FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

DISEÑO DE LA ARQUITECTURA DE UNA RED SDN MEDIANTE EL PROTOCOLO OPENFLOW CON SIMULACION EN EL SOFTWARE MININET PARA LA INFRAESTRUCTURA DE UNA PYMES

Autor

Edison Daniel Guanoluisa Jaramillo

Año

2019

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

DISEÑO DE LA ARQUITECTURA DE UNA RED SDN MEDIANTE EL PROTOCOLO OPENFLOW CON SIMULACION EN EL SOFTWARE MININET PARA LA INFRAESTRUCTURA DE UNA PYMES

Trabajo de titulación presentado en conformidad con los requisitos establecidos para optar por el título de Ingeniero en Redes y Telecomunicaciones

Profesor Guía

MSc. Milton Neptalí Román Cañizares

Autor

Edison Daniel Guanoluisa Jaramillo

Año

2019

DECLARACION DEL PROFESOR GUIA

"Declaro haber dirigido el trabajo, Diseño de la arquitectura de una red SDN mediante el protocolo openflow con simulación en el software Mininet para la infraestructura de una pymes a través de reuniones periódicas con el estudiante Edison Daniel Guanoluisa Jaramillo, en el semestre 201910 orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación".

_____

Milton Neptalí Román Cañizares
Magister en Gerencia de Redes y Telecomunicaciones
C.I.050216344-7

DECLARACION DEL PROFESOR CORRECTOR

"Declaro haber revisado el trabajo, Diseño de la arquitectura de una red SDN mediante el protocolo OpenFlow con simulación en el software Mininet para la infraestructura de una pymes a través de reuniones periódicas con el estudiante Edison Daniel Guanoluisa Jaramillo, en el semestre 201910 dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación".

_____

Johanna Rafaela Ortega Briones
Master en Telecomunicaciones
C.I.17145789-4

DECLARACION DE AUDITORIA DEL ESTUDIANTE

“Declaro que este trabajo es original, de mi autoría, que se han citado todas las fuentes correspondientes y que en que su ejecución se respetaron todas las disposiciones legales que protegen los derechos de autor vigentes”

_____

Edison Daniel Guanoluisa Jaramillo

C.I.172125935-4

AGRADECIMIENTOS

Agradezco a Dios por haberme dado la paciencia, la perseverancia y la sabiduría para poder culminar esta etapa, tropecé muchas veces, pero nunca me rendí, a mis padres y mis hermanos que siempre estuvieron apoyándome cada vez que quería renunciar, a mi esposa y mi Nicolito que ellos fueron mi principal motivación para poder culminar mis estudios los amo con mi corazón, finalmente a toda la gente profesional y de gran corazón que me ha enseñado a trazarme objetivos y poder cumplirlos.

DEDICATORIA

Dedico este trabajo a mis padres que me apoyaron en todo este trayecto, a mi esposa Pao y mi hijo Nicolás que fueron mi principal motivación para no rendirme y poder alcanzar mi objetivo.

¡¡Esto es por ustedes!!

RESUMEN

El siguiente documento abarca fundamentos teóricos y describe el funcionamiento de la tecnología SDN (Redes Definidas por Software), mediante una simulación que trata de demostrar cómo se puede administrar y el alcance que tienen; además, demostrar el futuro de las conexiones de datos convergentes definidas por software. También presenta una propuesta de un diseño de red con la utilización de tecnologías SDN, para una PYMES, luego de realizar un análisis de su infraestructura de red y de identificar cada uno de los aspectos en los cuales se puede mejorar con la tecnología SDN.

ABSTRACT

The following document covers theoretical foundations and describes the functioning of the SDN (Software Defined Networks) technology, by means of a simulation that tries to demonstrate how it can be managed and the scope that they have; In addition, demonstrate the future of convergent data connections defined by software. It also presents a proposal for a network design with the use of SDN technologies, for the PYMES, after performing an analysis of its network infrastructure and identifying each of the aspects in which it can be improved with the SDN *technology*.

ÌNDICE

1. CAPITULO I. INTRODUCCIÓN 1
1.1. Alcance 1
1.2. Justificación 2
1.3. Objetivos 2
1.3.1. Objetivo General 2
1.3.2. Objetivos Específicos 2
2. CAPITULO II. MARCO TEÓRICO 3
2.1. SDN (Software Defined Networking) 3
2.1.1. Características de la Red SDN 4
2.1.2. Componentes de la Red SDN 4
2.1.3. Factibilidad Operativa 7
2.1.4. Ventajas de la Red SDN 7
2.1.5. Diferencias entre Redes Tradicionales vs Redes SDN 8
2.2. OpenFlow 9
2.2.1. Componentes del Protocolo OPENFLOW 10
2.2.2. Puertos OPENFLOW 12
2.2.3. Tablas OPENFLOW 12
2.2.3.1. Entrada de las tablas de Flujo 13
2.2.4. Controlador SDN 14
2.3. MININET 15
2.3.1 Componentes de la Red MININET 17
2.4. Virtualización de Redes 18
2.4.1. Características de una Red Virtualizada 19
2.4.2. NFV, Virtualización de Servicios de Red 20
2.4.3. Diferencias Redes SDN Y NFV (Network Functions Virtualization) . 21
2.4.4. Integración SDN y NFV 21
2.5. Seguridades en la Red SDN 22
2.5.1. Ataques Volumétricos, Como Inundaciones de SYN 23

2.5.2. ATAQUES DDOS 23
2.5.3. Software Defined Secure Networks (SDSN) 24
2.6. Data Center definido por Software (SDDC) 24
2.6.1. Beneficios SDDC 25
2.7. SD-WAN 25
2.7.1. SDN Híbrida 26
2.7.2. Funcionalidades de SD-WAN 26
3. CAPITULO III. INFRAESTRUCTURA ACTUAL DE LA PYMES. 28
3.1. Infraestructura De la Red 28
3.2. Estado actual de la Red 29
3.2.1. Ancho de banda 30
3.2.2. Seguridad 31
3.2.3. Uso de recursos de *hardware* 34
3.2.4. Tráfico de red 35
3.3. Aspectos que considerar 36
3.4. Aspectos favorables con SDN 37
4. CAPITULO IV. DISEÑOS DE LA RED SDN EN LA PYMES 39
4.1. Diseño del Controlador Floodlight 39
4.2. Diseño de la Red SDN en Mininet 40
4.2.1. Comandos utilizados para la creación de redes en Mininet 41
4.2.2. Componentes utilizados en Miniedit 43
5. CAPITULO IV. SIMULACIÓN DE LA RED SDN 44
5.1. Instalación Mininet 44
5.1.1. Diseñando la red SDN en Mininet 46
5.1.2. Instalación Controlador Foodligth 54
5.1.3. Escenario de emulación conectividad de la red 56
5.1.4. Escenario de simulación de Seguridades 64

5.1.5. Escenario de simulación SDN-WIFI 69
6. CAPITULO VI. EVALUACIÓN DE LA RED SDN 75
7. CAPITULO VIII. CONCLUSIONES Y RECOMENDACIONES 79
7.1. Conclusiones 79
7.2. Recomendaciones 80
REFERENCIAS 82
ANEXOS 85

ÌNDICE DE FIGURAS

Figura 1. Arquitectura de Red SDN 5
Figura 2. Arquitectura OpenFlow 9
Figura 3. Estructura Conmutador OpenFlow 11
Figura 4. Puertos OpenFlow 12
Figura 5. Tablas OpenFlow 13
Figura 6. Tabla de flujo OpenFlow 13
Figura 7. Pipeline Openflow 14
Figura 8. Arquitectura del diseño en Mininet 18
Figura 9. Arquitectura del diseño en Mininet 20
Figura 10. Tradicional WAN VS SD-WAN 27
Figura 11. Diagrama de Red de datos de la PYMES. 29
Figura 12. Ancho de banda interno 30
Figura 13. Tráfico web producido por los usuarios. 31
Figura 14. Protocolo de descubrimiento SSDP 32
Figura 15. Ataques de firewall 32
Figura 16. Informe de ataques de red mediante el origen. 33
Figura 17. Informe ataques de red basada en puertos de conexión. 34
Figura 18. Uso de disco memoria y CPU 35
Figura 19. Tráfico de red de acuerdo con los paquetes de envío y recepción. 36
Figura 20. Red SDN CON PORTOCOLO OPEN FLOW 38
Figura 21. Diseño del controlador Floodlight. 40
Figura 22. Red por defecto Mininet 46
Figura 23. Topología de red SDN en Mininet para simulación 47
Figura 24. Configuración del controlador 49
Figura 25. Programación Python 52
Figura 26. Programación Python 53
Figura 27. Panel de inicio controlador Floodlight. 55
Figura 28. Imagen del Log de inicio del controlador 55
Figura 29. Panel principal Floodlight 56
Figura 30. Conectividad red interna Quito 57

Figura 31. Conectividad red interna Guayaquil 58
Figura 32. Conectividad de servidores 58
Figura 33. Conectividad de servidores 59
Figura 34. Switch Virtual Quito. 60
Figura 35. Host y Vlans creadas. 60
Figura 36. Detalle del Direccionamiento IP y evidencia de asignación de Vlans. 62
Figura 37. Vlans Ventas. 63
Figura 38. Hosts PC3 y PC4 añadidos a Vlans Ventas 63
Figura 39. Diagrama de control de acceso. 64
Figura 40. Listas de control de acceso configuradas. 65
Figura 41. Evidencia Bloqueo PC2 a SRVBK 66
Figura 42. Evidencia acceso sin bloqueo PC1 a SRVBK 66
Figura 43. Resultado Wireshark tráfico bloqueado 67
Figura 44. Resultado Wireshark tráfico bloqueado 67
Figura 45. Resultado Wireshark tráfico bloqueado 67
Figura 46. API Firewall Floodlight. 68
Figura 47. Tráfico de Ping desde Quito a la red de Guayaquil 69
Figura 48. Wireshark tráfico permitido. 69
Figura 49. Listado de regla de firewall. 69
Figura 50. Diseño de la red SDN-WIFI 70
Figura 51. Configuración Miniedit wifi controlador WIFI 71
Figura 52. Asociación de los host a los Acces Point. 71
Figura 53. Red Wifi Python 72
Figura 54. Integración de los hosts al controlador WIFI 72
Figura 55. Conectividad de host a controlador WIFI. 73
Figura 56. Tiempo de trasmisión y recepción de host conectado a WIFI. 73
Figura 57. Estadística de conexiones WIFI. 74
Figura 58. Cobertura que tienen los AP y como es la movilidad. 74
Figura 59. Controlador Floodlight 75
Figura 60. Topología SDN FLODLIGTH 76
Figura 61. Políticas de Firewall. 76

Figura 62. Access Control 77
Figura 63. Detalle OpenSwitch S1. 77
Figura 64. Detalle Openswitch S2 78
Figura 65. Detalle OpenSwitch S3 78

ÌNDICE DE TABLAS

Tabla 1 Comparación entre la Red tradicional Vs Red SDN. 8
Tabla 2 Diferencias entre SDN y NFV. 21
Tabla 3 Componentes de la PYMES. 28
Tabla 4 Componentes de la red simulada SDN. 41
Tabla 5 Comandos Para redes en Mininet. 42
Tabla 6 Componentes MINIEDIT 43
Tabla 7 Direccionamiento IP Sucursal Quito. 48
Tabla 8 Direccionamiento IP sucursal GYE..... 48
Tabla 9 Configuraciones de los Switches virtuales 50
Tabla 10 Configuraciones hosts Red SDN Quito 51
Tabla 11 Configuración Red SDN Guayaquil 51
Tabla 12 Requerimiento de Hardware para el controlador Floodlight 54
Tabla 13 Vlans para configurar. 59

1. CAPITULO I. INTRODUCCIÓN

En la actualidad el modelo de infraestructura tradicional que mantienen las empresas genera ciertas limitaciones en su topología de redes de datos tales como la complejidad al identificar el correcto tráfico de protocolos que intervienen en la red, los administradores de red tienden a perder tiempo configurando varios mecanismos y equipos para lograr una política que abarque las necesidades solicitadas por la empresa. Dicho proceso genera que la escalabilidad sea más compleja.

