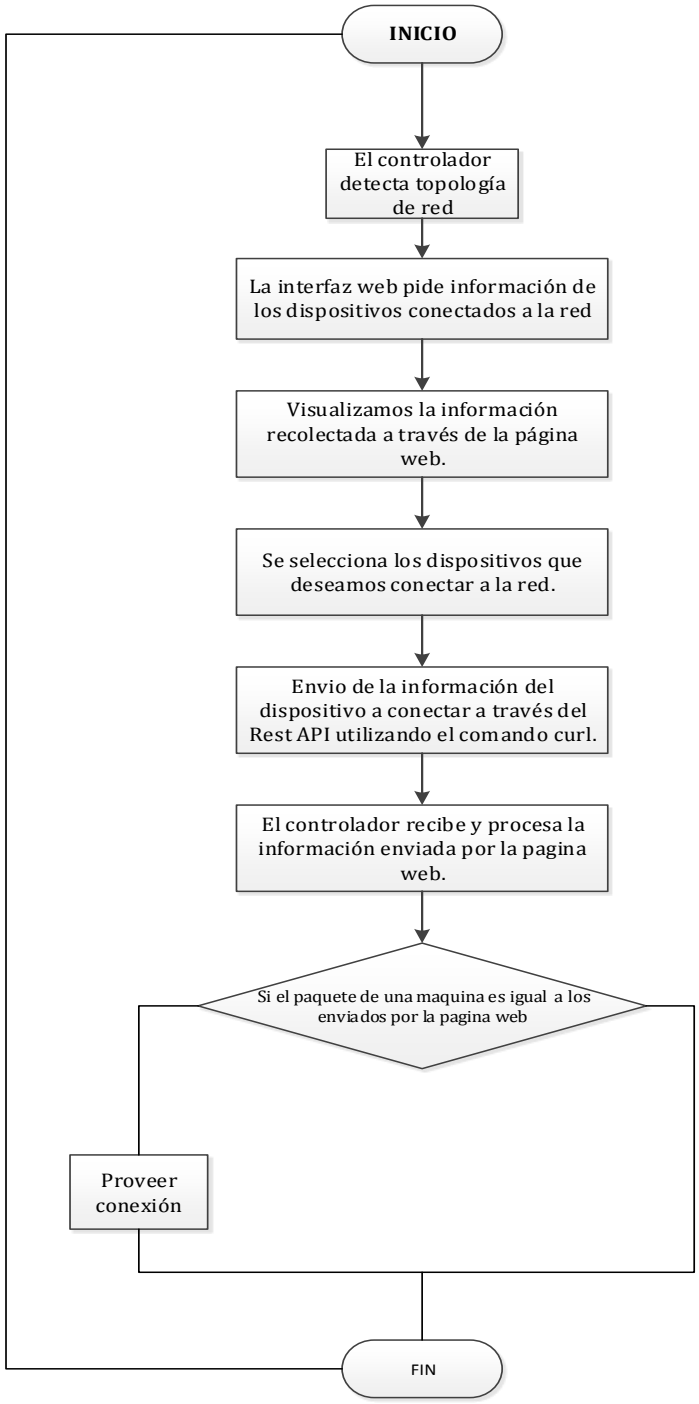
Fuente: Diego Armando Maldonado Hidalgo [8]

5.4.1 Funcionamiento

El módulo desarrollado en el controlador recibe las órdenes para su ejecución por parte de la interfaz Web, la cual a la vez genera peticiones curl utilizando el REST API de Floodlight [27], (servicio que se detalla a continuación) con el objetivo de obtener información de los dispositivos conectados a la red y establecer reglas de flujo para su comunicación.

El funcionamiento se detalla en el diagrama de flujo que se muestra a continuación:

Figura Nº 32 Funcionamiento

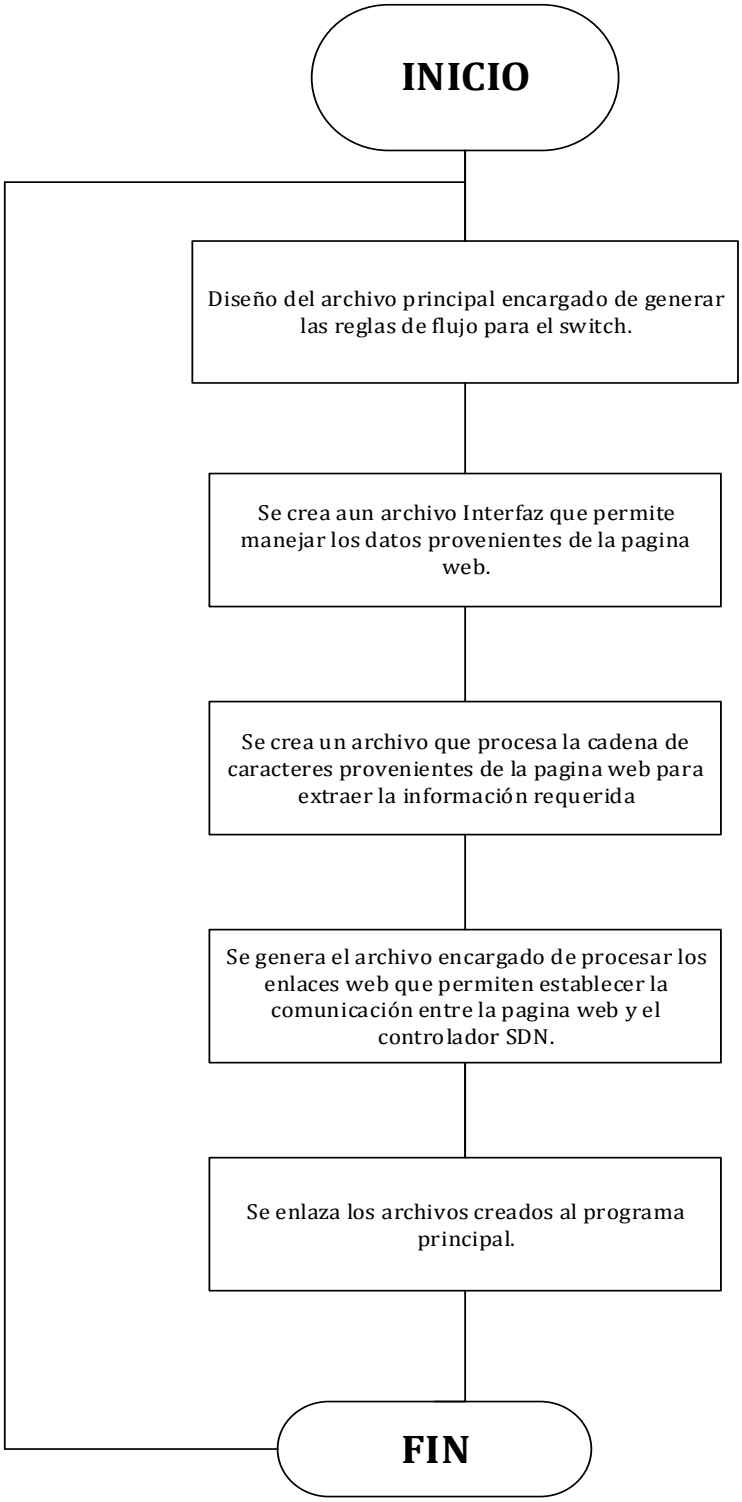


Fuente: Elaboración Propia

5.4.1.1 Diseño del Módulo de Comunicación

Para el diseño del módulo SDN se debe seguir el proceso que se detalla en el flujograma que se muestra a continuación:

Figura Nº 33 Diagrama del Módulo de Comunicación

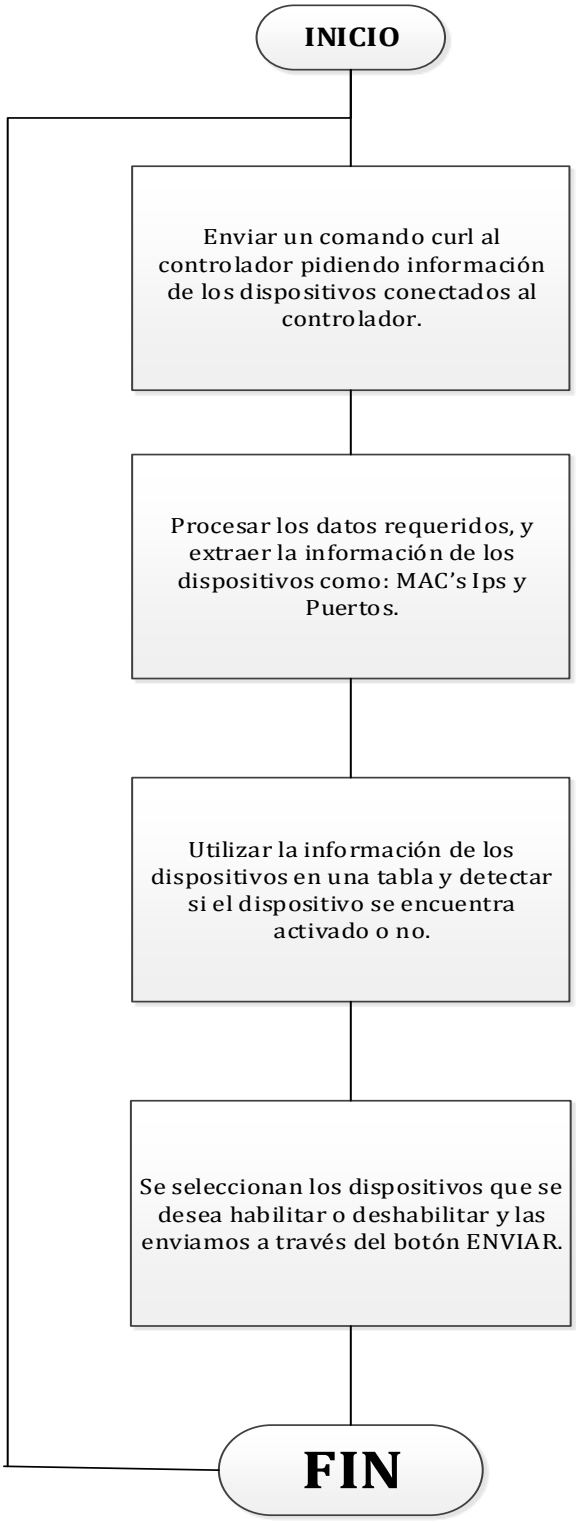


Fuente: Elaboración Propia

5.4.1.2 Creación de la interfaz web

Para la creación de la Interfaz Web se debe seguir el proceso que se detalla en el flujograma que se muestra a continuación:

Figura Nº 34 Flujograma de Creación de la interfaz Web



Fuente: Elaboración Propia

5.4.1.3 Simulación de la Red en MININET

Para la configuración de la red en MININET se siguieron los pasos que se detallan a continuación:

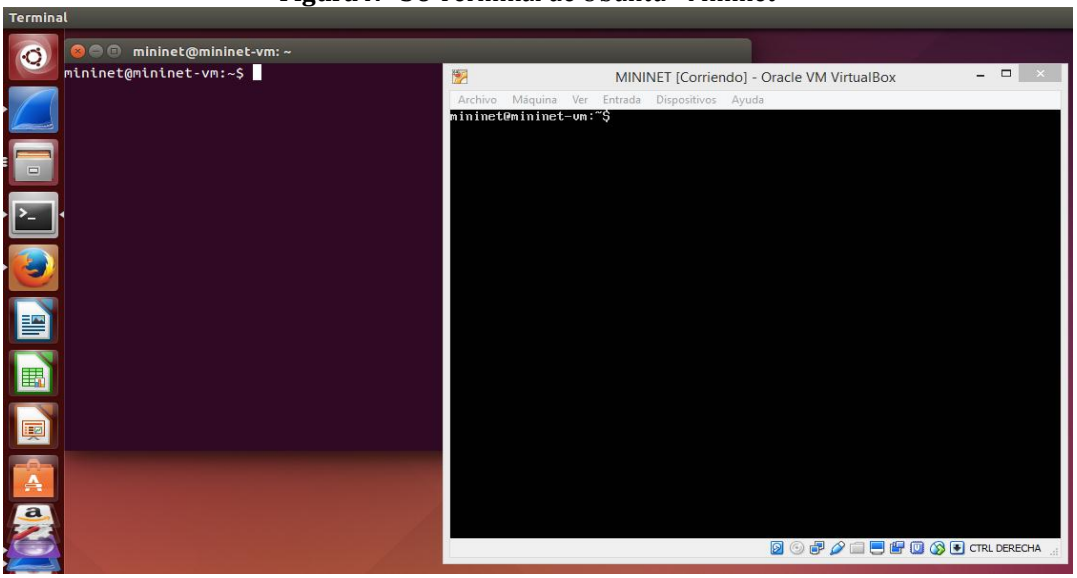
1. Iniciar las máquinas virtuales, tanto Mininet como Ubuntu