La SDN es un nuevo modelo acerca de cómo los administradores de TI (Tecnologías de la información) plantean la gestión de redes. La introducción de aplicaciones de rendimiento intensivo, Big Data y otras tendencias hacen que la capacidad de la red tradicional se convierta en un cuello de botella. Frente a ello, SDN permite a las empresas que la administración e implementación de componentes de redes sea mucho más sencilla y además permite controlar el despliegue de recursos de infraestructura subyacente, ya que la red definida por software utiliza políticas que administran y asignan los recursos de red.

La PYMES considera de gran importancia elaborar una nueva arquitectura de red que pueda ser aplicable en su entorno tecnológico.

1.1. Alcance

Se realizará una descripción de la arquitectura de red SDN, sus protocolos a utilizar y una simulación bajo el software Mininet (Herramienta de simulación) y controlador SDN aplicando *kernel* real y código de aplicación para la virtualización de los componentes de red como firewall, routers, etc.

La simulación de SDN y la evaluación de cada uno de los controladores permitirán valorar las capacidades y funcionalidades de la nueva arquitectura de red. Se realizará un análisis de resultados del diseño planteado.

Se va a proponer reproducir todo el modelo de red en software que permitirá crear y aprovisionar cualquier topología de red desde una red simple hasta una red compleja de múltiples niveles. Se podrá habilitar una biblioteca de elementos y servicios de red lógicos, como Switches lógicos, enrutadores y firewall balanceadores de carga y lo más importante la seguridad para la carga de trabajo.

1.2. Justificación

Las redes de datos han experimentado un crecimiento exponencial tanto en tamaño como en su complejidad llevando a las arquitecturas de red tradicionales a su límite, por lo cual ha generado problemas en los aspectos de control y orquestación en los recursos tecnológicos.

Una de las alternativas tecnológicas que ha surgido en los últimos tiempos es la arquitectura Software Defined Networking (SDN).

La convergencia de los dispositivos móviles, el desarrollo en los contenidos audiovisuales, la virtualización de servidores y la extensión a plataformas Cloud son elementos que generan la necesidad de impulsar un cambio en las arquitecturas de red tradicionales.

1.3. Objetivos

1.3.1. Objetivo General

Diseñar una arquitectura de red SDN (Software Defined Networking) bajo el protocolo Openflow, para mejorar la estructura de red actual y la seguridad informática interna de la PYMES.

1.3.2. Objetivos Específicos

- Analizar la infraestructura de red actual que mantiene la empresa.

- Diseñar una topología de red mediante la arquitectura SDN considerando el estándar Openflow
- Simular con el software Mininet diferentes escenarios de operatividad, funcionalidad y usos útiles de acuerdo con los avances tecnológicos.
- Evaluar la funcionalidad y el comportamiento del nuevo diseño de red SDN para solventar las necesidades tecnológicas de la pyme.

2. CAPITULO II. MARCO TEÓRICO

Este capítulo recopila fundamentos teóricos sobre la red SDN, sus características, estándares y normas junto con ventajas sobre redes tradicionales.

2.1. SDN (Software Defined Networking)

En la actualidad la red SDN es una alternativa para reducir las dificultades en las empresas, permitiendo que estas puedan controlar el despliegue de recursos de infraestructura.

> (…)El término SDN (Software Defined Network o red definida por software) se ha venido acuñando en los últimos dos años para hacer referencia a una arquitectura de red que permite separar el plano del control, del plano de datos, para conseguir redes más programables, automatizables y flexibles. Con SDN se virtualiza la red independizándola de la infraestructura física subyacente (Norberto Figuerola, 2014)

Las redes SDN hoy en día son una solución a problemas que surgen por la demanda de recursos. Además, tienen un nuevo enfoque para programar de una manera dinámica mediante software. Siendo una alternativa aplicable en el entorno tecnológico de las empresas.

2.1.1. Características de la Red SDN

Las características principales de la arquitectura SDN según el blog de Sicrom Technology (Sicrom Technology, s.f.), son:

- **Es directamente programable:** Se puede tener el control de la red mediante programación, ya que está desacoplada de las funciones de redirección.
- **Gestión de forma centralizada:** La inteligencia de la red está centralizada en un software basado en controladores SDN que mantienen una visión global de la red, que aparece en aplicaciones y motores de políticas como un único Switch lógico.
- **Es Ágil:** Abstrayendo el control de desvío permite a los administradores ajustar dinámicamente el flujo de tráfico de toda la red para satisfacer las necesidades cambiantes.
- **Basados en estándares abiertos y de proveedor neutral:** Cuando se implementa a través de estándares abiertos, SDN simplifica el diseño de la red y la operación porque las instrucciones son proporcionados por los controladores SDN en lugar de múltiples dispositivos y protocolos específicos de los fabricantes.
- **Programación configurada**: SDN permite a los administradores de red configurar, administrar, asegurar y optimizar los recursos de red muy rápidamente a través de los programas de SDN dinámicos, automatizados, que pueden escribir ellos mismos ya que los programas no dependen de software propietario.

2.1.2. Componentes de la Red SDN

La arquitectura SDN está compuesta por tres capas funcionales: capa de infraestructura, capa de control y capa de aplicación. La red SDN mantiene su lógica de control de la red La red SDN trabaja mediante interfaces abiertas. Como se puede observar en la Figura 1.

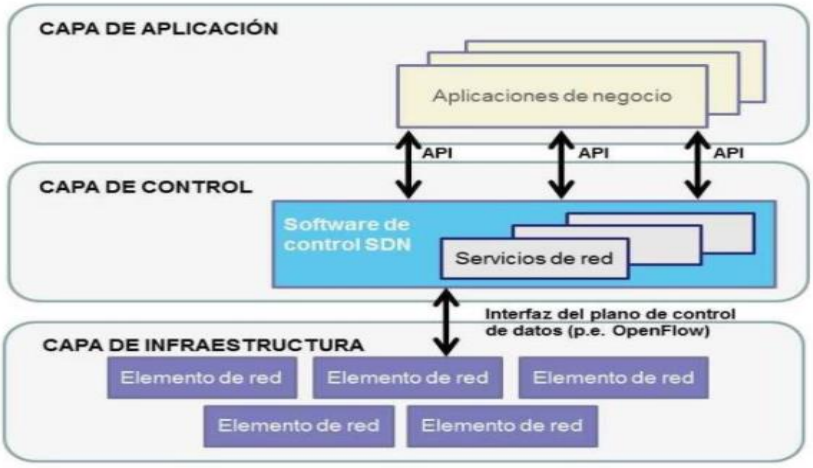


Figura 1. Arquitectura de Red SDN
Tomada de (Tejedor, 2014)

2.1.2.1. Interfaces

La capa controladora tiene interfaces abiertas, la primera interactúa con la capa de datos denominada *Southbound* “hacia el sur”, y otra que trabaja con la capa de gestión denominada *Northbound* “hacia el norte”. Es así que para comunicarse con los demás controladores en la red se tiene las interfaces *Eastbound* “hacia el este” y *Westbound* “hacia el oeste”.

- **Interfaz Southbound:** Este conjunto de instrucciones son las API expuestas a capas inferiores que permiten la externalización del plano de control de la SDN al plano de datos.
- **Interfaz Northbound:** Permite el intercambio de datos entre el controlador y la aplicación. El tipo de información que se intercambia, su forma y su frecuencia depende de cada aplicación. No hay estandarización para esta interfaz.
- **Interfaces *Eastbound* y *Westbound*:** Permite la interconexión de redes convencionales con las SDN, además sirven como conducto de información entre varios planos de control de la SDN de diferentes dominios de SDN. Ayudan a lograr una visión de red global e influyen en las decisiones de enrutamiento de cada controlador. Adaptado de (Akram Hakiri, 2014)

2.1.2.2. Capa de Infraestructura

La capa de infraestructura o capa de datos está compuesta por nodos de red cuyo objetivo es la conmutación y el encaminamiento de paquetes. Su principal función es proporcionar el servicio fundamental de reenvío de datos.
Esta capa hace referencia a los dispositivos de red como por ejemplo enrutador (Routers), conmutador (Switch) o contrafuegos (Firewall) estos pueden ser físicos o virtuales.

2.1.2.3. Capa de control

Esta capa es la más importante dentro de la arquitectura de las SDN esta será la encargada de centralizar las aplicaciones y las capacidades de la red también permite visualizar la estructura de la red y configurar cada uno de los nodos de red estableciendo distintas rutas para el envío y recepción de tráfico el mismo que tomará todas las decisiones de la red.

El controlador viene hacer un elemento importante en el diseño de Red SDN, debido a que es una capa necesaria para las decisiones de la red y cada una de las configuraciones generadas por el administrador.

2.1.2.4. Capa de Aplicación

La capa de aplicación o también conocida como capa de gestión será la de más alto nivel y desde la cual controlará la actividad de la red, por lo cual se mantiene gestión de diferentes API del controlador, cuya función es la orquestación o aplicaciones para la infraestructura corporativa.

Según (Bermudes, 2018), la funcionalidad de estas APIs es comunicar cada una de las acciones a la capa de control, donde se define que uso se le dará a la red, se toman decisiones oportunas y se comunica a la capa de infraestructura mediante el protocolo openflow para que esta pueda establecer la acción generada.

Se puede entender que el plano de aplicación es el grupo de APIS cuya función es tomar cada una de las operaciones otorgadas por la interfaz *Northbound* para insertar la administración de Red y el funcionamiento.

2.1.3. Factibilidad Operativa

La funcionalidad de las redes SDN en el futuro serán menos complejas que redes tradicionales, dado que toda su tecnología estará centralizada en el controlador, es así que SDN proyecta una nueva arquitectura de red que sea más eficiente que una red convencional.

2.1.4. Ventajas de la Red SDN

La tecnología SDN tiene varios beneficios para las empresas que buscan nuevas alternativas para ejecutar la administración de la red. Las ventajas según el blog puntinformátic (Punt Informátic, 2018), son:

Las redes SDN son más flexibles y ahorran costes: Cuando una empresa virtualiza su infraestructura de red, el personal IT puede adaptar la misma a las diferentes necesidades de la red sin necesidad de comprar nuevos nodos. Simplemente, bastará con reprogramar el software para hacer que la red se ajuste a los cambios.

Gestión centralizada y simple: SDN permite equilibrar la carga y distribuir el tráfico de manera eficiente para evitar los puntos de bloqueo, lo que contribuye a una mejora del rendimiento. Además, este tipo de redes se instalan y configuran de manera automática, reduciendo los costes operativos y simplificando las funciones que ya han sido instaladas.