Figura Nº 35 Administrador de maquinas virtuales

Fuente: Elaboración Propia

2. Visualización de la terminal de Ubuntu y de la máquina virtual de mininet.

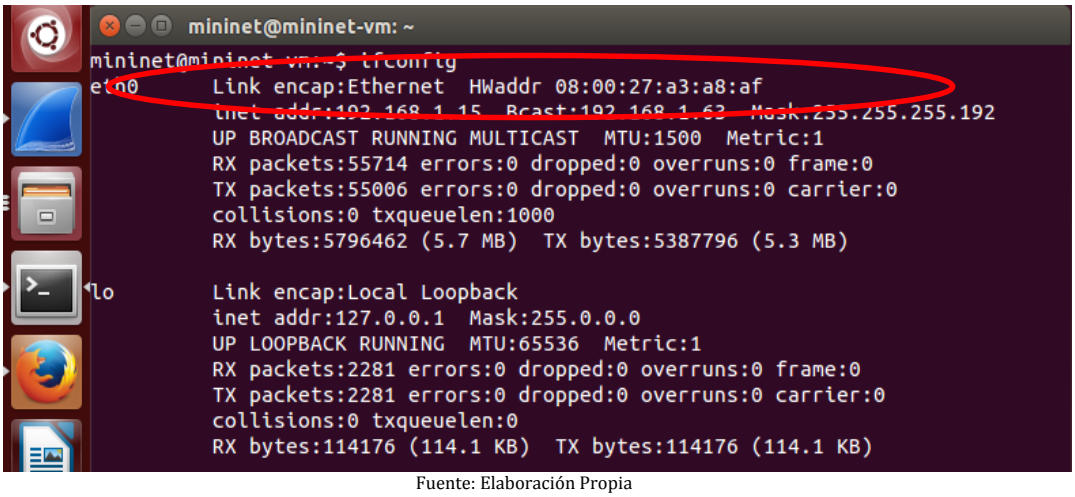
Figura Nº 36 Terminal de Ubuntu - Mininet



Fuente: Elaboración Propia

3. Comando ifconfig para verificar la dirección de red que en este caso es la del controlador

Figura Nº 37 Dirección de la interfaz



Fuente: Elaboración Propia

4. Configuración de red en el controlador en la Terminal de Ubuntu

Figura Nº 38 Configuración de la interfaz

```
VirtualBox:~$ sudo ifconfig eth0 192.168.0.104 netmask 255.255.255.0 up
```

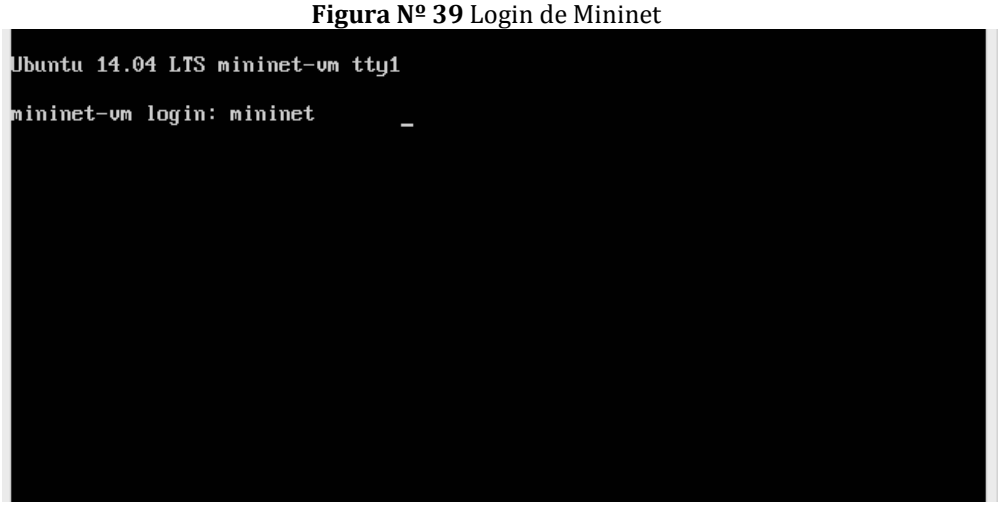
Fuente: Elaboración Propia

5. Acceso a Mininet

Login: mininet

Password: mininet

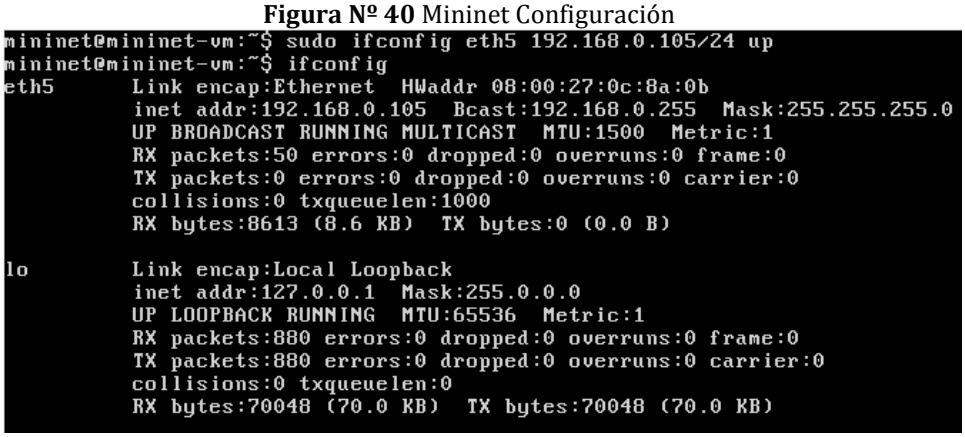
Figura Nº 39 Login de Mininet



Fuente: Elaboración Propia

6. Configuración de red en mininet

Figura Nº 40 Mininet Configuración

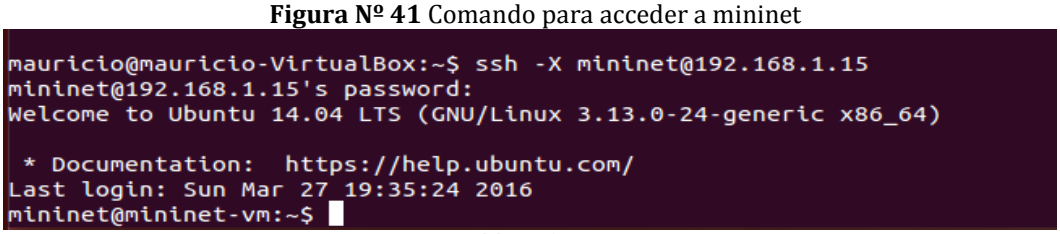


Fuente: Elaboración Propia

7. Acceso a mininet colocando el siguiente comando desde Ubuntu

ssh –X mininet@192.168.1.15

Figura Nº 41 Comando para acceder a mininet

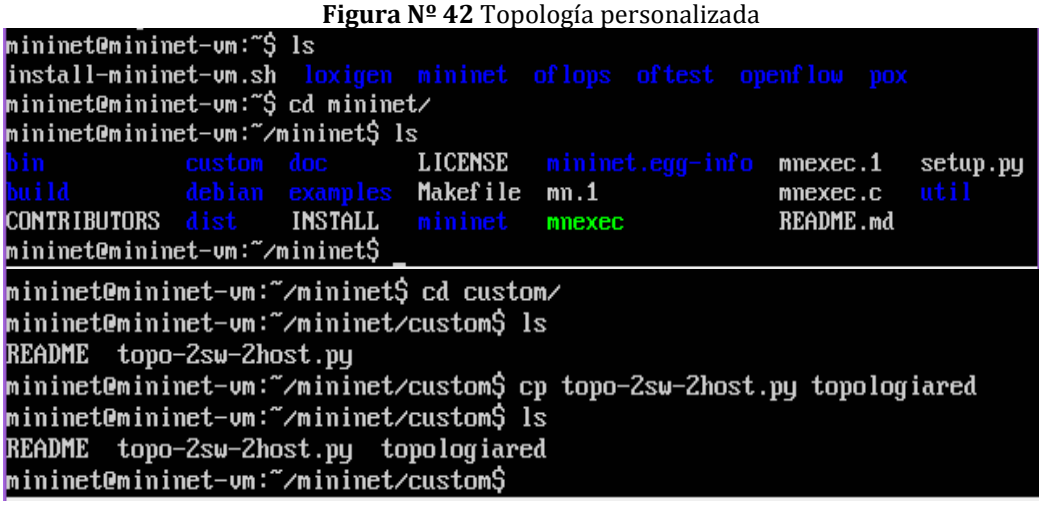


Fuente: Elaboración Propia

8. Creación de la topología de red en mininet.

Para crear una topología personalizada en mininet se debe dirigir a la carpeta **custom** en donde se encuentran almacenadas todas las topologías personalizadas, en la Figura N° 39 se indica como dirigirse a dicha carpeta y la manera de cómo crear el archivo de la topología.

Figura Nº 42 Topología personalizada



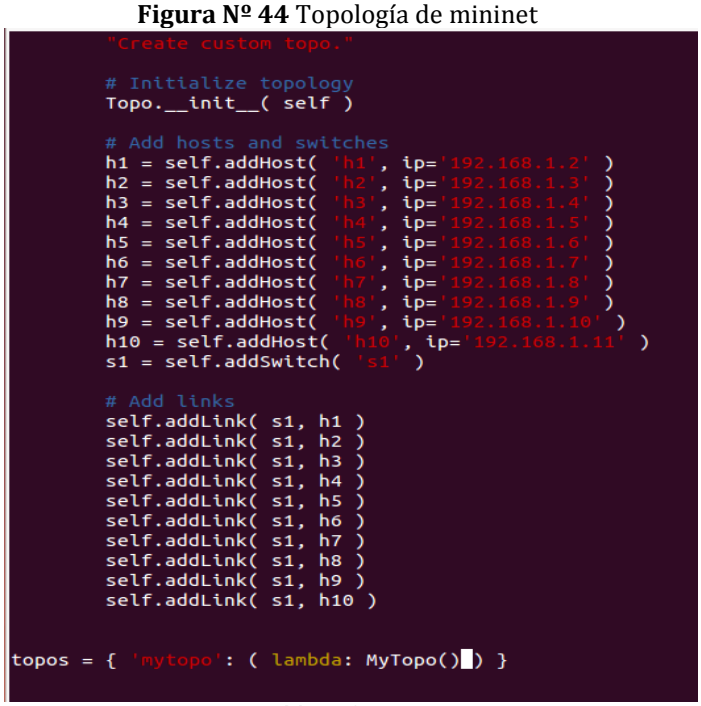
Fuente: Elaboración Propia

9. A continuación se muestra el código de la creación de la topología el cual se lo codifica en el lenguaje de programación Phyton dentro de la máquina de mininet.

Figura Nº 43 Topología de mininet

Fuente: Elaboración Propia

Figura Nº 44 Topología de mininet

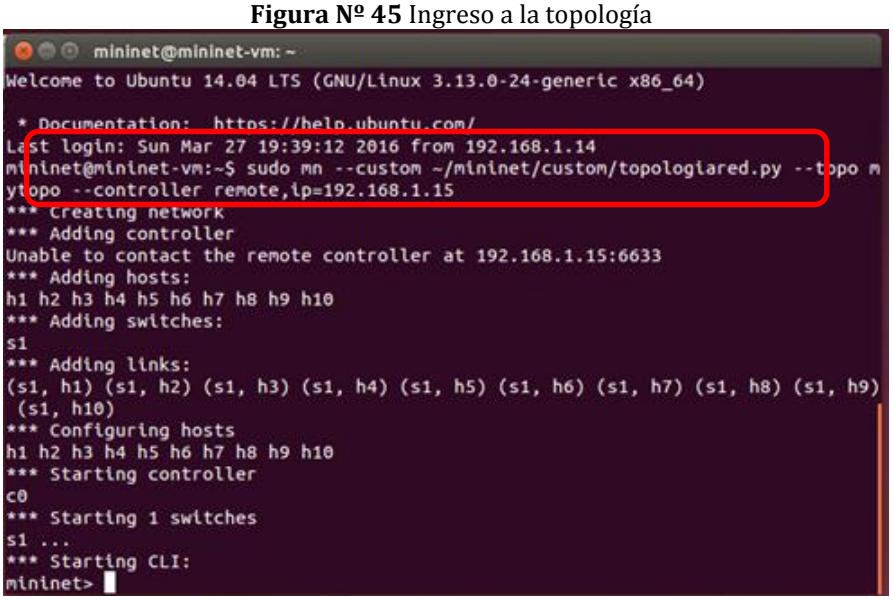


Fuente: Elaboración Propia

10. Acceso al controlador desde la terminal de Ubuntu en la que se visualiza la red y sus enlaces colocando el siguiente comando:

sudo mn –custom ~/mininet/custom/topologiared.py –topo mytopo –controller remote, ip=192.168.1.15

Figura Nº 45 Ingreso a la topología



Fuente: Elaboración Propia

11. Posteriormente se debe arrancar el controlador con el modulo desarrollado y conectarlo con la topología en mininet con la ayuda de Eclipse[16].

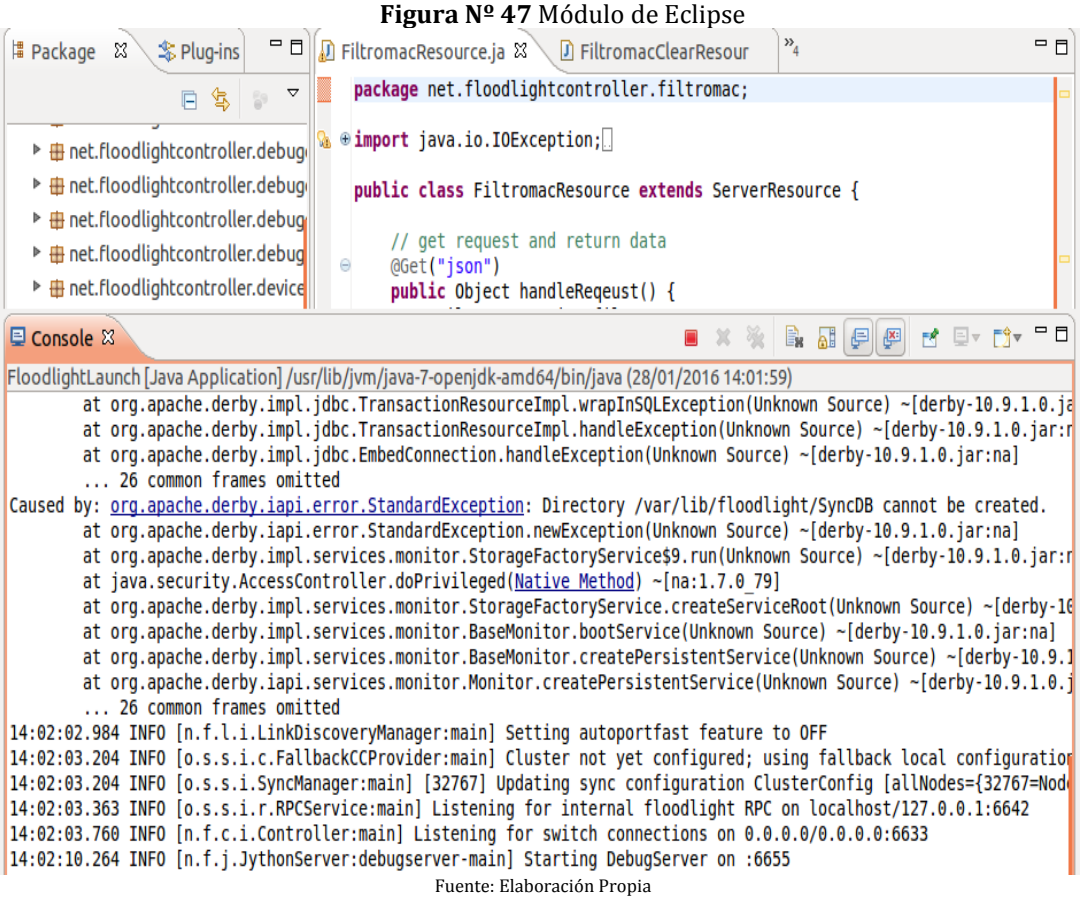
Figura Nº 46 Ejecución de Eclipse

Fuente: Elaboración Propia

[16] Eclipse: Plataforma de software con programación de código abierto

12. Arranque del controlador y el enlace de la topología con el mismo.

Figura Nº 47 Módulo de Eclipse



Fuente: Elaboración Propia

13. Visualización de la información de la red con la ejecución del comando dump y net.

Figura Nº 48 Nodos de la red

```
mininet> dump
<Host h1: h1-eth0:192.168.1.2 pid=3483>
<Host h2: h2-eth0:192.168.1.3 pid=3487>
<Host h3: h3-eth0:192.168.1.4 pid=3489>
<Host h4: h4-eth0:192.168.1.5 pid=3491>
<Host h5: h5-eth0:192.168.1.6 pid=3493>
<Host h6: h6-eth0:192.168.1.7 pid=3495>
<Host h7: h7-eth0:192.168.1.8 pid=3497>
<Host h8: h8-eth0:192.168.1.9 pid=3499>
<Host h9: h9-eth0:192.168.1.10 pid=3501>
<Host h10: h10-eth0:192.168.1.11 pid=3503>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None,s1-eth4:None,s1-eth5:None,s1-eth6:None,s1-eth7:None,s1-eth8:None,s1-eth9:Non
e,s1-eth10:None pid=3508>
<RemoteController{'ip': '192.168.0.107'} c0: 192.168.0.107:6633 pid=3477>
mininet>
```

Fuente: Elaboración Propia

Figura Nº 49 Enlaces de red

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
h3 h3-eth0:s1-eth3
h4 h4-eth0:s1-eth4
h5 h5-eth0:s1-eth5
h6 h6-eth0:s1-eth6
h7 h7-eth0:s1-eth7
h8 h8-eth0:s1-eth8
h9 h9-eth0:s1-eth9
h10 h10-eth0:s1-eth10
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0 s1-eth3:h3-eth0 s1-eth4:h4-eth0 s1-eth5:h5-eth0 s1-eth6:h6-eth0 s1-eth7:h7-eth0 s1-eth8:h8-eth0 s1-eth9
:h9-eth0 s1-eth10:h10-eth0
c0
mininet>
```

Fuente: Elaboración Propia

14. Visualización y descripción breve de la información visualizada en la página web.

En la siguiente Figura N° 50 se indica la información de los dispositivos conectados a la red, y a la vez permite que el administrador pueda otorgar acceso a la red o lo deniegue; todo estos pasos se lo realiza ingresando en el navegador web Firefox 45.0.1 colocando la siguiente dirección ***https://localhost/tesis/tesis.php***.

Figura Nº 50 Interfaz web para la gestión de dispositivos

Fuente: Elaboración Propia

15. Activación de puertos en la página web.

Figura Nº 51 Puertos activados

RED

DE DISPOSITIVOS CONECTADOS: 10 ENVIAR

IP	MAC	PUERTO	HABILITAR	DESHABILITAR	STATUS
192.168.1.10	ae:68:3f:3c:7e:9a	9	☐ON	☐OFF	DESACTIVADO
192.168.1.11	82:a1:12:05:ec:5f	10	☑ON	☐OFF	ACTIVADO
192.168.1.2	ce:c1:94:4c:6c:3e	1	☐ON	☐OFF	DESACTIVADO
192.168.1.3	2e:a5:b6:b0:37:42	2	☑ON	☐OFF	ACTIVADO
192.168.1.4	fa:fe:1d:43:35:89	3	☐ON	☐OFF	DESACTIVADO
192.168.1.5	9a:a8:64:43:bc:31	4	☑ON	☐OFF	ACTIVADO
192.168.1.6	9a:ce:4f:25:0d:43	5	☐ON	☐OFF	DESACTIVADO
192.168.1.7	4a:24:15:3f:79:f8	6	☐ON	☐OFF	DESACTIVADO
192.168.1.8	ee:0a:66:30:0b:e6	7	☐ON	☐OFF	DESACTIVADO
192.168.1.9	5a:92:31:42:46:cf	8	☐ON	☐OFF	DESACTIVADO

Fuente: Elaboración Propia

16. Prueba de conexión entre los dispositivos activados en el terminal de Ubuntu

A continuación se muestra que existe conectividad entre los host que fueron previamente activados.

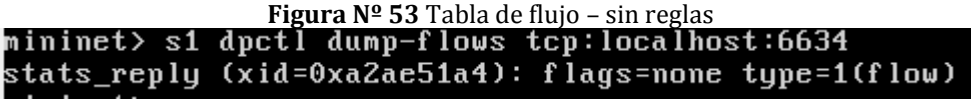
Figura Nº 52 Conexión entre los dispositivos

```
mininet> h2 ping -c3 h10
PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data.
64 bytes from 192.168.1.11: icmp_seq=1 ttl=64 time=88.8 ms
64 bytes from 192.168.1.11: icmp_seq=2 ttl=64 time=0.495 ms
64 bytes from 192.168.1.11: icmp_seq=3 ttl=64 time=0.077 ms

--- 192.168.1.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.077/29.823/88.899/41.773 ms
mininet> h2 ping -c3 h4
PING 192.168.1.5 (192.168.1.5) 56(84) bytes of data.
64 bytes from 192.168.1.5: icmp_seq=1 ttl=64 time=73.2 ms
64 bytes from 192.168.1.5: icmp_seq=2 ttl=64 time=0.424 ms
64 bytes from 192.168.1.5: icmp_seq=3 ttl=64 time=0.070 ms

--- 192.168.1.5 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.070/24.572/73.223/34.401 ms
mininet> h10 ping -c3 h4
PING 192.168.1.5 (192.168.1.5) 56(84) bytes of data.
64 bytes from 192.168.1.5: icmp_seq=1 ttl=64 time=60.4 ms
64 bytes from 192.168.1.5: icmp_seq=2 ttl=64 time=0.395 ms
64 bytes from 192.168.1.5: icmp_seq=3 ttl=64 time=0.073 ms
```

Fuente: Elaboración Propia

17. Visualización de las reglas de flujo en el switch posterior a la activación de puertos a través de la interfaz web, accediendo al terminal de mininet colocando el comando que se indica a continuación

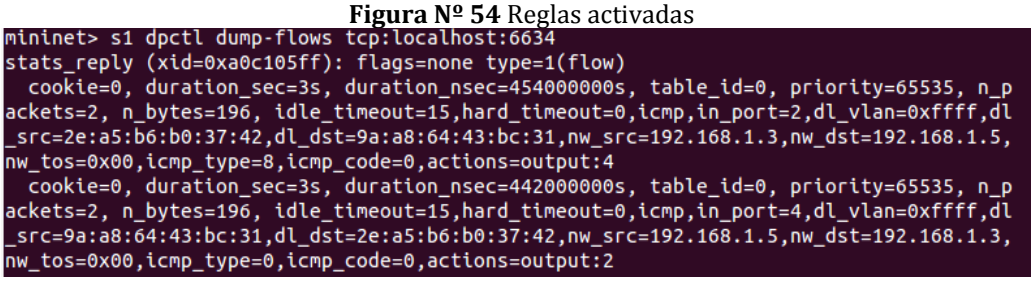
Figura Nº 53 Tabla de flujo – sin reglas



Fuente: Elaboración Propia

18. En contraste con la figura anterior, cuando los dispositivos fueron activados se insertaron automáticamente las reglas para su conexión, lo cual se muestra en la siguiente figura:

Figura Nº 54 Reglas activadas

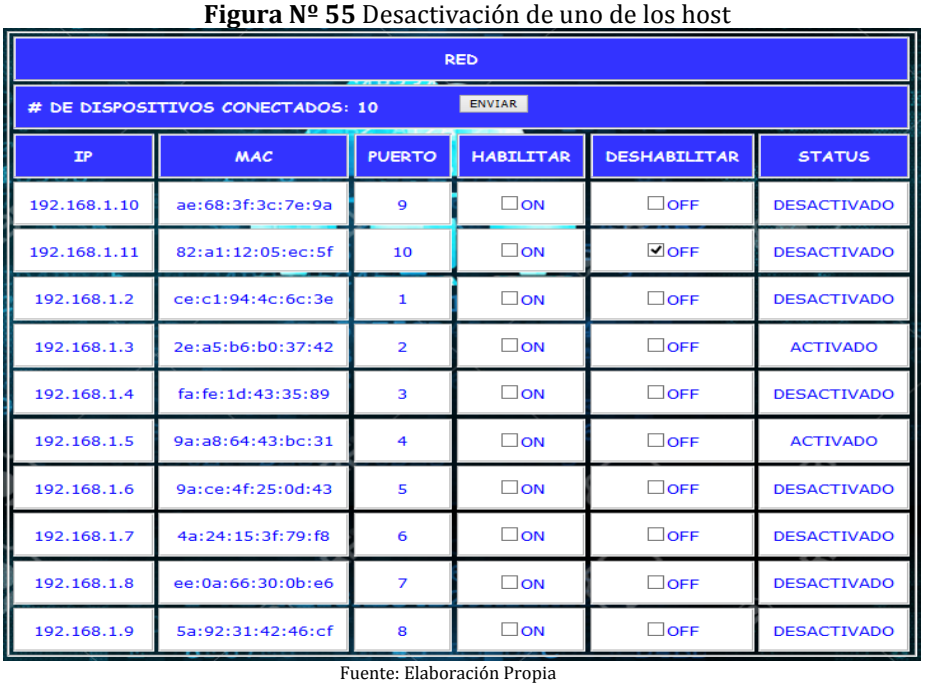


Fuente: Elaboración Propia

19. Prueba de conexión entre los dispositivos desactivados.

A continuación se procede a desactivar uno de los host que fue activado previamente, y se realiza las pruebas de conexión para probar que ha perdido la conexión.

Figura Nº 55 Desactivación de uno de los host



IP	MAC	PUERTO	HABILITAR	DESHABILITAR	STATUS
192.168.1.10	ae:68:3f:3c:7e:9a	9	☐ON	☐OFF	DESACTIVADO
192.168.1.11	82:a1:12:05:ec:5f	10	☐ON	☑OFF	DESACTIVADO
192.168.1.2	ce:c1:94:4c:6c:3e	1	☐ON	☐OFF	DESACTIVADO
192.168.1.3	2e:a5:b6:b0:37:42	2	☐ON	☐OFF	ACTIVADO
192.168.1.4	fa:fe:1d:43:35:89	3	☐ON	☐OFF	DESACTIVADO
192.168.1.5	9a:a8:64:43:bc:31	4	☐ON	☐OFF	ACTIVADO
192.168.1.6	9a:ce:4f:25:0d:43	5	☐ON	☐OFF	DESACTIVADO
192.168.1.7	4a:24:15:3f:79:f8	6	☐ON	☐OFF	DESACTIVADO
192.168.1.8	ee:0a:66:30:0b:e6	7	☐ON	☐OFF	DESACTIVADO
192.168.1.9	5a:92:31:42:46:cf	8	☐ON	☐OFF	DESACTIVADO

Fuente: Elaboración Propia

Figura Nº 56 Conexión de hosts