La automatización: La automatización contribuye a crear un entorno más predecible y consistente y promueve la escalabilidad para acomodar picos y valles en la carga de tráfico, en función de las necesidades surgidas en cada momento.

Aumentan la seguridad: Con la automatización de los procesos, una red SDN es más robusta y mejora la seguridad, ya que, gracias al despliegue de una infraestructura de red de este tipo, los profesionales IT de una compañía pueden resolver fácilmente cualquier incidencia desde un plano de control centralizado en menos tiempo que en las redes convencionales.

Una infraestructura SDN se aprovecha del Cloud Computing: La computación en la nube es una de las grandes revoluciones en la industria de la informática y de Internet. A medida que las empresas se trasladan a la nube, su infraestructura debe ser virtualizada y administrada centralmente para que los profesionales de IT creen nuevos servicios en la parte superior de la red.

2.1.5. Diferencias entre Redes Tradicionales vs Redes SDN

Entre la red tradicional y la redes SDN se pueden encontrar diferencias que surgieron debido a los cambios en la tecnología de las redes, siendo hoy en día la red SDN una tendencia. A continuación, se realiza un cuadro comparativo.

Tabla 1
Comparación entre la Red tradicional Vs Red SDN.

RED TRADICIONAL	RED SDN
La Red es estática y tiene su complicación al aplicar políticas.	Dinámica.
Incapaz de ser Escalable	Es capaz de ser Escalable
Configuración descentralizada.	Configuración centralizada.
Interfaces Propietarias.	APIs bien definidas y abiertas.
Envió de paquetes basados en coincidencias de la tabla.	Formato y acciones de las tablas claramente especificadas.
Nuevas aplicaciones en mayor tiempo.	Nuevas aplicaciones en menos tiempo.
Altos costos de recursos operativos y financieros.	Reducción de costos operativos y financieros.

2.2. OpenFlow

Openflow es el primer protocolo para el diseño de las SDN, cuyo objetivo es que la red sea programable mediante una plataforma centralizada. Openflow separa los tráficos y escoger la mejor ruta adicional podrá mejorar la forma en que los paquetes son procesados. Su función principal es el desarrollo de nuevas maneras de direccionamiento de tráfico, seguridad, control de datos, arquitectura de direccionamiento, etc. De acuerdo con (Spera, 2013) "*OpenFlow es uno de los componentes claves para posibilitar "software-define-network" y uno de los pocos protocolos estandarizados de SDN que permite manipulación directa a la capa de forwarding de los dispositivos de red*"

La arquitectura de OpenFlow tiene como núcleo un Switch Ethernet el mismo que tiene una trama de datos de forma interna y una interface que añade o borra entradas de flujo.

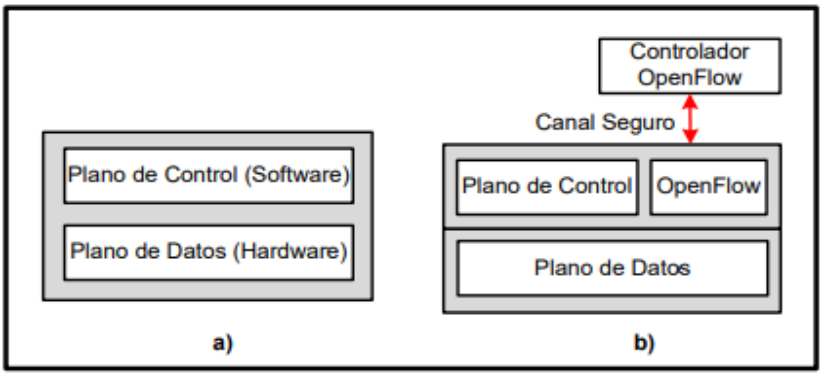


Figura 2. Arquitectura OpenFlow
Tomado de (López, 2013)

El Switch tradicional incluye en su estructura toda la gestión de los datos que viajan a través de él, así como el control de la red.

La función del plano de control es el enrutamiento de rutas, topología, estado y el comportamiento con redes adyacentes, adapta configuraciones en el sistema de administración y de gestión e intercambio de paquetes de enrutamiento este plano señala a la capa de control la dirección de cada uno de los paquetes entrantes y salientes.

La función del plano de datos es procesar los datos en función a su trama de enrutamiento.

En Switch Openflow, la inteligencia de la red se encuentra en una controladora externa. Un Switch Openflow permite que la comunicación lógica entre un Switch y el controlador SDN mediante un canal seguro.

Cualquier equipo físico o virtual que pueda trabajar con el protocolo openflow podrá ser administrado por la capa de control sin importar las marcas fabricantes.

Las ventajas de Openflow son:

- Administración fácil y eficaz de la red
- Escalabilidad más funcional en la red
- División del flujo de datos y del control
- Avance tecnológico
- Aplicaciones programables

2.2.1. Componentes del Protocolo OPENFLOW

De acuerdo con (Muro, 2016): "*La mayoría de los conmutadores Ethernet contienen tablas de flujos, el paradigma de OpenFlow es la posibilidad de modificar estas tablas de forma dinámica con el objetivo de implementar cortafuegos, NAT, QoS y recolectar estadísticas. Es posible experimentar con nuevos protocolos, nuevos modelos de seguridad, esquemas de direccionamiento, incluso alternativas para IP lo que supone una mayor innovación*".

De acuerdo a lo expuesto por el autor podríamos entender que uno de los principales objetivos de Openflow es que los usuarios puedan definir los flujos para tomar los caminos a seguir en la red sin generar problemas en el tráfico frecuente.

Según (Muro, 2016), el dispositivo OpenFlow dispone de, al menos, tres partes que son las siguientes:

- **Tabla de flujos:** Con una acción asociada a cada entrada de la tabla, indicando al switch como debe procesar ese flujo.
- **Canal seguro** que conecte el switch a un proceso de control remoto (controlador), que permita el envío de comandos y paquetes entre el conmutador y el controlador usando el protocolo OpenFlow.
- **Controlador:** Un controlador añade y elimina entradas en la tabla de flujos.

Tal como se indica en la Figura 3.

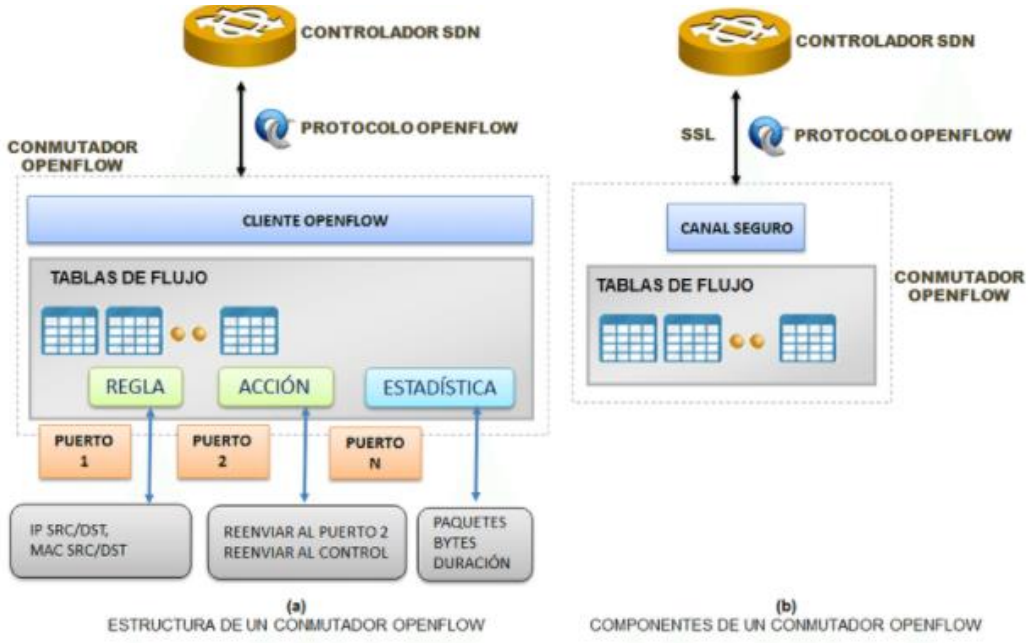


Figura 3. Estructura Conmutador OpenFlow
Tomado de (Muro, 2016)

Openflow utiliza el identificador denominado Flows para determinar las reglas configuradas en el controlador las mismas que pueden ser de forma dinámica o estática.

Las acciones que se pueden ejecutar son:

- Convertir tramas en flujo privado
- Eliminar Flujos
- Enrutar Flujos
- Permitir flujos

2.2.2. Puertos OPENFLOW

A continuación, en la Figura 4 se puede observar la tabla de puertos que utiliza el protocolo en estudio el mismo que se divide en Puertos físicos, lógicos y reservados y sus especificaciones en cada uno.

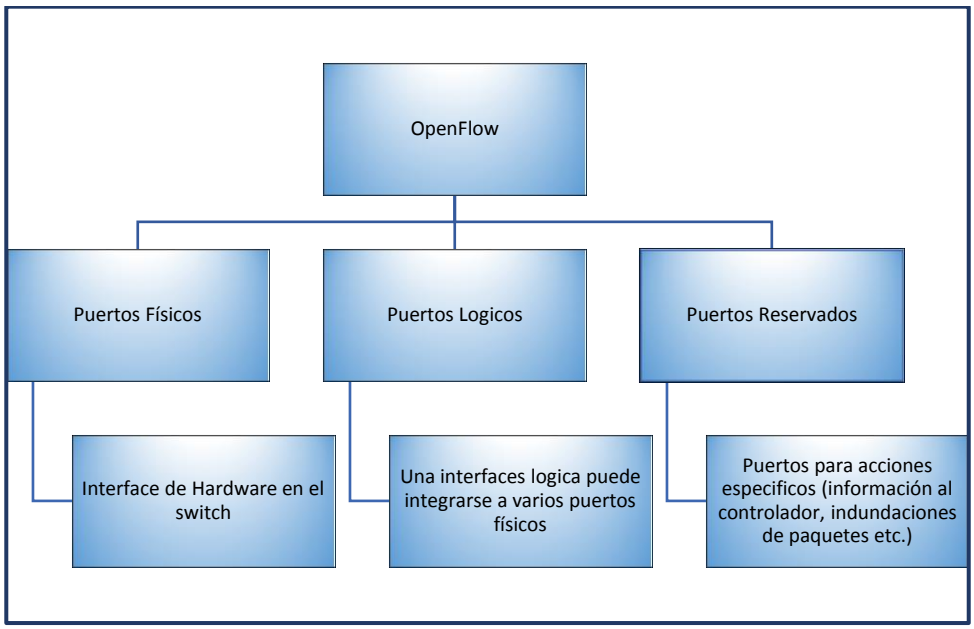


Figura 4. Puertos OpenFlow

2.2.3. Tablas OPENFLOW

De acuerdo con la figura 5 explica que existen tablas de flujo de grupo y tablas de acciones, las cuales se encargan de buscar coincidencias entre los paquetes que circulan por la red con el objetivo que no exista ningún conflicto de red o suplantación de rutas o de IPs, cada una de estas tablas ejecutan diferentes acciones como integrar flujos de datos entrantes y salientes, reenviar tráfico a un Gateway y acciones como calidad de servicio.