```
mininet> h2 ping  h10
PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data.
From 192.168.1.3 icmp_seq=1 Destination Host Unreachable
From 192.168.1.3 icmp_seq=2 Destination Host Unreachable
From 192.168.1.3 icmp_seq=3 Destination Host Unreachable
From 192.168.1.3 icmp_seq=4 Destination Host Unreachable
From 192.168.1.3 icmp_seq=5 Destination Host Unreachable
From 192.168.1.3 icmp_seq=6 Destination Host Unreachable
^C
--- 192.168.1.11 ping statistics ---
7 packets transmitted, 0 received, +6 errors, 100% packet loss, time 6009ms
pipe 4
mininet> h4 ping  h10
PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data.
From 192.168.1.5 icmp_seq=9 Destination Host Unreachable
From 192.168.1.5 icmp_seq=10 Destination Host Unreachable
From 192.168.1.5 icmp_seq=11 Destination Host Unreachable
From 192.168.1.5 icmp_seq=12 Destination Host Unreachable
From 192.168.1.5 icmp_seq=13 Destination Host Unreachable
From 192.168.1.5 icmp_seq=14 Destination Host Unreachable
```

Fuente: Elaboración Propia

Como se observa en las figuras, se desactivo el host 10 y se realizó una prueba de conexión con los otros hosts activados, la cual fue fallida comprobando así el correcto funcionamiento de la página web y del módulo desarrollado.

20. Prueba de conexión con todos los dispositivos conectados.

Posteriormente se procede a realizar una prueba de conexión entre todos los dispositivos, para lo cual se activa cada uno de estos como se muestra a continuación.

Figura Nº 57 Prueba de conexión

RED

DE DISPOSITIVOS CONECTADOS: 10 ENVIAR

IP	MAC	PUERTO	HABILITAR	DESHABILITAR	STATUS
192.168.1.10	ae:68:3f:3c:7e:9a	9	☑ON	☐OFF	ACTIVADO
192.168.1.11	82:a1:12:05:ec:5f	10	☑ON	☐OFF	ACTIVADO
192.168.1.2	ce:c1:94:4c:6c:3e	1	☑ON	☐OFF	ACTIVADO
192.168.1.3	2e:a5:b6:b0:37:42	2	☑ON	☐OFF	ACTIVADO
192.168.1.4	fa:fe:1d:43:35:89	3	☑ON	☐OFF	ACTIVADO
192.168.1.5	9a:a8:64:43:bc:31	4	☑ON	☐OFF	ACTIVADO
192.168.1.6	9a:ce:4f:25:0d:43	5	☑ON	☐OFF	ACTIVADO
192.168.1.7	4a:24:15:3f:79:f8	6	☑ON	☐OFF	ACTIVADO
192.168.1.8	ee:0a:66:30:0b:e6	7	☑ON	☐OFF	ACTIVADO
192.168.1.9	5a:92:31:42:46:cf	8	☑ON	☐OFF	ACTIVADO

Fuente: Elaboración Propia

Figura Nº 58 Test para validar el funcionamiento del módulo

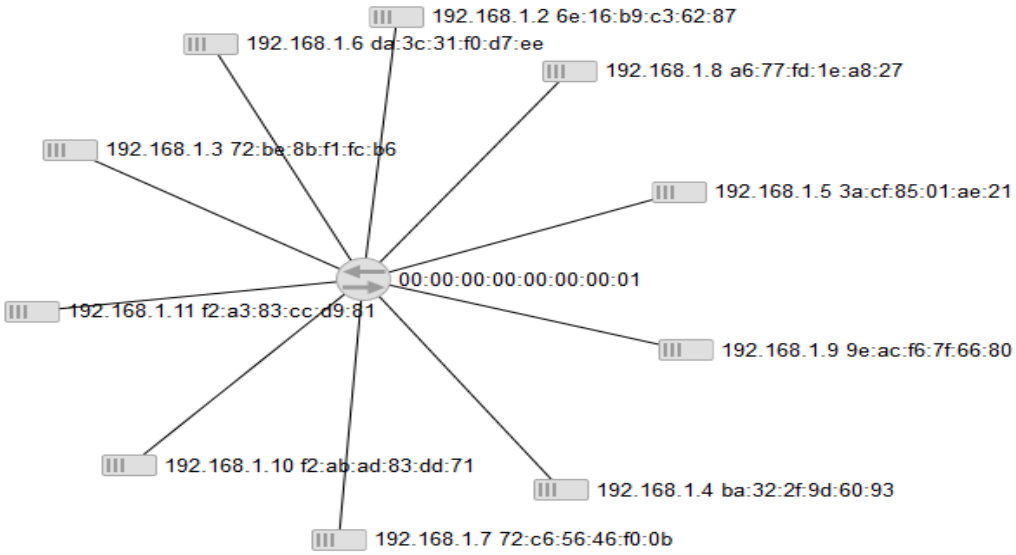
```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Results: 0% dropped (90/90 received)
```

Fuente: Elaboración Propia

21. Visualización de la topología y de las reglas de flujo en la página web de Floodlight.

Ingresando a la dirección: ***http://(dirección_ip_controlador):8080/vi/index.html***

Figura Nº 59 Topología interfaz web floodlight



Fuente: Elaboración Propia

Figura Nº 60 Reglas de Flujo

Flows (4)

Cookie	Priority	Match	Action	Packets	Bytes	Age	Timeout
0	0	port=2, VLAN=-1, src=2e:a5:b6:b0:37:42, dest=9a:a8:64:43:bc:31, ethertype=0x0806, proto=2, src=192.168.1.3, dest=192.168.1.5	output 4	1	42	4 s	15 s
0	-1	port=4, VLAN=-1, src=9a:a8:64:43:bc:31, dest=2e:a5:b6:b0:37:42, ethertype=0x0806, proto=1, src=192.168.1.5, dest=192.168.1.3	output 2	1	42	4 s	15 s
0	-1	port=2, VLAN=-1, src=2e:a5:b6:b0:37:42, dest=9a:a8:64:43:bc:31, ethertype=0x0800, proto=1, IP src port=8, IP dest port=0, src=192.168.1.3, dest=192.168.1.5, TOS=0	output 4	3	294	9 s	15 s
0	-1	port=4, VLAN=-1, src=9a:a8:64:43:bc:31, dest=2e:a5:b6:b0:37:42, ethertype=0x0800, proto=1, IP src port=0, IP dest port=0, src=192.168.1.5, dest=192.168.1.3, TOS=0	output 2	3	294	9 s	15 s

Fuente: Elaboración Propia

Capítulo 6

Conclusiones y Recomendaciones

6.1 Conclusiones

En el presente desarrollo se proporciona una clara visión de lo que significa la implementación de la red definida por software SDN con características, arquitecturas y cambios extremos en las condiciones de la infraestructura de la red.

Gracias a la arquitectura SDN permitió el mejoramiento de la capacidad de gestión por parte del administrador de red mediante el uso de recursos de estándar abierto, permitiendo así que la red sea escalable y flexible al momento de utilizar controladores.

La selección de controladores de la red definida por software SDN se ajustan a las necesidades de virtualización, funcionalidad, confiabilidad, monitorización, rendimiento y seguridad de la red, tal es el caso de OpenFlow.

Se demostró que la gestión se convierte en centralizada para todos los dispositivos de la red en la topología requerida para la toma de correctas decisiones, lo que genera que exista un poco probabilidad de inconsistencias en las reglas de flujos de red.

6.2 Recomendaciones

Para evitar problemas en monitorear la aplicación se debe agregar una base de datos correlacional para almacenar la información de notificación de eventos ocurridos en los distintos equipos de la red, y estos sean considerados para posteriores análisis de la misma.

Para alcanzar un control de la arquitectura de Red granular, se debe analizar y socializar las políticas en las que se verán inmerso los usuarios con sus distintos dispositivos de la red.

Se requiere tomar en cuenta que los prerrequisitos para el desarrollo de este prototipo deben ser los conocimientos básicos sobre el sistema Operativo Linux para así poder crear topologías intrínsecas a la empresa.

Se debe hacer un estudio sobre componentes de Red como son las del controlador, número de hosts, número de switches a los que se puede controlar con un equipo de forma que se eviten colisiones provocando cuellos de botella provocando baja velocidad de transmisión de datos causando así un bajo rendimiento de la red.

Apéndice A

PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR SEDE AMBATO

MAESTRÍA EN GERENCIA INFOMÁTICA

Entrevista a Administradores de Red de la Empresa Avícola Agoyán

1. Indique con una escala de 1 a 5 siendo :

1	2	3	4	5
Muy en desacuerdo	En desacuerdo	Indeciso	De acuerdo	Muy de acuerdo

Su administrador de red le permite optimizar el uso de los paquetes informáticos según las necesidades de cada dependencia.

2. Indique con una escala de 1 a 5 siendo :

1	2	3	4	5
Muy en desacuerdo	En desacuerdo	Indeciso	De acuerdo	Muy de acuerdo

El uso de los recursos tecnológicos que dispone la institución, está en concordancia con los resultados planteados en cada departamento

3. Indique con una escala de 1 a 5 siendo :

1	2	3	4	5
Muy en desacuerdo	En desacuerdo	Indeciso	De acuerdo	Muy de acuerdo

Cree Usted que debe existir estadísticas acerca del uso de las tecnologías de la información y la comunicación

4. Indique con una escala de 1 a 5 siendo :

1	2	3	4	5
Muy en desacuerdo	En desacuerdo	Indeciso	De acuerdo	Muy de acuerdo

El tiempo de respuesta del servidor es el ideal para la toma de decisiones en los procesos de producción y comercialización

5. Indique con una escala de 1 a 5 siendo :

1	2	3	4	5
Muy en desacuerdo	En desacuerdo	Indeciso	De acuerdo	Muy de acuerdo

Si se puede utilizar servicios tecnológicos más eficientes para mejores resultados en la empresa.

Apéndice B

Listado de Direcciones para Descargas de Programas

Tabla Nº 13 Direcciones

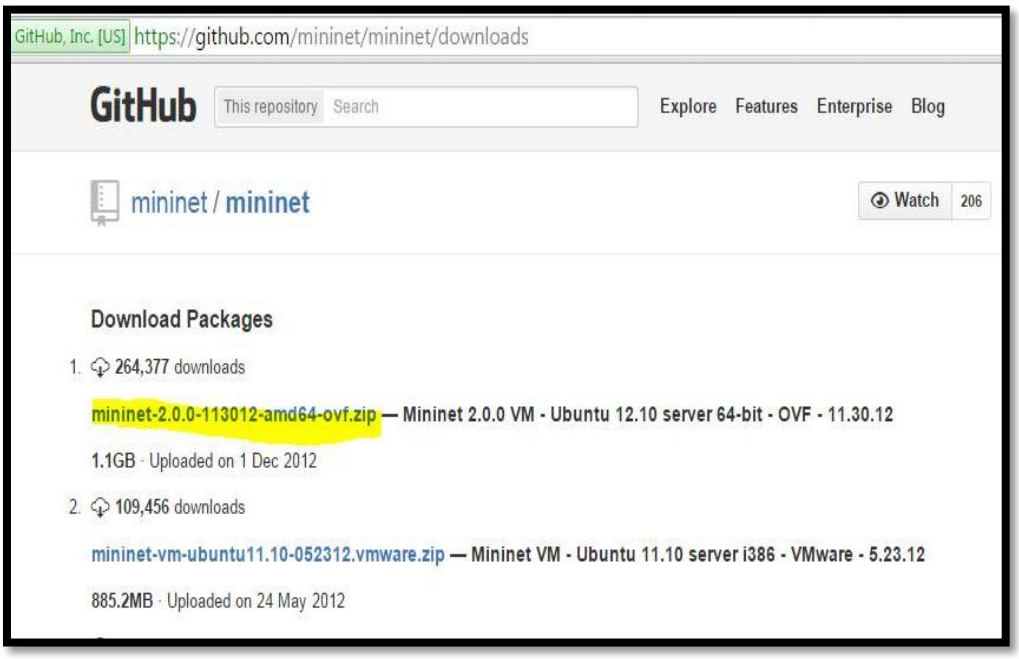
VMware player from "www.vmware.com/products/player/
mininet VM (mininet-2.0.0-113012-amd64-ovf.zip) switch from mininet.org
Ubuntu desktop from http://www.ubuntu.com/download/desktop
OpenDaylight Controller from https://jenkins.opendaylight.org/controller/job/controller-nightly/lastSuccessfulBuild/artifact/opendaylight/distribution/opendaylight/target/
telnet/SSH/Xterm emulator Putty from chiark.greenend.org.uk/~sgtatham/putty/download.html
Download XMING XSERVER for MS Windows from sourceforge.net/projects/xming/

Fuente: SDN and Open Flow for beginners

Apéndice C

Descarga de MININET (virtual switch – virtual machine)

Figura Nº 61 Mininet



Fuente: https://github.com/mininet/mininet/downloads

Apéndice D

Opendaylight Controller Installation

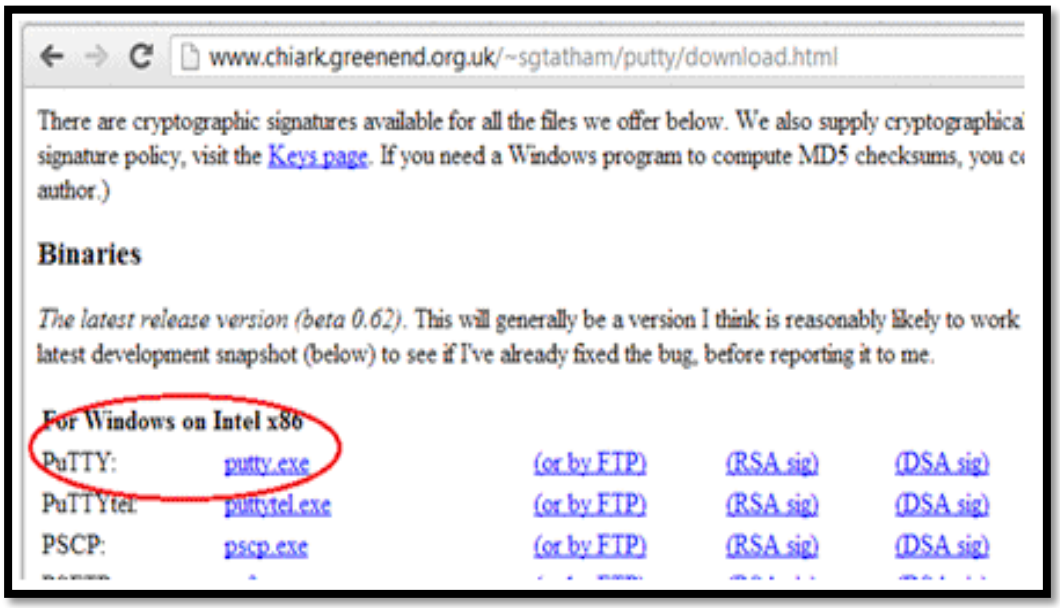
Figura Nº 62 OpenDayLight Controller Installation

Fuente: http://www.opendaylight.org/software/downloads

Apéndice E

Descarga Putty

Figura Nº 63 Descarga PUTTY



Fuente: www.chiark.greenend.org.uk

Apéndice F

Descarga X-Ming

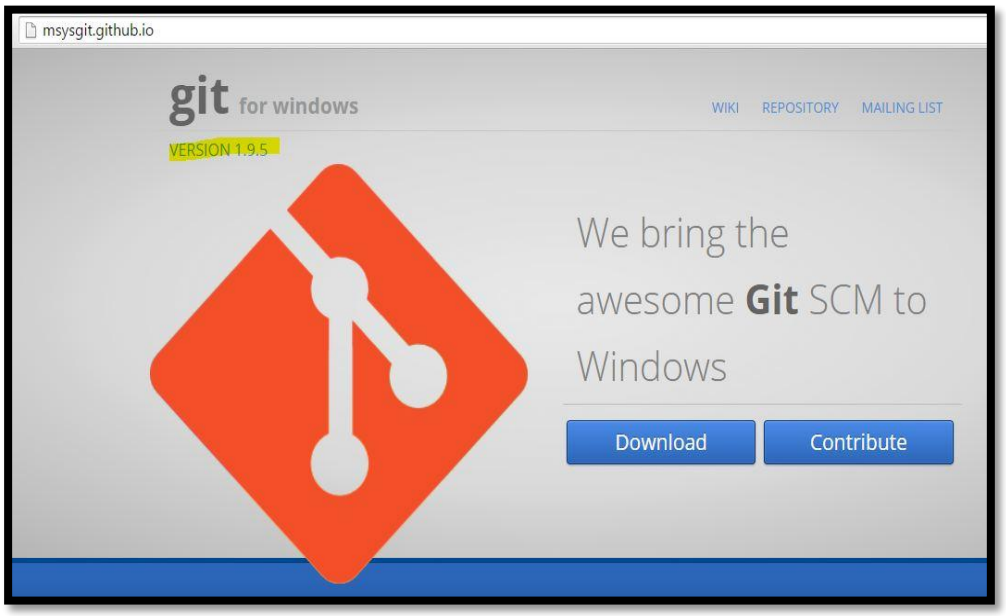
Figura Nº 64 X-MING

Fuente: http://sourceforge.net/projects/xming/?source=directory

Apéndice G

Parámetros Git Setup

Figura Nº 65 Parametro de opendaylight

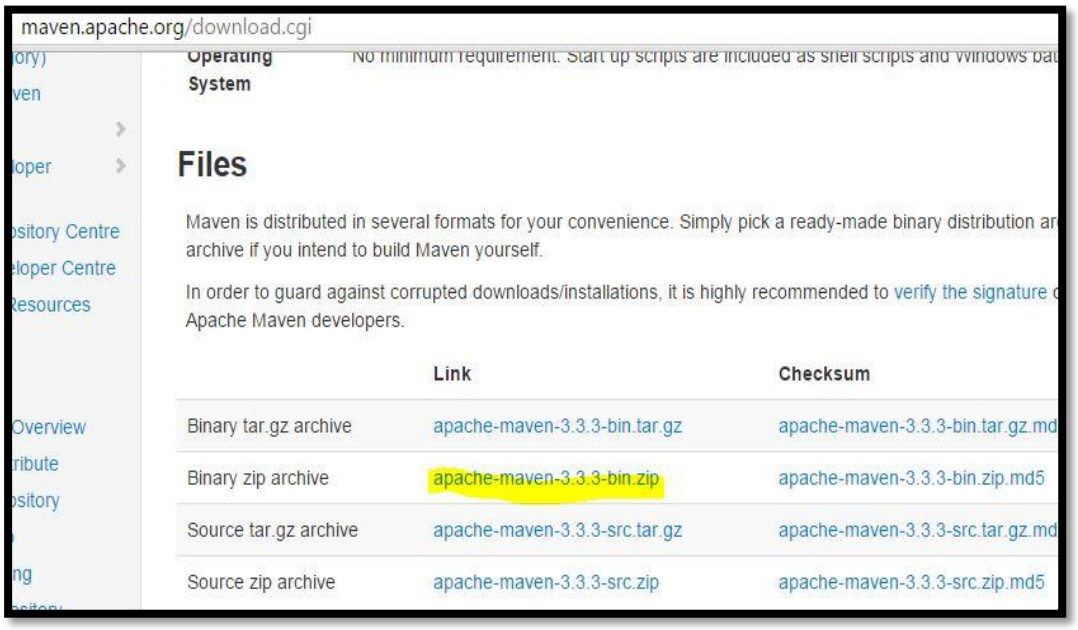


Fuente: http://msysgit.github.io/

Apéndice H

Descarga Apache Maven

Figura Nº 66 Apache Maven

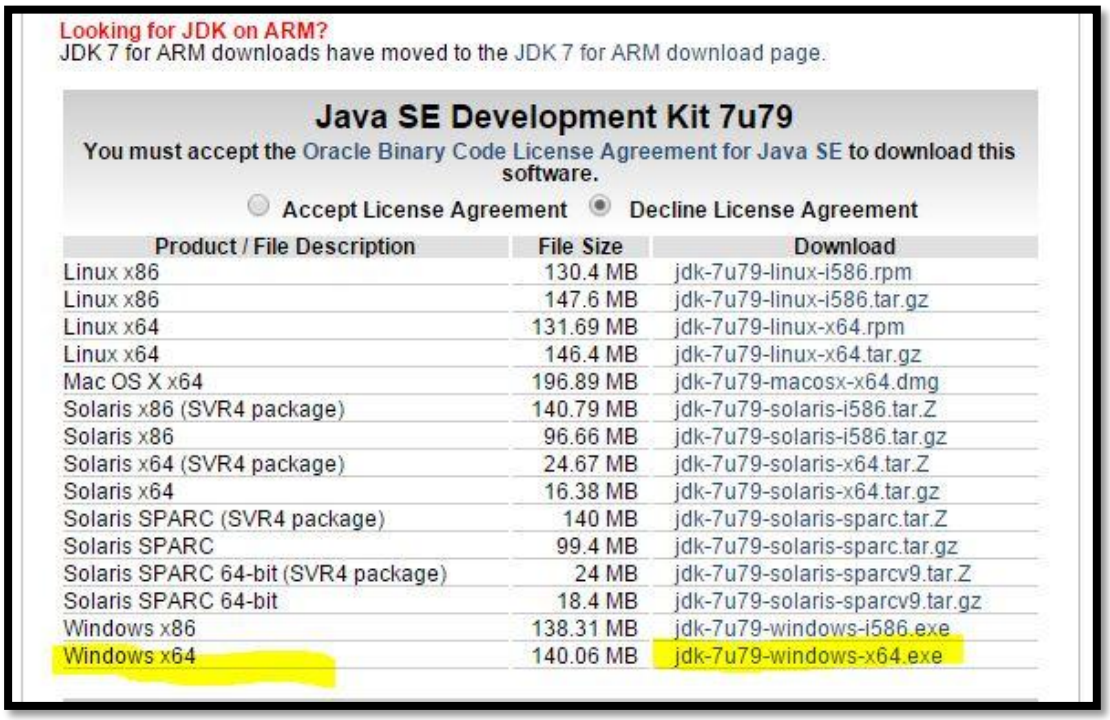


Fuente: http://maven.apache.org/download.cgi

Apéndice I

Descarga JAVA JDK/JRE

Figura Nº 67 JAVA JDK/JRE



Looking for JDK on ARM?
JDK 7 for ARM downloads have moved to the JDK 7 for ARM download page.

Java SE Development Kit 7u79

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

○ Accept License Agreement ◉ Decline License Agreement

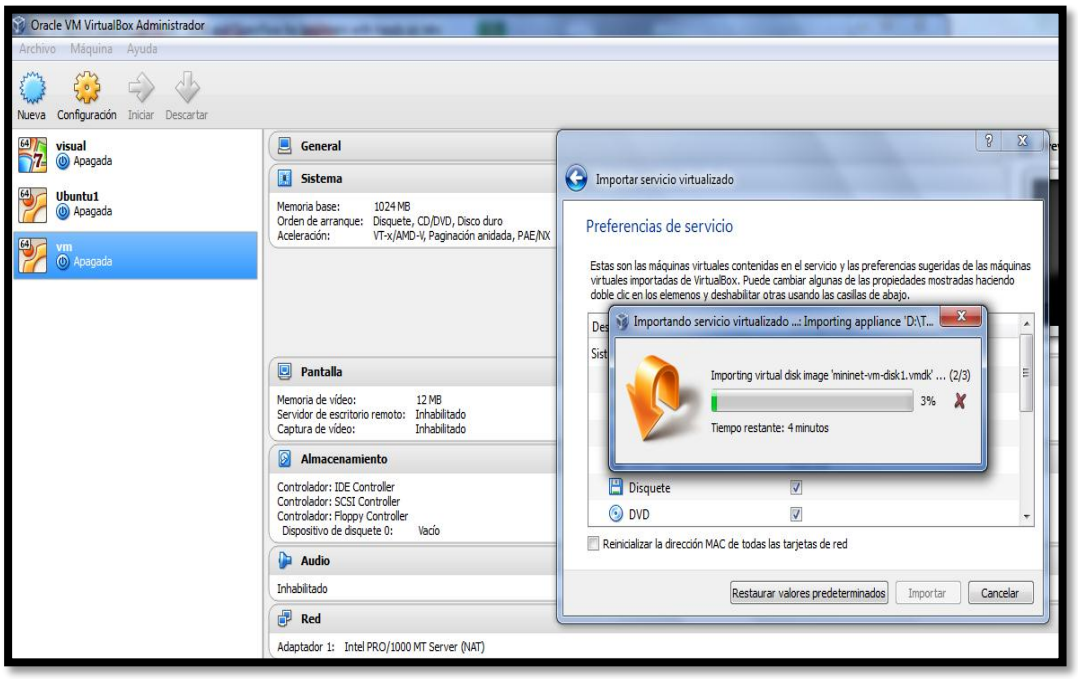
Product / File Description	File Size	Download
Linux x86	130.4 MB	jdk-7u79-linux-i586.rpm
Linux x86	147.6 MB	jdk-7u79-linux-i586.tar.gz
Linux x64	131.69 MB	jdk-7u79-linux-x64.rpm
Linux x64	146.4 MB	jdk-7u79-linux-x64.tar.gz
Mac OS X x64	196.89 MB	jdk-7u79-macosx-x64.dmg
Solaris x86 (SVR4 package)	140.79 MB	jdk-7u79-solaris-i586.tar.Z
Solaris x86	96.66 MB	jdk-7u79-solaris-i586.tar.gz
Solaris x64 (SVR4 package)	24.67 MB	jdk-7u79-solaris-x64.tar.Z
Solaris x64	16.38 MB	jdk-7u79-solaris-x64.tar.gz
Solaris SPARC (SVR4 package)	140 MB	jdk-7u79-solaris-sparc.tar.Z
Solaris SPARC	99.4 MB	jdk-7u79-solaris-sparc.tar.gz
Solaris SPARC 64-bit (SVR4 package)	24 MB	jdk-7u79-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	18.4 MB	jdk-7u79-solaris-sparcv9.tar.gz
Windows x86	138.31 MB	jdk-7u79-windows-i586.exe
Windows x64	140.06 MB	jdk-7u79-windows-x64.exe

Fuente: http://www.oracle.com/technetwork/es/java/javase/downloads/jdk7-downloads-1880260.html

Apéndice J

Importe de MININET

Figura Nº 68 Mininet

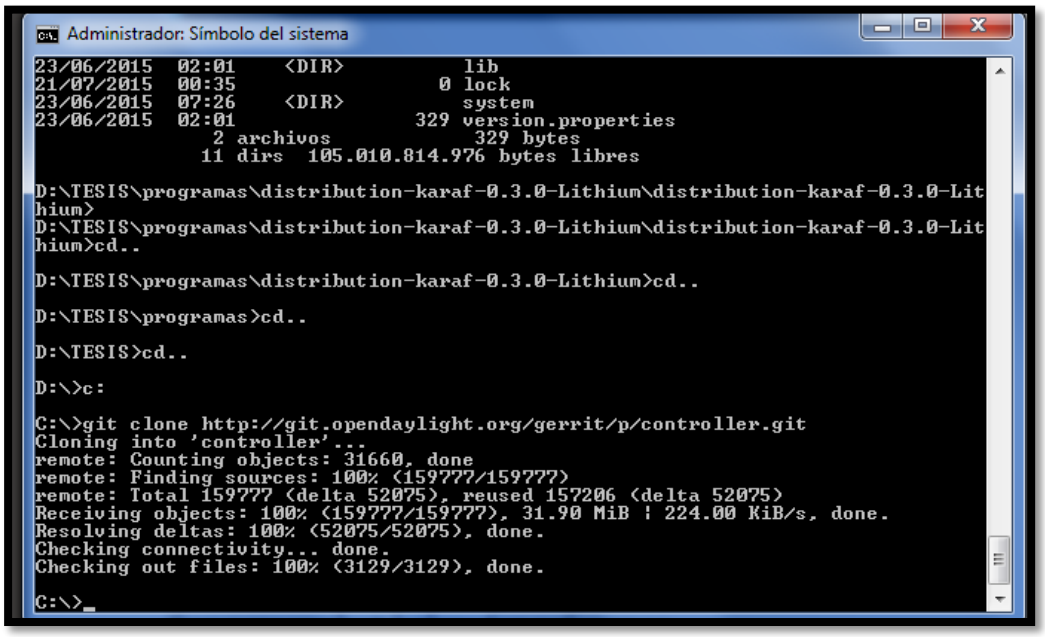


Fuente: Elaboración Propia

Apéndice K

Opendaylight

Figura Nº 69 OpendayLight



Fuente: Elaboración Propia

Apéndice L

Código de la interfaz principal