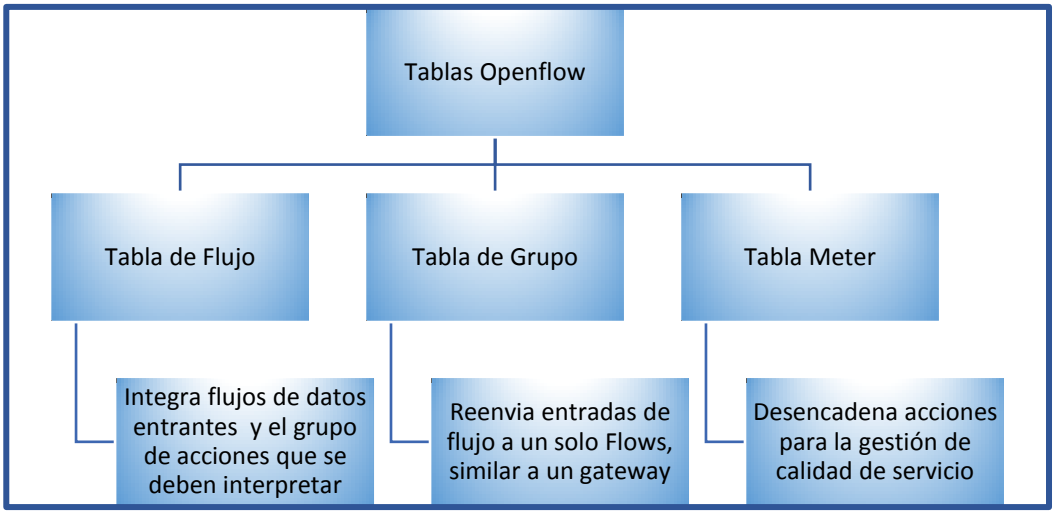


Figura 5. Tablas OpenFlow

Según (Jain, 2014), las tablas del protocolo OpenFlow en su versión 1.3 tiene ciertos componentes en los flujos entrantes si se compara con la versión 1.0, esta versión se compone de campos *Priority*, *Timeouts* y *Cookie*.

En la Figura 6 se observan los componentes de los flujos entrantes y posteriormente se explicará el funcionamiento de cada uno.

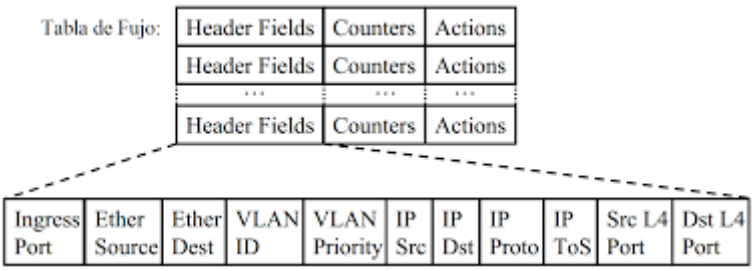


Figura 6. Tabla de flujo OpenFlow
Tomado de (Jain, 2014)

2.2.3.1. Entrada de las tablas de Flujo

La tabla de flujo de OpenFlow cuenta con componentes de los flujos entrantes que son los siguientes:

- **Match Field:** Es la cabecera del paquete de entrada especificada en una tabla previa.
- **Priority:** El flujo de entrada realiza la operación de comparación con una tabla del Switch, este proceso se presenta principalmente cuando hay nuevos flujos de datos, el Switch envía una nueva gestión del controlador.
- **Counters:** Esta cabecera se utiliza estadísticas de cada uno de los flujos de entrada.
- **Instructions:** Cada flujo tienen un grupo de instrucciones las cuales al ejecutarse hace comparaciones con los paquetes así sabrá qué proceso se realiza. Adaptado de (Pardo, 2014)

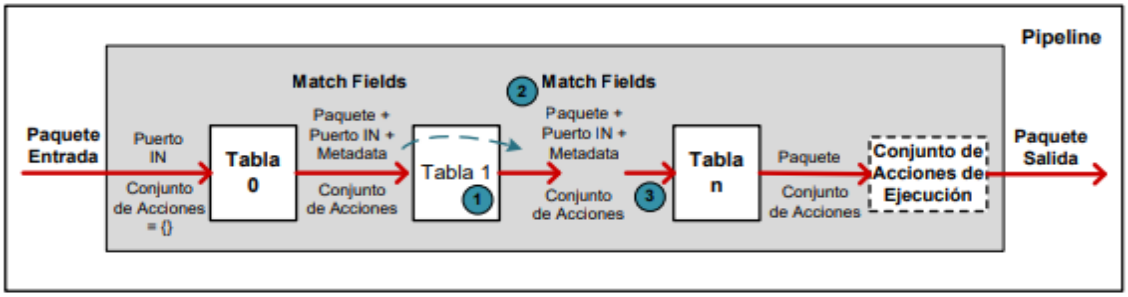


Figura 7. Pipeline Openflow
Tomado de (Pardo, 2014)

2.2.4. Controlador SDN

La capa de control es el núcleo y el componente más importante de la red con el fin de dirigir toda la red.

También sus funciones son centralizar las comunicaciones que se pasa a través de los dispositivos de red, así mismo mantiene una visualización total de la red y verificará el comportamiento de los flujos.

En la actualidad existen varias soluciones de controladores, entre los cuales se puede mencionar POX, Beacon, Floodlight, SDN VAN HP, cada uno de estas algunas ventajas la principal diferencia es el lenguaje de programación con el cual fue desarrollado.

2.2.4.1. Controlador Floodlight

El controlador *Floodlight* permite el control de la red SDN, en el cual se puede agregar APIs de código abierto para establecer diferentes funciones en cuanto a la gestión de la Red.

> El Floodlight Open SDN *Controller* es un *OpenFlow Controller* de clase empresarial, con licencia Apache y basada en Java. Cuenta con el apoyo de una comunidad de desarrolladores, incluidos varios ingenieros de Big Switch Networks. (..)*Floodlight* está diseñado para funcionar con el creciente número de conmutadores, enrutadores, conmutadores virtuales y puntos de acceso que admiten el estándar OpenFlow. (Floodlight, 2018)

Floodlight contiene una aplicación de firewall que aplica ACL de manera reactiva. El controlador impone reglas de ACL al monitorear los mensajes de entrada de paquetes y luego empujar las entradas de flujo relevantes. Sin embargo, de forma proactiva, el controlador impone reglas de ACL sin que el conmutador lo solicite, para evitar demoras adicionales.

> Esta aplicación de ACL analiza la solicitud REST del usuario para la actualización de la ACL y la aplica mediante entradas de flujo estático de forma proactiva sin monitorear los mensajes de entrada de paquetes. También puede eliminar la entrada de flujo de ACL que generó oportunamente cuando se elimina la regla de ACL relacionada. (Floodlight, 2017)

2.3. MININET

Es un software basado en Linux que mantiene un conjunto de host, de switches y de controladores, conmutadores, enlaces, es utilizado para poder realizar pruebas de conceptos de SDN y OpenFlow, esto trabaja mediante una interfaz de comandos (CLI) y también de forma gráfica.

Mininet mantiene una licencia BSD de código permitido la misma que está abierta para fallos.

Se detalla las características por las cuales Mininet es el emulador más utilizado:

- **Fácil Uso:** el comando sudo **mn** puede desplegar una red completa de forma rápida.
- **Editable:** Se puede establecer cualquier tipo de topología y se puede experimentar nuevos ejemplos.
- **Ambiente de simulación real:** se puede emular cualquier tipo de envío de paquetes por medio de las interfaces reales Ethernet configurando anchos de banda o QoS, además tiene herramientas de diagnóstico que permite ver el estado de la comunicación en esta red emulada.
- **Escalable:** Se pueden permitir escalar redes con miles de *switches* sobre una única máquina física.
- **Realista:** la simulación es muy parecida a la implementación real.

La característica principal de Mininet son sus Switches que van a soportar el protocolo OpenFlow por lo tanto se lo usa para las pruebas de concepto de SDN.

Su Core interno es en plataforma Linux y siendo soportado en entornos virtuales permitiendo la creación de nodos, estos pueden ser maquinas o controladores o Switches, al ser una arquitectura basada en Linux permitirá el funcionamiento de tablas ARP y de enrutamiento. El código de Mininet es desarrollado por el lenguaje Python y en su minoría con C++.

La versión actual en la que trabaja MININET es 2.0 lanzada el 22 de noviembre del 2012.

Se considera los siguientes aspectos para poder inclinarse para poder usar Mininet.

- Su arranque es más rápido, en segundos estar con la red operativa
- Se puede crear sin límites de host y switches
- Ofrece mayor ancho de banda como básico 2Gbps

2.3.1 Componentes de la Red MININET

- **Host Aislados**

  Un grupo de procesos del nivel de usuario mantendrá un espacio de nombres de red que proporciona propiedad exclusiva de interfaces, puertos y tablas de enrutamiento.
- **Enlaces simulados**

  Linux Trafic Control (TC) permitirá imponer la velocidad de datos de cada enlace para dar forma al tráfico a una velocidad configurada. Cada host emulado tiene su propia interfaz Ethernet Virtual.
- **Interruptores simulados**

  El Linux Bridge predeterminado o el Open vSwitch que se ejecuta en modo Kernel se utilizan para cambiar paquetes a través de interfaces. Los conmutadores y enrutadores pueden ejecutarse en el Kernel o en el espacio de usuario.

  En la figura 8, se puede observar cómo se integra la red Mininet a una red emulada por un software o también a un hardware físico. Según la imagen se puede apreciar que en los 2 casos el controlador de red tiene la misma funcionalidad así como también las aplicaciones SDN, todo dependerá de cómo queremos armar nuestra infraestructura SDN.

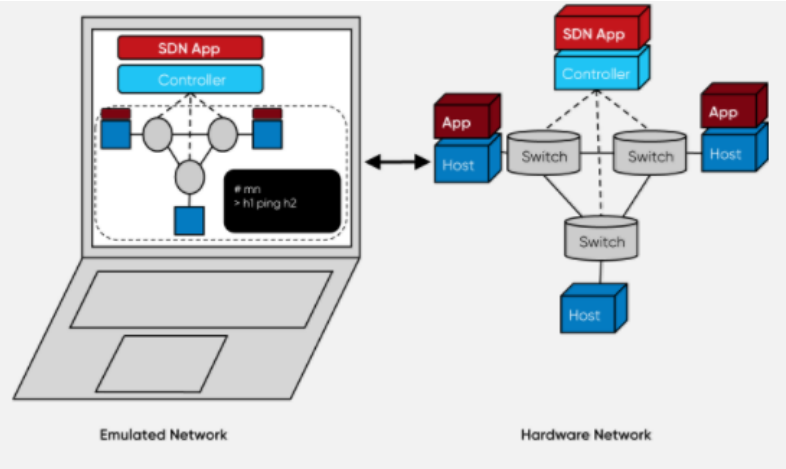


Figura 8. Arquitectura del diseño en Mininet
Tomado de (opennetworking, s.f.)

2.3.1.1. Limitaciones

Una de las limitaciones más significativas es la pérdida de rendimiento cuando se recurre a grandes cargas, ya que todos los Switches trasmitirán a la misma velocidad. Además, se ejecuta sobre una sola plataforma que emula enlaces cableados. No puede manejar diferentes kernels del sistema operativo simultáneamente.