```
package net.floodlightcontroller.filtromac;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Collections;
import java.util.HashMap;
import java.util.HashSet;
import java.util.List;
import java.util.Map;
import java.util.Set;
import org.openflow.protocol.OFFlowMod;
import org.openflow.protocol.OFMatch;
import org.openflow.protocol.OFMessage;
import org.openflow.protocol.OFPacketIn;
import org.openflow.protocol.OFPacketOut;
import org.openflow.protocol.OFPort;
import org.openflow.protocol.OFType;
import org.openflow.protocol.action.OFAction;
import org.openflow.protocol.action.OFActionDataLayerDestination;
import org.openflow.protocol.action.OFActionNetworkLayerDestination;
import org.openflow.protocol.action.OFActionOutput;
import org.openflow.util.HexString;
import org.openflow.util.U16;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import net.floodlightcontroller.core.FloodlightContext;
import net.floodlightcontroller.core.IFloodlightProviderService;
import net.floodlightcontroller.core.IOFMessageListener;
import net.floodlightcontroller.core.IOFSwitch;
import net.floodlightcontroller.core.IListener.Command;
import net.floodlightcontroller.core.module.FloodlightModuleContext;
import net.floodlightcontroller.core.module.FloodlightModuleException;
import net.floodlightcontroller.core.module.IFloodlightModule;
```