2.4. Virtualización de Redes

La virtualización de la red es una funcionalidad que sirve para convertir pequeñas instancias virtuales de redes sobre máquinas físicas mediante la división de canales independientes.

Cada una de las capas puede ser administrada, y ser colocada en un servidor puntual. Con estos recursos lógicos se puede configurar túneles de red o segmentos distintos, cada uno con diferentes políticas de control.

Estar conectados es fundamental al momento de desplegar infraestructuras TI, de la funcionalidad que esto implica depende que haya una orquestación óptima de cada uno de los recursos que forman parte del centro de datos. Por

esto es muy importante la actualización continua, en estos tiempos es necesario una actualización recurrente ya que los data center han dejado de ser estructuras físicas cerradas.

En la actualidad, las empresas tienen la necesidad de desarrollar el concepto de virtualización de red como instrumento para poder ahorrar costos, acortar el ciclo de desarrollo de servicios y simplificar el despliegue de nuevas funcionalidades de redes para hacer la administración más sencilla de personal de TI.

Existen escenarios actuales que son Cloud Computing que requieren una postura al día de la red, es así que surge la necesidad de poder abordar el tema de la Virtualización de redes.

> La virtualización de redes permite adecuar las necesidades de red a esta nueva situación. La virtualización de las funciones de red consiste en centralizar en un software todos los servicios que se ejecutan tradicionalmente en el hardware de un CPD, es el caso del enrutamiento o la optimización WAN y Firewall. (GROUP, 2017)

2.4.1. Características de una Red Virtualizada

Algunas características se detallan a continuación:

- Los componentes de red pueden comunicarse unos con otros sea cual sea su marca de fabricante.
- Este tipo de virtualización siempre vienen acompañados de hipervisores para realizar la gestión de configuraciones.
- Existen funcionalidades para un mejor control de las redes virtualizadas como el mantener un cortafuegos o balanceadores, QoS estas se pueden definir por software, y gestionar su ubicación en cualquier parte de la red.

En la Figura 9, se observa una representación del desacoplamiento entre la topología física y la lógica a nivel de toda la red de una organización.

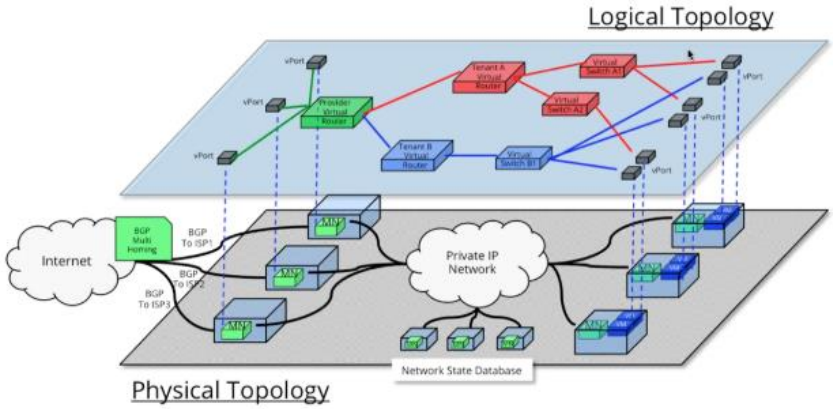


Figura 9. Arquitectura del diseño en Mininet
Tomada de (OpenWebinars.net, 2015)

2.4.2. NFV, Virtualización de Servicios de Red

La NFV forma parte de estos avances tecnológicos, pues permite implementar una infraestructura de la red con capacidades de automatización y programación más dinámicas para cumplir con las necesidades de las empresas.

> La virtualización de las funciones de red (NFV) es un enfoque de red en evolución que permite la sustitución de dispositivos de hardware dedicados y costosos tales como routers, firewalls y equilibradores de carga con dispositivos de red basados en software que se ejecutan como máquinas virtuales en servidores estándares de la industria. (Ciena Experience. Outcomes, 2018)

Esta virtualización permite que el mantenimiento de la red sea de menor costo. Además, cuenta con actualizaciones sencillas que generan poco espacio en el *Hardware*.

2.4.3. Diferencias Redes SDN Y NFV (Network Functions Virtualization)

El objetivo de SDN es agilitar el proceso de creación, control y automatización de las redes y responder rápidamente a las necesidades de los negocios, así como también la tecnología NFV que ayuda a virtualizar elementos de red crear todo un conjunto de máquinas virtuales.

La siguiente tabla muestra las diferencias entre SDN Y NFV.

Tabla 2
Diferencias entre SDN y NFV.

Software Define Network (SDN)	**Network Function Virtualización (NFV)**
Plano de datos y control Independientes Incapaz de ser Escalable	Funciones de red de appliance dedicado a servidores genéricos
Utilización en Data Center/Cloud	Utilización en Proveedores de servicios.
Orquestación Cloud y Networking	*Routes, Firewall, Gateways*, CDN,WAN, Aceleradores, ADC
Openflow	Ninguno.
Open Networking Foundation (ONF).	ETS NFV Working Group.

2.4.4. Integración SDN y NFV

SDN Y NFV pueden trabajar de forma conjunta, Por ejemplo, un prestador de servicios de internet puede trabajar en la virtualización de un Router para poder configurar las diferentes rutas que necesitan comunicarse.

La red definida por software se encargará de separar el módulo de control y los datos. Estos datos podrán empaquetarse y serán optimizados, mientras que la capa de control ejecutará la máquina virtual.

El resultado es una solución en la que no hace falta tener un dispositivo costoso. Se puede reemplazar por un Software específico y Hardware genérico. También se optimizan costos en los centros de datos, al no depender de una plataforma dedicada.

2.5. Seguridades en la Red SDN

La seguridad que preocupa a la red tradicional también preocupa a la red SDN. Al tener una programación en tiempo real y controles simplificados por medio del controlador SDN, esto también introduce nuevos desafíos de seguridad. La red SDN está expuesta a varios riesgos de seguridad, que vienen heredados según la perspectiva del diseño y arquitectura.

Uno de los factores de riesgo que más comprometen a las redes SDN es un ataque que pueda comprometer al controlador SDN a nivel del plano de control. Esto se debe al diseño centralizado según la arquitectura SDN ya que el controlador se transforma en el cerebro de la arquitectura SDN. Si el atacante logra ingresar a la capa del control y consigue ganar los permisos, el controlador SDN puede ser utilizado para dirigir los dispositivos de red que controla (Por ejemplo, los Switches) para poder descartar o autorizar tráfico inesperado entrante y saliente, o a si vez generar un ataque en contra de otros objetivos como dispositivos finales para enviar tráfico malicioso a fin de controlar sus recursos.

Una de las mayores características de SDN es que es capaz de ofrecer un nuevo nivel de visibilidad de red, esto permitirá anticiparse a una serie de ataques que podrían comprometer la red de la empresa

2.5.1. Ataques Volumétricos, Como Inundaciones de SYN

Este tipo de ataque consiste en grandes cantidades de paquetes TCP solamente con las banderas SYN configuradas. Las consecuencias de este tipo de ataque es un bloqueo del ancho de banda. A la transmisión de paquetes SYN por un cliente hostil para el propósito de encontrar puertos abiertos y *hackear* uno o más de ellos, se le llama análisis SYN. Un cliente hostil siempre sabe si un puerto está abierto cuando el servidor responde con un paquete SYN/ACK.

Cuando se establece un ataque, el servidor analiza el equivalente de varios intentos de conexión. El servidor puede responder a cada petición con un paquete SYN / ASK (sincronización reconocida) partiendo el origen de cada puerto abierto con un paquete de RST (RESET) desde cada puerto cerrado.
Para bloquearlos están los Interruptores programados para SDN, que actúan en primera línea para poder identificar las tramas de conexión, estos interruptores son capaces de dejar ir el tráfico o reoriéntalo a partir de diferentes técnicas y protocolos.

2.5.2. ATAQUES DDOS

Los ataques DoS se propagan a través de inundaciones de múltiples puertos atacando a un servidor o host haciendo que el servidor colapse y quede sin funcionar. La forma más común de realizar este tipo de ataque es a través de una red de Bots, siendo una técnica de ciberataque más usual y eficaz por su sencillez tecnológica.

Hay que tomar en cuenta que SDN puede llegar a simular funciones y características de un Firewall, dado que sus controladores pueden ejecutar una serie de comandos que actualizan direcciones MAC e IP filtrado de paquetes y de puertos y algunas características adicionales.

2.5.3. Software Defined Secure Networks (SDSN)

A través de la automatización de las redes y soporte de sistemas inteligentes y centralizados avanzados, con la clasificación de información de las amenazas y el malware, el *Software Defined Security Networks* (SDSN) identifica los riesgos con eficiencia y agilidad para ponerles fin con prontitud.

Se trata de ver a las redes como un ecosistema donde todos los componentes puedan actuar de forma sincronizada y orquestada para detectar y bloquear los ataques de malware y de red. Con esto la seguridad de las redes SDN va más allá que los típicos Firewall, IPS, IDS que trabajan de forma apartada.

La protección pasaría a ser gestionada por todos los componentes de la red lo que reducirá el tiempo de respuesta de los ataques, por medio de la automatización y de los sistemas centralizados que identificará las amenazas y los malware verificará la fuente y generará una cuarentena el objetivo principal es poder mantener un propio firewall en cada dispositivo que cuente la red.

2.6. Data Center definido por Software (SDDC)

En el centro de datos definido por software (SDDC) es gestionado por una infraestructura virtual y es proporcionado como servicio. El centro de datos definido por software es ejecutado por una SDN en la que cual mantiene la administración de tráfico y el procesamiento definido por software, calidad de servicio y almacenamiento definido por software.

Los nuevos administradores de infraestructura y operaciones TI utilizan el SDN por que mantiene mucha flexibilidad y agilidad en cada uno de los lineamientos del negocio de las empresas. Mejora también la disponibilidad de las aplicaciones así mismo con la convergencia y la eficiencia en los recursos de TI, ya que aumenta la adopción de los servicios en la nube.

2.6.1. Beneficios SDDC

Mayor agilidad y aprovisionamiento rápido de aplicaciones

Cada una de las configuraciones de los recursos de la tecnología de información está basada en políticas que permiten administrar cargas de trabajo en pocos minutos de acuerdo con la necesidad de urgencia, seguridad, etc. Cada uno de los componentes virtualizados se balancean entre ellos de manera automática.

Aumento de eficacia y reducción de costes

Generará reducción de costes ya que la automatización y la virtualización disminuirán gastos en recursos y personal IT, liberado de complejas tareas operativas tradicionales.

Alta disponibilidad y seguridad

La continuidad del negocio está segura con el cumplimiento de normativas definidas en las configuraciones. La automatización permitirá el conocer y controlar en todo momento cada uno de los recursos de red.

Flexibilidad garantizada: Nube publica, hibrida o privada

Cualquier carga de trabajo se ejecuta en cualquier lugar del mundo gracias a la flexibilidad de la nube, los usuarios tienen acceso a las aplicaciones de forma segura e inmediata desde todo tipo de ordenador, escritorio virtualizado o dispositivo móvil.

2.7. SD-WAN

En el estudio de las redes definidas por software se puede dar cuenta que cada vez esta solución crece y al momento se han expandido tanto que ahora existe

una red de área amplia definida por software (SD-WAN) que tiene un enfoque ágil basado en software para las redes extensas (WAN). Lo reemplazara al hardware de red tradicional como routers que su configuración se basa en (CLI) con alguna administración centralizada, el objetivo principal es poder reducir los costos de los enlaces WAN privados y aumentar el rendimiento y hacer que la red sea más ágil para la nube.

Actualmente la red WAN funciona enrutando el tráfico mediante enlaces MPLS, la red SD-WAN permite a los gestores de red poder reducir o eliminar la dependencia de costos de los circuitos MPLS arrendados y banda ancha o inalámbricos, así mismo SDN-WAN permite el sobre provisionamiento, reduciendo los gastos generales de la WAN.

2.7.1. SDN Híbrida

Una red híbrida es una red en la que los protocolos de las redes tradicionales como los de SDN trabajan en conjunto. Una red híbrida permite a los administradores de red agregar nuevas tecnologías SDN, como Openflow, en entornos heredados, sin una revisión completa de la red.

En una red híbrida el administrador de red podría ejecutar tecnologías SDN y protocolos de conmutación estándar de forma sincronizada sobre un hardware físico, se podría dividir el tráfico mediante el tráfico de control que permite SDN y controlar cierto tráfico con protocolos tradicionales mientras estos controlan la red.

Los Switches OpenFlow híbridos soportan la tecnología de conmutación tradicional Ethernet, en la actualidad existen fabricantes que admiten este tipo de convergencia entre estas 2 soluciones.

2.7.2. Funcionalidades de SD-WAN

Según (Rouse, 2015), en la figura 10 la tecnología SD-WAN ofrece muchas ventajas sobre las conexiones tradicionales entre ellas las siguientes:

- Se puede conectar a múltiples proveedores, esto permitirá mezclar y combinar o reemplazar conexiones MPLS con banda ancha esto permitirá tener un mejor SLA en el servicio.
- La escalabilidad y la capacidad en cada una de las sucursales con respecto al ancho de banda y las conexiones se puedes realizar posiblemente en minutos, no en semanas o días.
- El rendimiento de red mejorará y los problemas de saturación de paquetes se pueden solventar de forma inmediata.
- SD-WAN permitirá una mejor visualización de red
- Reduce los costos operativos generales.

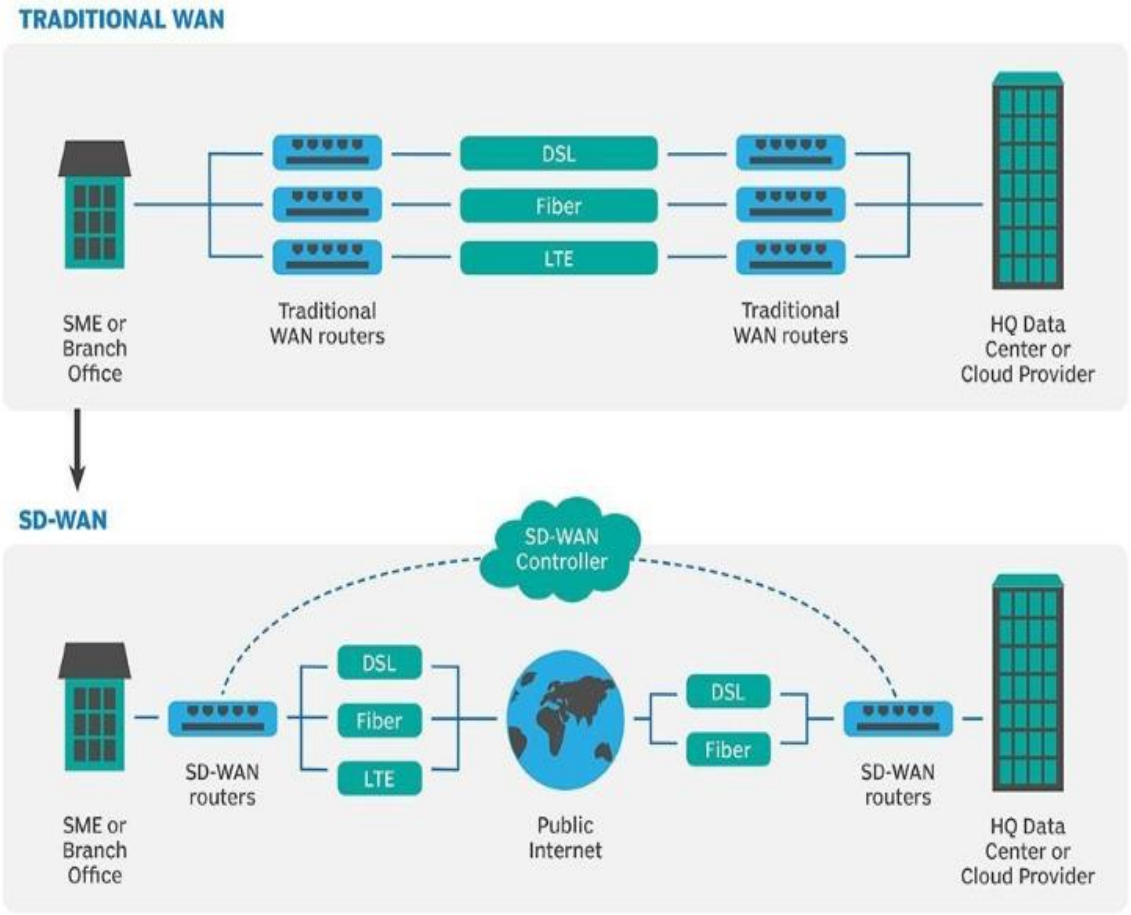


Figura 10. Tradicional WAN VS SD-WAN

Tomado de (Rouse, 2015)

3. CAPITULO III. INFRAESTRUCTURA ACTUAL DE LA PYMES.

En este capítulo se evaluará la situación actual de la infraestructura de la red de la PYMES y las condiciones de mejora en su red.

3.1. Infraestructura De la Red

La empresa tiene una infraestructura tradicional, esto quiere decir que su ambiente tecnológico está compuesto por equipos de redes físicos, sin contingencia alguna, lo que aumenta el riesgo a sufrir algún evento crítico como pérdida de conectividad e información.

A continuación, en la Tabla 3 se detalla los componentes de la Red servidores y de más dispositivos de la empresa.

Tabla 3
Componentes de la PYMES.

SERVICIO	MODELO O MARCA
Servidor de correo.	Zimbra, Centos 7.
Servidor de almacenamiento.	NAS *Synology*.
Firewall	*Sophos* UTM
Ubicaciones	Quito, Guayaquil
Numero de host	150
Servidor Web	Apache server
Ancho de banda	20 MB
Proveedor de internet y medio de trasmisión	Netlife / Fibra Óptica.
Comunicación Quito Guayaquil	Telconet /Fibra Óptica
Topología de Red	Tipo Estrella.
Switch capa 2	2 unidades / marca Cisco.
Routers	3 unidades /marca Cisco.

En la Figura 11 se puede observar el diseño de la arquitectura actual de la red con cada uno de los componentes físicos y lógicos con los que hacen que la red pueda comunicarse entre sí y de esta manera comparte sus datos hacia el internet, así como en su red interna.

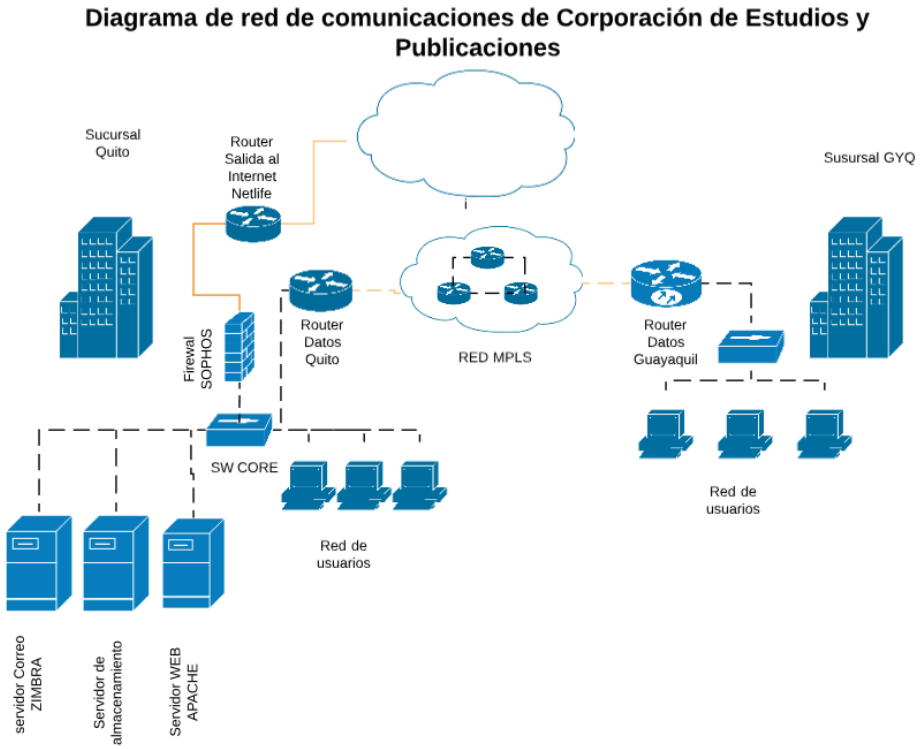


Figura 11. Diagrama de Red de datos de la PYMES.

3.2. Estado actual de la Red

El estado de la red mantiene algunos problemas como la escalabilidad y la actualización de dispositivos físicos, así como problemas con el consumo excesivo de ancho de banda de los usuarios y brechas de seguridad, por tal motivo la red está expuesta por desconocimiento de lo que sucede en el tráfico a nivel interno.

Mediante el equipo SOHOS UTM que se encuentra en plataforma Linux y sirve como firewall perimetral se puede evidenciar lo antes expuesto, adicional con *Wireshark* se puede identificar puertos desconocidos a los cuales los usuarios acceden y atacan a la red interna.

3.2.1. Ancho de banda

Para la salida al internet es necesario segmentar el tráfico que tendrán los usuarios con el fin de evitar consumos excesivos en internet ya que esto puede ocasionar lentitud en aplicaciones web importantes.