```
import net.floodlightcontroller.core.module.IFloodlightService;
import net.floodlightcontroller.packet.Ethernet;
import net.floodlightcontroller.packet.IPv4;
import net.floodlightcontroller.restserver.IRestApiService;

public class Filtromac implements IFloodlightModule, IOFMessageListener, IFiltromacService {

        protected IFloodlightProviderService floodlightProvider;
        protected IRestApiService restApi;
        private HashSet hs=new HashSet();
        private ArrayList<String> buffer = new ArrayList<String>();
        protected Map<IOFSwitch, Map<Long,Short>> macTables = new HashMap<IOFSwitch,
Map<Long,Short>>();
         protected static Logger log = LoggerFactory.getLogger(Filtromac.class);
        @Override
        public String getName() {
                // TODO Auto-generated method stub
                return "Filtromac";
        }

        @Override
        public boolean isCallbackOrderingPrereq(OFType type, String name) {
                // TODO Auto-generated method stub
                return false;
        }
        @Override
        public boolean isCallbackOrderingPostreq(OFType type, String name) {
                // TODO Auto-generated method stub
                return false;
        }
        private void initMACTable(IOFSwitch sw) {
    Map<Long,Short> macTable = macTables.get(sw);

    if (macTable == null) {
      macTable = new HashMap<Long,Short>();
      macTables.put(sw, macTable);
     }
  }
```

```
private void reglas (IOFSwitch sw, OFPacketIn pi, FloodlightContext cntx) {
                        initMACTable(sw);
                        Map<Long,Short> macTable =macTables.get(sw);
                        OFMatch match = new OFMatch();
                        match.loadFromPacket(pi.getPacketData(),pi.getInPort());
                        System.out.println(match);
                        System.out.println(match.getInputPort());
                        System.out.println(buffer);

                        if (buffer.contains(Short.toString((match.getInputPort()))))
                        {
                                System.out.println("ENTRE A LA CONDICION DEL PUERTO ");
                                macTable.put(Ethernet.toLong(match.getDataLayerSource()),
pi.getInPort());
                                Short
puerto_salida=macTable.get(Ethernet.toLong(match.getDataLayerDestination()));
                                System.out.println(macTable);
                                System.out.println(puerto_salida);

                                if (puerto_salida==null)
                        {
                                        System.out.println("ENTRE A LA CONDICION DEL
PUERTO NULO ");
                                        OFPacketOut po = (OFPacketOut)
floodlightProvider.getOFMessageFactory().getMessage(OFType.PACKET_OUT);
                                po.setBufferId(pi.getBufferId());
                                po.setInPort(pi.getInPort());
                                System.out.println("BUFFER ID= "+pi.getBufferId());

// Establecer las acciones que realizará
                                OFActionOutput action = new
OFActionOutput().setPort(OFPort.OFPP_FLOOD.getValue());
                                po.setActions(Collections.singletonList((OFAction)action));
                                po.setActionsLength((short) OFActionOutput.MINIMUM_LENGTH);
                                System.out.println("VOY A ENVIAR EL PAQUET OUT");

// Evaluar si el switch soporta buffer
                                if (pi.getBufferId() == OFPacketOut.BUFFER_ID_NONE) {
                                  byte[] packetData = pi.getPacketData();
```

```
                    po.setLength(U16.t(OFPacketOut.MINIMUM_LENGTH +
po.getActionsLength() + packetData.length));
                    po.setPacketData(packetData);
                } else {
                    po.setLength(U16.t(OFPacketOut.MINIMUM_LENGTH +
po.getActionsLength()));
                }

                try {
                    System.out.println("VOY A ENVIAR EL PAQUET OUT");
                    sw.write(po, cntx);
                } catch (IOException e) {
                    log.error("Error al escribir el mensaje packet-out", e);
                }
                    }
                    else
                    {
                            System.out.println("ENTRE AL MÉTODO DE AGREGAR
REGLAS");
                            OFFlowMod fm = new OFFlowMod();
                fm.setCommand(OFFlowMod.OFPFC_ADD);
                fm.setBufferId(pi.getBufferId());
                fm.setIdleTimeout((short) 15);
                fm.setMatch(match);

                List<OFAction> actions = new ArrayList<OFAction>();
                OFAction action=new OFActionOutput().setPort(puerto_salida);


fm.setLength((short)(OFFlowMod.MINIMUM_LENGTH+OFActionOutput.MINIMUM_LENGTH));
                actions.add(action);
                fm.setActions(actions);

                try {
                  sw.write(fm, cntx);
                  } catch (IOException e) {
                  log.error("Error al escribir la entrada de flujo", e);
                  }
                  }
```

```
            }
        }


        @Override
        public Command receive(IOFSwitch sw, OFMessage msg, FloodlightContext cntx) {

                OFPacketIn pi = (OFPacketIn) msg;
    reglas(sw,pi, cntx);
                        /*initMACTable(sw);
                        Map<Long,Short> macTable =macTables.get(sw);
                        OFPacketIn pi = (OFPacketIn) msg;
                        OFMatch match = new OFMatch();
                        match.loadFromPacket(pi.getPacketData(),pi.getInPort());

                        if
(buffer.contains(HexString.toHexString(match.getDataLayerSource())))
                        {
                                macTable.put(Ethernet.toLong(match.getDataLayerSource()),
pi.getInPort());
                                Short
puerto_salida=macTable.get(Ethernet.toLong(match.getDataLayerDestination()));

                                if (puerto_salida==null)
                    {
                                        OFPacketOut po = (OFPacketOut)
floodlightProvider.getOFMessageFactory().getMessage(OFType.PACKET_OUT);
                            po.setBufferId(pi.getBufferId());
                            po.setInPort(pi.getInPort());

// Establecer las acciones
                            OFActionOutput action = new
OFActionOutput().setPort(OFPort.OFPP_FLOOD.getValue());
                            po.setActions(Collections.singletonList((OFAction)action));
                            po.setActionsLength((short) OFActionOutput.MINIMUM_LENGTH);

// Evaluar si el switch soporta buffer
                            if (pi.getBufferId() == OFPacketOut.BUFFER_ID_NONE) {
                               byte[] packetData = pi.getPacketData();
```

```
                    po.setLength(U16.t(OFPacketOut.MINIMUM_LENGTH +
po.getActionsLength() + packetData.length));
                    po.setPacketData(packetData);
                } else {
                    po.setLength(U16.t(OFPacketOut.MINIMUM_LENGTH +
po.getActionsLength()));
                }

                try {
                    sw.write(po, cntx);
                } catch (IOException e) {
                    log.error("Error al escribir el mensaje packet-out", e);
                }
                    }
                    else
                    {
                            OFFlowMod fm = new OFFlowMod();
                fm.setCommand(OFFlowMod.OFPFC_ADD);
                fm.setBufferId(pi.getBufferId());
                fm.setIdleTimeout((short) 15);
                fm.setMatch(match);

                List<OFAction> actions = new ArrayList<OFAction>();
                OFAction action=new OFActionOutput().setPort(puerto_salida);

fm.setLength((short)(OFFlowMod.MINIMUM_LENGTH+OFActionOutput.MINIMUM_LENGTH));
                actions.add(action);
                fm.setActions(actions);

                try {
                   sw.write(fm, cntx);
                   } catch (IOException e) {
                   log.error("Error al escribir la entrada de flujo", e);
                   }

                if (pi.getBufferId()==OFPacketOut.BUFFER_ID_NONE)
                {
                        OFPacketOut po = (OFPacketOut)
floodlightProvider.getOFMessageFactory().getMessage(OFType.PACKET_OUT);
```

```
                                OFActionOutput action1 = new
OFActionOutput().setPort(puerto_salida);

po.setActions(Collections.singletonList((OFAction)action1));
                                po.setActionsLength((short)
OFActionOutput.MINIMUM_LENGTH);
                        po.setBufferId(OFPacketOut.BUFFER_ID_NONE);
                        po.setInPort(pi.getInPort());
                        byte[] packetData = pi.getPacketData();
                                po.setLength(U16.t(OFPacketOut.MINIMUM_LENGTH +
po.getActionsLength() + packetData.length));
                                po.setPacketData(packetData);

                                try {
                            sw.write(po, cntx);
                            } catch (IOException e) {
                            log.error("Error al escribir la entrada de flujo", e);
                            }

                    }
                      }
                }*/
                        return Command.CONTINUE;
        }
        @Override
        public Collection<Class<? extends IFloodlightService>> getModuleServices() {
                Collection<Class<? extends IFloodlightService>> l = new ArrayList<Class<?
extends IFloodlightService>>();
                l.add(IFiltromacService.class);
                return l;
        }
        @Override
        public Map<Class<? extends IFloodlightService>, IFloodlightService> getServiceImpls() {
                Map<Class<? extends IFloodlightService>, IFloodlightService> m = new
HashMap<Class<? extends IFloodlightService>,IFloodlightService>();
                m.put(IFiltromacService.class,this);
                return m;
        }
        @Override
```