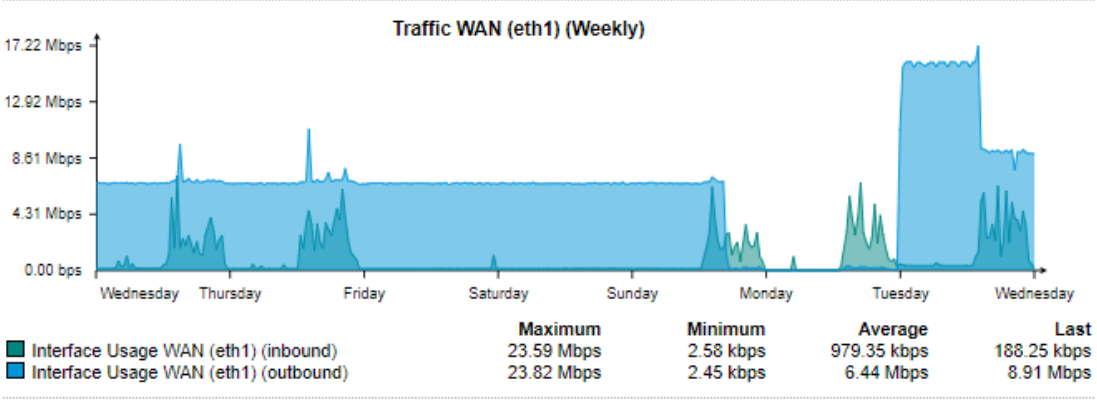


Figura 12. Ancho de banda interno

De acuerdo con la Figura 12 , el monitoreo corresponde a la interface (eth1) que es la salida WAN (red de área amplia) a través de la IP pública 186.4.249.34/25 otorgada por el ISP, la misma que está conectada al equipo de seguridad perimetral donde converge todo el tráfico de internet, se puede evidenciar que existe saturación en cuanto al ancho de banda, pues este llega a los 18 Mbps, en algunas ocasiones el tráfico de salida prevalece ante el tráfico de entrada por lo cual se puede concluir que se debe al uso de navegación o de compartición de archivos en la nube por cualquier sistema de transferencia como FTP (Protocolo de transferencia de archivos).

Según la figura 13 es una toma de la estadística de tráfico a los servicios que usan las aplicaciones se descubre que el tráfico que más consumen los usuarios es a través del protocolo HTTP (Protocolo de transferencia de hipertexto), tráfico no categorizado, él mismo que hace referencia a páginas con mala reputación que en muchas ocasiones presenta malware, conexiones para la trasferencia de archivos mediante el puerto SFTP (Secure File Tranfer protocole) en la cual proporciona compartición de archivos los cuales puede provocar fuga de información.

	Application	total Traffic	%
1	HTTP	2.7 GB	42.84
2	Unclassified	2.4 GB	38.24
3	SSL	312.0 MB	4.84
4	SFTP	204.5 MB	3.17
5	MS Online	172.5 MB	2.67
6	Eset	88.3 MB	1.37
7	Google	58.7 MB	0.91
8	Microsoft	58.0 MB	0.90
9	WhatsApp	52.2 MB	0.81
10	MS Office 365	51.5 MB	0.80

Figura 13. Tráfico web producido por los usuarios.

3.2.2. Seguridad

La seguridad es lo más importante para la compañía, por tal motivo la red de datos debe estar cubierta por capas de protección, tanto para el tráfico perimetral como para el tráfico Este -Oeste (tráfico interno), esto evitará presencia de ataques de red dirigidos o de día Zero también la protección contra ataques de tipo malware gusanos o troyanos etc.

En algunos casos existe malware o ataques de red que en muchas ocasiones toma tiempo en detectarlos e incluso se torna difícil controlarlos por no contar con una protección en capas separadas o simplemente no tener protección.

Mediante la herramienta Wireshark se pudo descubrir peticiones de tráfico hacia el puerto SSDP (Protocolo Simple de Descubrimiento de Servicios) es un protocolo, el mismo que utiliza para poder descubrir servicios de red y dispositivos extraíbles conectados en cada uno de los terminales. Observar en la Figura 14.

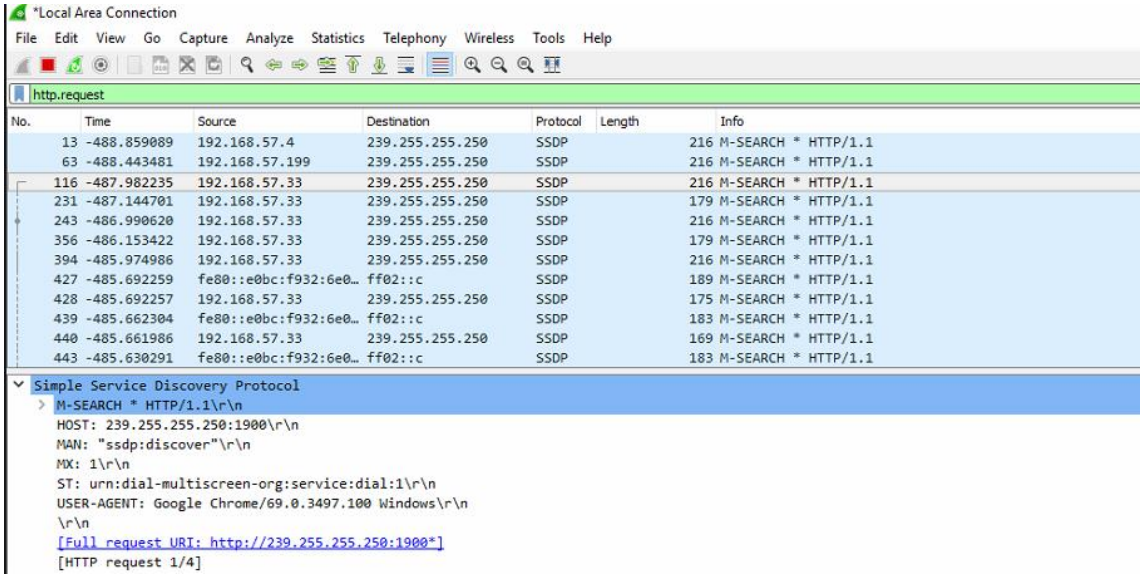


Figura 14. Protocolo de descubrimiento SSDP

Las violaciones de firewall están presentes en esta red, mediante la figura 15 se puede notar que los ataques de firewall que en muchos de ellos son ataques dirigidos que suceden cuando IPs externas quieren tener conexión mediante algún puerto especifico a nuestra red, en algunos casos instalan programas que reportan los puertos TCP / UDP abiertos y pueden atacar por cada uno de ellos.

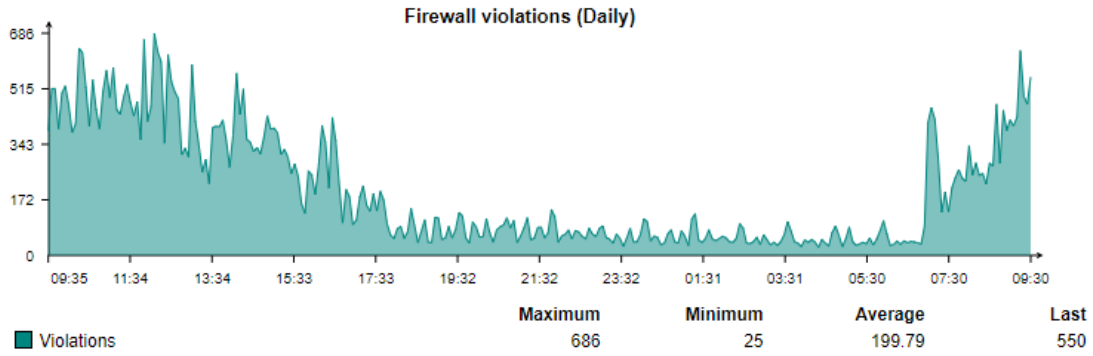


Figura 15. Ataques de firewall

En la Figura 16, se evidencia que los ataques de red proveniente de IPs de Estados Unidos, Brasil, Perú y Chile, atacan un puerto específico con mucha congruencia.

Top Destination Hosts ▼
Yesterday ▼
number of rows ▼

« » Update
Results: 1-20 of 668

Top	Destination IP	Packets	%	Services	%
1	186.4.249.34	14 278	20.95	1 435	43.11
2	192.188.57.115	7 302	10.71	1	0.03
3	192.168.57.100	5 343	7.84	11	0.33
4	216.239.35.0	2 705	3.97	1	0.03
5	224.0.0.1	2 132	3.13	1	0.03
6	74.82.35.79	2 030	2.98	1	0.03
7	192.16.48.200	1 233	1.81	1	0.03
8	188.172.244.158	1 050	1.54	1	0.03
9	204.79.197.200	924	1.36	1	0.03
10	13.107.21.200	901	1.32	1	0.03
11	10.40.101.255	863	1.27	3	0.09
12	172.31.0.255	860	1.26	1	0.03
13	162.250.2.190	825	1.21	1	0.03
14	37.252.251.222	817	1.20	1	0.03
15	192.168.57.70	742	1.09	3	0.09
16	188.172.252.94	691	1.01	1	0.03
17	181.39.186.11	672	0.99	1	0.03
18	217.146.28.158	669	0.98	1	0.03
19	192.168.57.120	643	0.94	108	3.24
20	192.168.57.137	583	0.86	98	2.94

Figura 16. Informe de ataques de red mediante el origen.

En la Figura 17, podemos detectar ataques al puerto tcp/50839 con demasiadas peticiones, este tráfico está dirigido con el fin de poder infectar a algún equipo de la red interna.

Top Services ▼ Source IP/Net: 13.107.5.88

Yesterday ▼

number of rows ▼

« » Update

Results: 1-20 of 341

Top	Service	Packets ▼	%
1	(tcp/50839)	12	0.59
2	(tcp/50563)	6	0.29
3	(tcp/64445)	6	0.29
4	(tcp/14404)	6	0.29
5	(tcp/5911)	6	0.29
6	(tcp/14665)	6	0.29
7	(tcp/15075)	6	0.29
8	(tcp/54522)	6	0.29
9	(tcp/9460)	6	0.29
10	(tcp/52355)	6	0.29
11	(tcp/14240)	6	0.29
12	(tcp/58386)	6	0.29
13	(tcp/52570)	6	0.29
14	(tcp/11012)	6	0.29
15	(tcp/53142)	6	0.29
16	(tcp/51393)	6	0.29
17	(tcp/51874)	6	0.29
18	(tcp/54547)	6	0.29
19	(tcp/14532)	6	0.29

Figura 17. Informe ataques de red basada en puertos de conexión.

3.2.3. Uso de recursos de *hardware*

Su equipo de seguridad perimetral se encuentra con una memoria y procesamiento elevado debido a un crecimiento de los usuarios y también debido a las peticiones de internet que convergen en este equipo. Se lo puede evidenciar en la Figura 18.