```
	public Collection<Class<? extends IFloodlightService>> getModuleDependencies() {
		Collection<Class<? extends IFloodlightService>> l = new ArrayList<Class<?
extends IFloodlightService>>();
		l.add(IFloodlightProviderService.class);
		l.add(IRestApiService.class);
		return l;
	}
	@Override
	public void init(FloodlightModuleContext context)throws FloodlightModuleException {
		floodlightProvider = context.getServiceImpl(IFloodlightProviderService.class);
		restApi = context.getServiceImpl(IRestApiService.class);
	}
	@Override
	public void startUp(FloodlightModuleContext context)throws FloodlightModuleException
{
		restApi.addRestletRoutable(new FiltromacWebRoutable());
		floodlightProvider.addOFMessageListener(OFType.PACKET_IN, this);
	}
	//---------------------------------------------------------------------
	// Métodos para visualizar, guardar, y eliminar las direcciones mac
	//---------------------------------------------------------------------
	@Override
	public String getBuffer() {
		String ret;
		String key;

		// "{\"key1\":\"value1\",\"key2\":\"value2\"}"
		ret = "{";
		for (int i=0; i < buffer.size(); i++) {
			key = buffer.get(i);
			ret += "\"" + key + "\""+ ":";
			ret += "\"\"";
			if (i < (buffer.size() - 1))
				ret += ",";
		}
		ret += "}";

		return ret;
	}
```

```
	@Override
	public void showBuffer() {
		for (String entry: buffer) {
			System.out.println(entry);
		}
		}
	@Override
	public void setBuffer(String key) {
		buffer.add(key);
		hs.addAll(buffer);
		buffer.clear();
		buffer.addAll(hs);
		System.out.println("ESTOY DENTRO DEL SETBUFFER");
		System.out.println(buffer);
	}
	@Override
	public void removeEntry(String key) {
		System.out.println("ESTOY DENTRO DE eliminar entrada");
		System.out.println(key);

		System.out.println(buffer);

		for (int i=0; i < buffer.size(); i++) {
			if ((buffer.get(i)).equals(key)) {
				buffer.remove(i);
				return;
			}
		}
	}

	@Override
	public void removeAllEntries() {
		buffer.clear();
	}

}
```

Apéndice M

Código de la interfaz web

CÓDIGO DE LA INTERFAZ WEB

```
<html>
  <head>
    <title>CONTROL DE RED</title>
    <link rel="stylesheet" type="text/css" href="estilos.css">
    <meta http-equiv="Autor" content="M. MURILLO" >
    <script type="text/javascript"></script>
    <script src="includes/jquery.js"></script>
  </head >
    <body background="img/fondo4.jpg">
    <font face="verdana"color="white">
    <h1 align="center">CONTROL DE RED</h1>

  <?php
    $inf="";
    $cont=0;
```

Solicito información al controlador de los dispositivos conectados a la red a través del comando curl haciendo uso del servicio rest de floodlight. Se establece la dirección web donde se encuentra alojado el controlador, señalando información de los dispositivos (/wm/device/).

```
    $url= "http://192.168.1.15:8080/wm/device/";
```

Genero la petición curl.

```
$ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $url);
    curl_setopt($ch, CURLOPT_HEADER, 0);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
```

Guardo la información que retorna el controlador dentro de la variable (inf) la cual esta expresada como una cadena de caracteres

```
$inf=curl_exec($ch);
 curl_close($ch);
```

Genero tres variables denominadas como h1, h2 y h3 las cuales servirán para procesar la información contenida dentro de la cadena de texto de la variable inf.

```
$h=$inf;
$h1=$inf;
$h2=$inf;
```

Se utiliza el comando (substr_count) para contar cuantas veces de repite la palabra (mac) con el objetivo de determinar el número de dispositivos conectados a la red.

```
$ndisp=substr_count($inf, 'mac');
echo "</br>";

$ip1="";
```

Cuando se determinó el número de dispositivos conectados a la red se procede a procesar la cadena de texto de la variable inf con la finalidad de extraer los pedazos de información necesarios para otorgar acceso a la red tales como:

- **Direcciones ip**
- **Direcciones mac**
- **Puertos**

```
for($i=0;$i<$ndisp;$i++)
{
```

Determina la posición de las palabras (mac,ipv4,port) para poder extraer sus respectivos valores

```
  $p=strpos($h, 'mac');
  $p1=strpos($h1, 'ipv4');
  $p2=strpos($h2, 'port');
```

Obtenemos el valores de las direcciones mac.

```
  $rest = substr($h, $p+7);
  $m=substr($rest,0,17);
```

Obtenemos los valores de los puertos del switch a los cuales se encuentran conectados cada host.

```
$rest2 = substr($h2, $p2+6);

if (substr($rest2,1,1)==',')
{
   $port=substr($rest2,0,1);
}
else
{
   $port=substr($rest2,0,2);
}
```

Obtenemos las direcciones Ips de los host.

```
$rest1 = substr($h1, $p1+7);
$ip=substr($rest1,0,1);
if ($ip=='"')
{
   $rest1=substr($h1, $p1+8);
   $pip=strpos($rest1, '"');
   $ip1=substr($rest1,0,$pip);
}
```

Almacenamos los parámetros extraídos en vectores para las direcciones ip, mac y puertos

```
   $ips[$i]=$ip1;
   $macs[$i]=$m;
   $ports[$i]=$port;
   $h=$rest;
   $h1=$rest1;
   $h2=$rest2;
}
```

O**rdenamos la información adquirida para su posterior visualización.**

```
asort($ips);
$k=0;
foreach ($ips as $key ) {
     $ips1[$k]=$key;
     $k++;
```

```
}

for ($k1=0;$k1<$ndisp;$k1++) {
      for ($k2=0;$k2<$ndisp;$k2++) {
        if($ips1[$k1]==$ips[$k2])
        {
          $macs1[$k1]=$macs[$k2];
          $ports1[$k1]=$ports[$k2];
        }
      }
}
```

Establecemos el url que nos permitirá consultar los dispositivos que están habilitados para que se comuniquen dentro de la red.

```
$urlc = "http://192.168.1.15:8080/wm/filtro/addmac | python -mjson.tool";
```

Genero la petición curl de consulta.

```
$ch2 = curl_init();
curl_setopt($ch2, CURLOPT_URL, $urlc);
curl_setopt($ch2, CURLOPT_HEADER, 0);
curl_setopt($ch2, CURLOPT_RETURNTRANSFER, TRUE);

$inf1=curl_exec($ch2);
curl_close($ch2);
      $nha=substr_count($inf1, ':');
$stx=substr($inf1,2,2);

if(substr_count($stx, '"')==1)
{
  $st[0]=substr($inf1,2,1);
}
else
{
  $st[0]=substr($inf1,2,2);
}

for ($k3=1;$k3<$nha;$k3++) {
```

```
   $p4=strpos($inf1, ',');
   $rest4 = substr($inf1, $p4+2);
   $xf=substr($rest4,0,2);
   if(substr_count($xf, '"')==1)
   {
      $con=1;
   }
   else
   {
      $con=2;
   }
      $inf1=$rest4;
   $st[$k3]=substr($rest4,0,$con);
}
    $f=$nha;

while($f<$ndisp){
   $st[$f]="0";
   $f++;
}

asort($st);
$k5=0;
foreach ($st as $key ) {
      $st1[$k5]=$key;
      $k5++;
}

$sw=0;

for ($l2=0;$l2<$ndisp;$l2++) {
   for ($l3=0;$l3<$ndisp;$l3++) {

      if($ports1[$l2]==$st1[$l3])
      {
         $sw=1;
      }

   }
```

```
        if($sw==1){
           $status[$l2]="ACTIVADO" ;
        }
        else
        {
           $status[$l2]="DESACTIVADO";
        }
        $sw=0;

    }
```

Consultamos los dispositivos activados.

```
    $inf1=curl_exec($ch2);
    curl_close($ch2);
```

Detectamos el número de host activados.

```
    $nha=substr_count($inf1, ':');
    $st[0]=substr($inf1,2,1);

    for ($k3=1;$k3<$nha;$k3++) {
       $p4=strpos($inf1, ',');
       $rest4 = substr($inf1, $p4+2);
       $da=substr($rest4,0,1);
       $inf1=$rest4;
       $st[$k3]=$da;
    }
```

Visualizamos el número de host activados y desactivados.

```
    $f=$nha;

    while($f<$ndisp){
       $st[$f]="0";
       $f++;
    }

    asort($st);
    $k5=0;
```

```
foreach ($st as $key ) {
    $st1[$k5]=$key;
    $k5++;
}
$sw=0;

for ($l2=0;$l2<$ndisp;$l2++) {
  for ($l3=0;$l3<$ndisp;$l3++) {

    if($l2+1==$st1[$l3])
    {
      $sw=1;
    }
  }
    if($sw==1){
      $status[$l2]="ACTIVADO" ;
    }
    else
    {
      $status[$l2]="DESACTIVADO";
    }
  $sw=0;
}
```

Enviamos la información de los dispositivos q deseemos conectar a la red a través del comando curl.

```
if ($_POST){
  $npa="";
  foreach ($_POST['check'] as $value) {

    $psend=$value;
    //echo $psend;
    $url1 = "http://192.168.1.15:8080/wm/filtro/addmac";

    if (substr_count($psend, 'd')==0) {

      $data = array($psend => $npa);
      $data_string = json_encode($data);
```

```
            $ch1=curl_init($url1);
            curl_setopt($ch1, CURLOPT_CUSTOMREQUEST, "POST");
            curl_setopt($ch1, CURLOPT_POSTFIELDS, $data_string);
            curl_setopt($ch1, CURLOPT_HTTPHEADER,
            array('Content-Type:application/json','Content-Length: ' . strlen($data_string)));
            curl_setopt($ch1, CURLOPT_RETURNTRANSFER, TRUE);
             $result1 = curl_exec($ch1);
            curl_close($ch1);
          }
else
          {
           //  echo "</br>";
            $pd=strpos($psend, 'd');
            $portd=substr($psend,0,$pd);
            //echo "puerto a desactivar </br>";
            //echo $portd;

            $data = array($portd => $npa);
            $data_string = json_encode($data);

            $ch1=curl_init($url1);
            curl_setopt($ch1, CURLOPT_CUSTOMREQUEST, "DELETE");
            curl_setopt($ch1, CURLOPT_POSTFIELDS, $data_string);
            curl_setopt($ch1, CURLOPT_HTTPHEADER,
            array('Content-Type:application/json','Content-Length: ' . strlen($data_string)));
            curl_setopt($ch1, CURLOPT_RETURNTRANSFER, TRUE);

            $result1 = curl_exec($ch1);
            curl_close($ch1);
          }
       }
    }
```

Creamos la interfaz para presentar la información.

```
    echo "<form name=\"formulario1\" method=\"POST\" action=\"tesis.php\">";
    echo "</br>";
    echo "<hr with='3' size='2' color='yellow'>";
```

```
    echo "<table border=5 align='center' cellspacing='5' cellpadding='15'>";
    echo "<tr><th colspan=\"6\" >RED</th></tr>
      <tr><th align='left' colspan=\"6\">#  DE DISPOSITIVOS CONECTADOS: $ndisp</th></tr>

      <tr>
      <th align='center'>IP</th>
      <th>MAC</th>
      <th>PUERTO</th>
      <th>HABILITAR</th>
      <th>DESHABILITAR</th>
      <th>STATUS</th>
      </tr>";

    for ($l=0;$l<$ndisp;$l++) {
      echo "<tr> ";
      echo "<td align='center'>".$ips1[$l]."</td>";
      echo "<td align='center'>".$macs1[$l]."</td>";
      echo "<td align='center'>".$ports1[$l]."</td>";
      echo "<td align='center'><input type=\"checkbox\" name=\"check[]\" value='$ports1[$l]'
with=\"200\">ON</td>";
      $pdes[$l]=$ports1[$l]."d";
      echo "<td align='center'><input type=\"checkbox\" name=\"check[]\" value='$pdes[$l]'
with=\"200\">OFF</td>";
      echo "<td align='center'>".$status[$l]."</td>";
      echo "</tr>";
    }

    echo"<div id=\"boton\">";
    echo "<input type=\"submit\"value=\"ENVIA\" with=\"200\">";
    echo "</div>";
  ?>
    </font>
  </body>
</html>
```

REFERENCIA-S

[1] S. Azodolmolky, «Software Defined Networking with OpenFlow,» de *Software Defined Networking with OpenFlow*, L. Place, Ed., Birmingham, Packt Publishing Ltd., 2013, p. 8.

[2] I. Bernal, «Prototipos de redes definidas por software,» Escuela Politécnica Nacion, Quito, 2010.

[3] C. F. M. Murillo, «Desarrollo de un prototipo de Red,» Ambato, 2014.

[4] Y. Chimeo, «SDN: ¿un cambio superficial?,» *en Route,* nº 23, pp. 2-3, Noviembre 2013.

[5] Cisco, «¿Pueden las redes definidas por software (SDN) mejorar la rentabilidad del operador?,» San José, 2014.

[6] O. N. Foundation, Enero 2015. [En línea]. Available: https://www.opennetworking.org/about/onf-overview.

[7] I. Citrix Systems, «SDN 101: Introducción a Software Defined Networking,» Fort Lauderdale, FL, EUA, 2014.

[8] H. D. A. Maldonado, «Diseño e implementación de una aplicación de red bajo la arquitectura SDN,» de *Diseño e implementación de una aplicación de red bajo la arquitectura SDN*, P. U. JAVERIANA, Ed., Bogotá, PONTIFICIA UNIVERSIDAD JAVERIANA, 2014, pp. 11-12.

[9] A. L. J. Arizmendi, Mayo 2015. [En línea]. Available: http://luisarizmendi.blogspot.com/2013/11/software-defined-network-sdn.html.

[10] H. Ó. Roncero, «Software Defined Networking,» CATALUNYA, s/n.

[11] R. A. V. Núñez, «Red Definida por Software (SDN) en base a una Infraestructura de Software de Libre Distribución,» s/n, Ambato, 2015.

[12] C. Rothenberg y M. Ribeiro, «OpenFlow e redes definidas por software: um novo paradigma de controle e inovação em redes de pacotes,» Campinas, 2010.

[13] S. Azodolmolky, «Software Defined Networking with OpenFlow,» de *Software Defined Networking with OpenFlow*, L. Place, Ed., Birmingham, Pack Publishing Ltd., 2013, p. 9.

[14] J. Metzler, Mayo 2015. [En línea]. Available: http://cioperu.pe/articulo/11606/explicacion-tecnica-software-defined-networking-sdn/.

[15] G. Ferro, «SDN Use Case: Firewall Migration in the Enterprise,» Etherealmin, S7n s/n 2008 . [En línea]. Available: http://etherealmind.com/sdn-use-case-firewall-migration-in-the-enterprise/. [Último acceso: 20 12 2015].

[16] O. N. Foundation, Noviembre 2014. [En línea]. Available: https://www.opennetworking.org/about/onf-overview.

[17] O. N. Foundation, «OpenFlow Switch Specification,» ONF TS-001, 2009.

[18] J. Ungerman, «Openflow,» s/n.

[19] A. Gämperli, Septiembre 2012. [En línea]. Available: http://www.csg.ethz.ch/education/lectures/ATCN/hs2012/schedule/hwpres/Gaemperli_slides.pdf.

[20] C. d. S. d. Cisco, «Introducción al Rapid Spanning Tree Protocol [protocolo de árbol de expansión rápida] (802.1w),» Document ID: 24062, s/n, 2015.

[21] N. Figuerola, «SDN – Redes definidas por Software,» 2013.

[22] M. G. R. De Tejada, «Evaluación de Controladores OpenFlow,» de *"Evaluación de Controladores OpenFlow*, Tesis de Maestría, 2012, pp. 23-24.

[23] A. Astudillo, G. Dreier, L. Bertholo y L. Tarouco, «Controladores para OpenFlow,» 16 Noviembre 2012. [En línea]. Available: http://eventos.redclara.net/indico/getFile.py/access?contribId=0&resId=4&materialId=slides&confId=197.. [Último acceso: 20 12 2015].

[24] B. Lantz, N. Handigol, B. Heller y V. Jeyakumar, «Introduction to Mininet,» Mininet Project, [En línea]. Available: https://github.com/mininet/mininet/wiki/Introduction-to-Mininet. [Último acceso: 20 12 2015].

[25] V. W. A. Velásquez, «Emulación de una red definida por software utilizando MiniNet,» 2015 . [En línea]. Available: https://www.academia.edu/5730624/Emulaci%C3%B3n_de_una_red_definida_por_software_utilizando_MiniNet. [Último acceso: 20 12 2015].

[26] M. Team, «http://mininet.org,» Octopress, [En línea]. Available: http://mininet.org/overview/. [Último acceso: 20 12 2015].

[27] Atlassian, «https://floodlight.atlassian.net,» 9 Julio 2014. [En línea]. Available: https://floodlight.atlassian.net. [Último acceso: 24 Enero 2016].

[28] C. N. d. Planificación, Plan Nacional para el Buen Vivir 2013-2017, Primera ed., S. T. d. S. Nacional, Ed., Quito: Consejo Nacional de Planificación, 2013.

[29] RedCLARA, RedCLARA: Nombre, voz e instrumento de la colaboración en América Latina, Primera ed., P. M. J. López, Ed., Brussels: Unit G2 Regional Programmes, 2013.

[30] S. M. Kerner, MArzo 2014. [En línea]. Available: http://www.enterprisenetworkingplanet.com/datacenter/why-facebook-does-sdn.html.

[31] E. M. González, «FUNDAMENTOS DE TOTALIDAD Y HOLISMO EN LAS COMPETENCIAS PARA LA,» *Laurus - Revista de Educación,* vol. XIII, nº 24, pp. 338-354, Mayo 2007.

[32] F. D. G. Morillo, «La Implementación de un Prototipo de Red Definida por Software (SDN) empleando una solución basada en Software,» Quito, 2014.

[33] B. Espino y F. Luis, «VIRTUALIZACIÓN DE REDES COMO ELEMENTO CLAVE PARA CLOUD COMPUTING,» Costa Rica, 2009.

[34] Cisco, «Private Cloud Computing for Enterprises: Meet the Demands of High Utilization and Rapid Change,» Usa, 2009.

[35] J. D. Wells, «Extreme Programming Project,» XP Extreme Programming, s/d s/m 2000. [En línea]. Available: http://oness.sourceforge.net/proyecto/html/ch05s02.html. [Último acceso: 4 Enero 2016].

[36] T. L. M. Echeverry y C. L. E. Delgado, «Caso Práctico de la Metdología Ágil XP al Desarrollo de Software,» de *Caso Práctico de la Metdología Ágil XP al Desarrollo de Software*, U. E. Herrera, Ed., Pereira, Facultad de Ingeniería: Eléctrica, Electrónica, Física y Ciencias de la Computación, 2007, p. 28.

[37] W. Hidalgo, *Avícola Agoyán,* Ambato, Tungurahua: s/e, 2015.

[38] J. Pérez, Interviewee, *Entrevista al Administrador de Redes.* [Entrevista]. 14 Octubre 2014.

[39] J. D. McCabe, «Network Analysis, Architecture, and Design,» de *Network Analysis, Architecture, and Design*, S/C, Morgan Kaufmann, 2007, p. 49.

[40] M. Amalvy, V. Antipani y I. Gallardo, «Modelo Jerárquico de una Base de Datos,» s/n, La Patagonia, 2015.

[41] D. A. Briones, «Networking and Internet Technologies,» http://blogs.salleurl.edu/, 17 Mayo 2013. [En línea]. Available: http://blogs.salleurl.edu/networking-and-internet-technologies/alta-redundancia-y-disponibilidad-i/. [Último acceso: 23 Junio 2015].

[42] D. M. Domínguez y Z. A. J. Morán, «Evaluación de Nagios para Linux,» Escuela Politécnica de Cáceres, Extremadura, 2003.

[43] G. Nazareno y L. Albares, «Implantación y puesta a punto de la infraestructura de un cloud computing privado para el despliegue de servicios en la nube,» Sevilla, 2011.

[44] J. S. Turner y D. E. Taylor, «Diversifying the Internet,» de *IEEE GLOBECOM* , Saint Louis, 2005.

[45] T. Szigeti, C. Hattingh, R. Barton y K. Briley, «End-to-End QoS Network Design: Quality of Service for Rich-Media & Cloud Networks,» de *End-to-End QoS Network Design: Quality of Service for Rich-Media & Cloud Networks*, C. Press, Ed., Cisco Press, 2013, pp. 1-1.

[46] J. M. J. Rodríguez, «Diseño de la investigación cualitativa: la entrevista en profundidad,» Alicante, 2008.

[47] A. Morales, Junio 2014. [En línea]. Available: http://cepra.cedia.org.ec/index.php/component/content/article/9-sin-categoria/127-implementacion-de-un-testbed-para-una-sdn-empleando-la-infraestructura-de-cedia.

[48] J. C. Chico, Implementación de un prototipo de una red Definda por Software (SDN) empleando una solución basada en Hardware, S. publicar, Ed., Quito: Revista Politécnica, 2013.

[49] Y. Chimeno, «SDN: ¿un cambio superficial?,» *en Route,* nº 23, pp. 2-3, Noviembre 2013.

[50] Y. Chimeno, «Soluciones TIC unificadas,» *enRoute,* nº 23, p. 4, Noviembre 2013.

[51] Logicalis, «Análisis independientes de tendencias tecnológicas para profesionales de TIC - SDN,» *Logicalis Now,* vol. I, nº 21, p. 7, Junio 2014.

[52] G. D. F. Blandón, «OPENFLOW… EL PROTOCOLO DEL FUTURO,» Noviembre 2013. [En línea]. Available: http://www.slideshare.net/daniblago/openflow-protocolo-del-futuro.

[53] O. A.C., Marzo 2015. [En línea]. Available: http://www.observatel.org/telecomunicaciones/Qu_significa_la_Convergencia.php.

[54] M. W. Marín, Marzo 2015. [En línea]. Available: http://www.ie.itcr.ac.cr.

[55] https://docs.fajardo.inter.edu, Marzo 2015. [En línea]. Available: https://docs.fajardo.inter.edu/Acad/atorres/CSIR2121/Shared%20Documents/El%20Modelo%20OSI.pdf.

[56] C. AUTELSI, «Manual Ciudades Digitales,» 2006.

[57] A. Huth y J. Cebula, «The Basics of Cloud Computing,» Carnegie Mellon , 2011.

[58] J. e. Keith y N.-L. Burkhard, «The Future of Cloud Computing,» Europe, 2010.

[59] UPAEP, «Visión General de Google,» 2015.

[60] W. S. Amazon, «Amazon EC2,» 2015.

[61] G. Lewis, «Basics About Cloud Computing,» Pittsburgh, 2010.

[62] P. y. A. Espiñeira, «Biometría:Generalidades y aplicaciones para los negocios,» *Boletín de Consultoría Gerencial,* pp. 3-12, 2013.

[63] P. N. López y A. J. J. Toro, «Técnicas de biometría basadas en patrones faciales del ser humano,» 2012.

[64] P. Rotter, «Las tecnologías de identificación personal: la biometría,» *Mirando al Futuro ,* pp. 72-75, 2008.

[65] G. Ferrer, «Sistema Biométrico de reconocimiento de huella dactilar en el control de Acceso de Entrada y Salida,» BOGOTA, 2013.

[66] M. Á. Ferrer, J. J. L. Zai, J. B. Hérnandez y C. M. Travieso, PALMAR, AUTENTICACIÓN DE PERSONAS A PARTIR DE LA BIOMETRÍA DE LA REGIÓN DÍGITO, 2008.

[67] V. A. Maya, «Sistema biométrico de Reconocimiento de huella Dactilar en Control de Acceso de entrada y salidas,» Bogota D.C., 2013.

[68] C. Date, Introducción a los Sistemas de base de Datos, México: Addison Wesley Iberoamericana, 1993.

[69] I. Ellislab, «Manual de Codeigniter,» Copyright, Estados Unidos, 2011.

[70] J. Lopez, C. P. Jesús y R. J. Obispo, «Desarrollo de Aplicaciones Multiplataforma y Web,» Alfaomega, México, 2013.

[71] D. M. Kroenke, Procesamiento de Base de Datos Funfamentos, diseño e implementación, México: Pearson Educación, 2003.

[72] J. K. Lyon, Introducción al diseño de Banco de Datos, México: Limusa, 1983.

Resumen Final

Desarrollo de un Prototipo de red definida por software SDN para la Gestión mediante Recursos de Estándar Abierto

Félix Mauricio Murillo Calderón

119 páginas

Proyecto dirigido por: Ing. José Marcelo Balseca Manzano, Mg.

Este documento presenta el desarrollo del prototipo de red definida por software SDN utilizando para la gestión recursos de estándar abierto, en el que se ha desarrollado aplicaciones con controladores OpenFlow que se adhieran a cualquier tipo de infraestructura de red, buscando generar una mayor eficiencia y optimización de la capacidad de la red en el servicio que se brinda a los usuarios; para lo cual se ha utilizado los modelos ágiles XP en la que su principal herramienta es la disponibilidad de cambios a medida que sea necesario. El aplicativo de red SDN debe monitorear el tráfico de información dependiendo de las políticas establecidas por el administrador de red; adicionalmente la aplicación del prototipo de red presentó comportamientos con niveles de satisfacción aceptables en las pruebas iniciales de enrutamiento, monitoreo y de la reducción del número de paquetes que debe generar el servidor de manera que exista un flujo de información sin interrupciones. Esto permite que la infraestructura de red utilizando el protocolo OpenFlow en base a Floodlight se convierta en una red directamente programable, ágil, escalable, de gestión centralizada y con programación configurada mediante la utilización de Mininet para la creación de redes virtuales reales, lo que permite a los administradores optimizar recursos y mantener la simplicidad en los diseños, convirtiéndolas así en redes flexibles, seguras y eficientes.

Palabras claves: SDN, OpenFlow, Mininet, Floodlight, red, eficiente, ágil, infraestructura, administración de red.