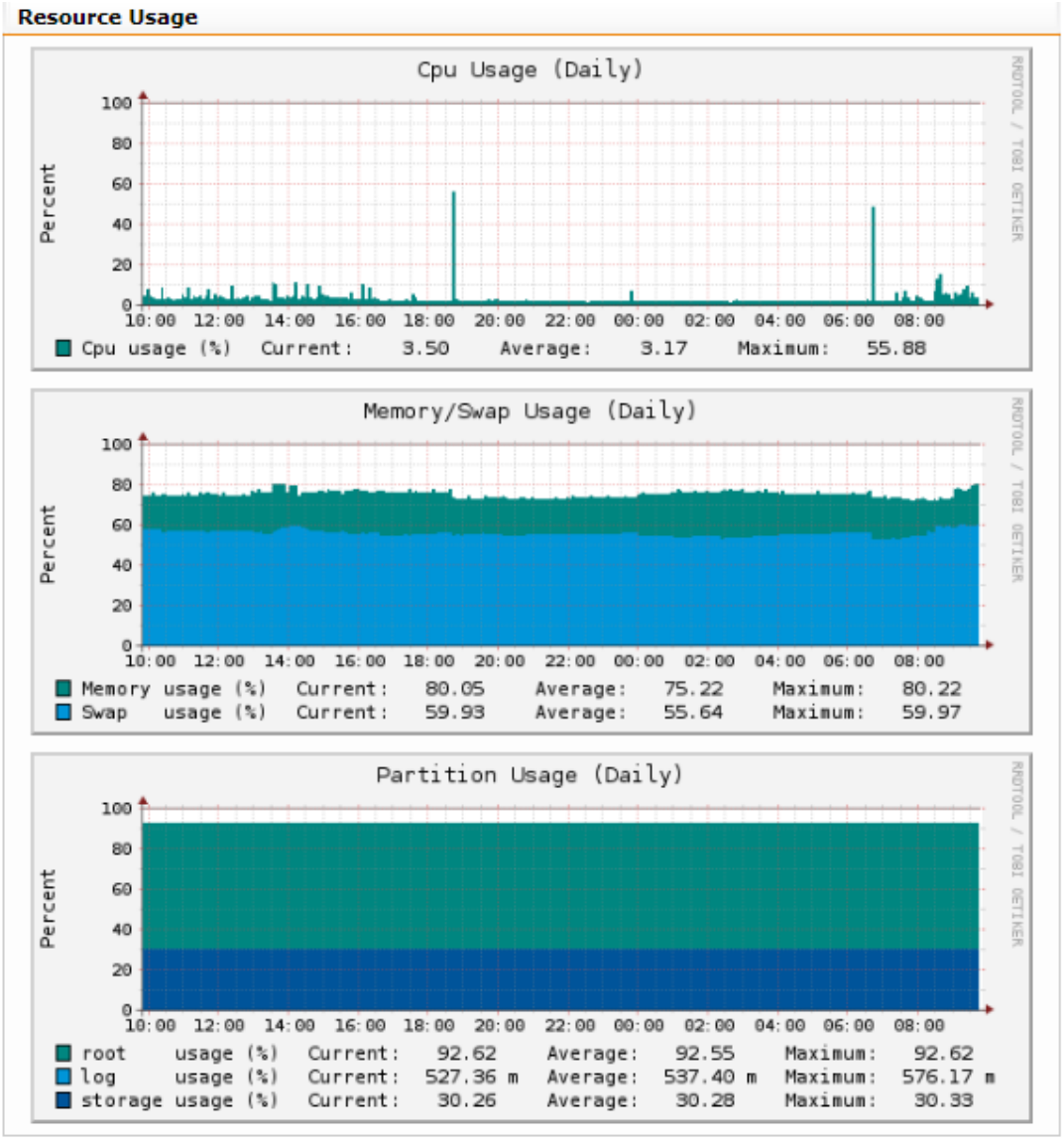


Figura 18. Uso de disco memoria y CPU

Al tener equipos físicos cuyo funcionamiento es la comunicación de la Red y la seguridad, dificulta tomar acciones como aumentar de forma manual y lógica los requerimientos de hardware y exponer a que un equipo al fallar no cuente con una solución inmediata.

3.2.4. Tráfico de red

El tráfico de comunicación converge en el Switch de Core. Según la Figura 19 se puede observar que en algunas ocasiones llega a generar picos de tráfico

manteniendo varias solicitudes de Ping generando lentitud de la red este resultado se obtuvo bajo la herramienta *Wireshark, donde se descubre* que existe la IP 192.168.57.52 /24, la misma que tiene conexiones hacia IPs externas con mucha frecuencia.

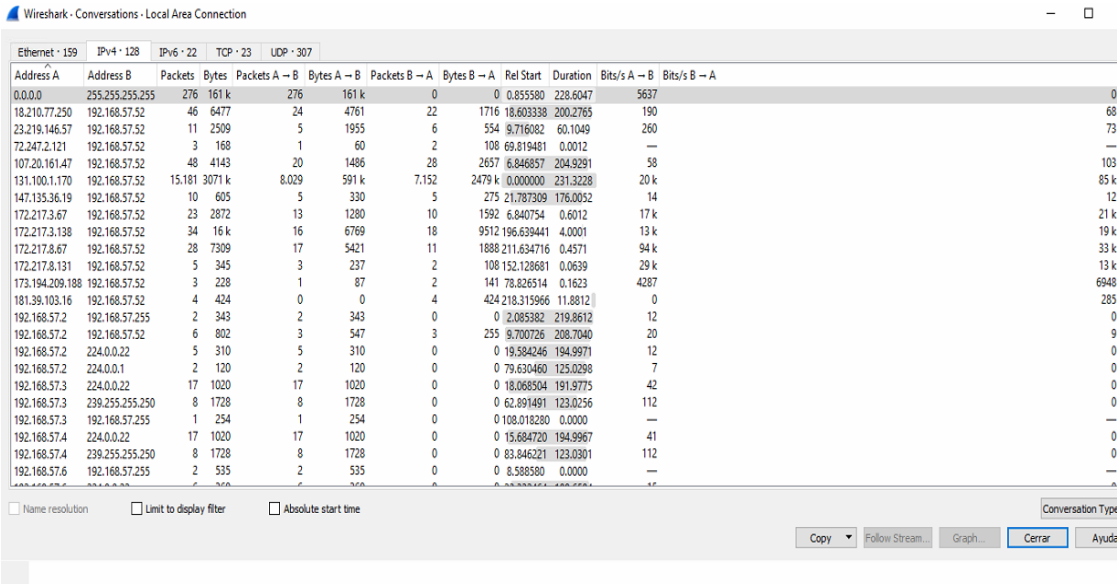


Figura 19. Tráfico de red de acuerdo con los paquetes de envío y recepción.

3.3. Aspectos que considerar

En esta red existen varios puntos clave que se debe considerar al momento de construir una red de datos, por ejemplo, la disponibilidad y operatividad del servicio. Para la Pymes es fundamental el uso del internet y la comunicación con su sucursal Guayaquil por la compartición de datos y de información, por ende, la estabilidad y eficiencia de todos sus elementos de red deben estar totalmente operativos.

Su proyección de crecimiento es de aproximadamente 5 usuarios cada 6 meses, así mismo la cantidad de ancho de banda se puede reducir al no tener segmentado o un control sobre su navegación.

Su estructura física aún mantiene switches de capa de 2100 Mbps conectados con cables ethernet cat 5 sin un sistema de redundancia ni de alta disponibilidad.

En la arquitectura actual se conectan los ordenadores a una red tradicional. Se recomienda centralizar la conexión de la red y distribuir todos los recursos y procesos tecnológicos que llevan a cabo la comunicación de datos. Además, se recomienda tener un control total de la red de datos y reducir el tiempo de crecimiento de red y mejorar tiempos de impacto en su centro de datos.

3.4. Aspectos favorables con SDN

Al ser una empresa cuya infraestructura de red es de forma tradicional, se observa que la red es compleja por tantos dispositivos conectados, fabricantes de software que no permiten una escalabilidad inmediata dificultando la gestión, el control de forma centralizada e imposibilita a la red establecer políticas de seguridad y calidad de servicio de extremo a extremo, por último, se depende mucho de los proveedores de servicio.

Bajo este antecedente el diseño de redes SDN permitirá tener una mejor gestión de los componentes de red, redundancia y la demanda de mayores anchos de banda, cambio de patrones de tráfico y la aparición de nuevos servicios con requerimientos diferentes a los tradicionales.

SDN permitirá mayor eficiencia mediante la posibilidad de la asignación de recursos de forma dinámica. Con respecto a la seguridad SDN tendrá la visibilidad de todo el tráfico ESTE-OESTE de la red con el fin de poder ver cada uno de los protocolos que circulan en la red.

La empresa se beneficiará de las Redes SDN por los siguientes motivos:

- Abaratará costos con respecto al Hardware de red.
- Reducción de personal técnico para poder montar recursos de red.
- Mejorará el uso de algoritmos de enrutamiento y el uso de la capacidad de la red puede ser mayor.
- Control centralizado de la red.

- Se puede administrar de forma eficaz el ancho de banda de red, delimitando o autorizando el uso de diferentes dispositivos.
- Ante una caída inesperada del sistema se puede actuar con rapidez evitando que el impacto sea menor.

Luego de analizar estos puntos, se puede entender que SDN nos permite actualizarnos y dar un giro en la búsqueda de soluciones a todos los inconvenientes que pueden existir en las redes tradicionales como: costos de mantenimiento, soporte técnico, estabilidad a la hora de implementar una nueva red de datos.

SDN permitirá el enrutamiento de paquetes de forma específica por diversos caminos para poder evitar la congestión de la red.

Como se puede evidenciar en la figura 20, el plano de datos y de control es separado el protocolo de comunicación es Openflow es mismo que proporciona que la red sea programable en los switches.

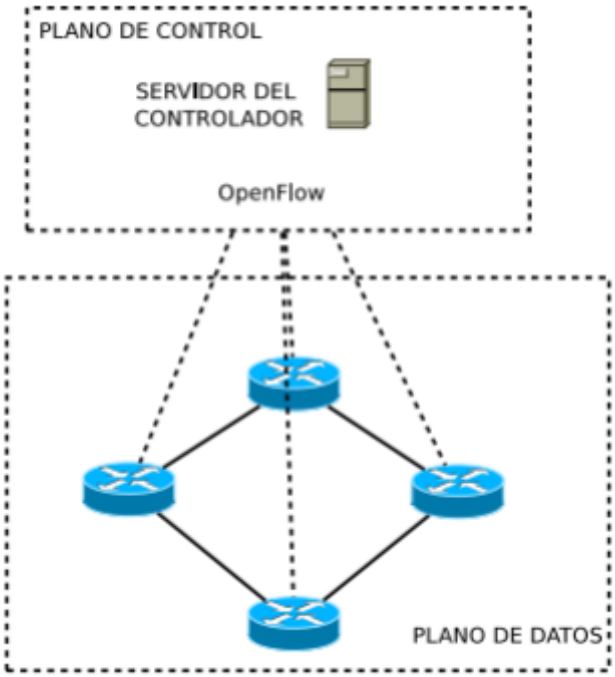


Figura 20. Red SDN CON PORTOCOLO OPEN FLOW
Tomado de (Gonzalez, 2014)

4. CAPITULO IV. DISEÑOS DE LA RED SDN EN LA PYMES

Este capítulo detalla el diseño de la red SDN en el emulador Mininet bajo el protocolo OpenFlow con base a la Red actual de la Pymes. Se plantea el diseño de manera dinámica, donde se podrá observar en tiempo real el comportamiento del flujo de datos mediante el simulador de red Mininet y el controlador Floodlight, el mismo que será demostrado en la simulación. Al observar la funcionalidad de la Red SDN se definirá la intervención de está para abarcar las necesidades de la empresa.

Para poder emular una red SDN se debe separar el control de los datos. Para eso se trabaja con el controlador emulado donde se puede tener toda la administración de la red. En la capa de datos se diseñará una red en una estructura plana tipo estrella con Mininet.

4.1. Diseño del Controlador Floodlight

En la figura 21 se muestra como está compuesto el controlador Floodligth uno de sus componentes es el módulo de aplicaciones donde se instalan diferentes APIS como por ejemplo políticas de firewall o reglas de control de acceso. El controlador gestiona toda la red y la operatividad de sus paquetes con los cuales se comunican entre host, así mismo podrá enrutar todos los paquetes mediante el protocolo OpenFlow, además, cuenta con una visualización de forma gráfica mediante un navegador en la cual podemos hacer diferentes configuraciones en cada uno de los servicios.

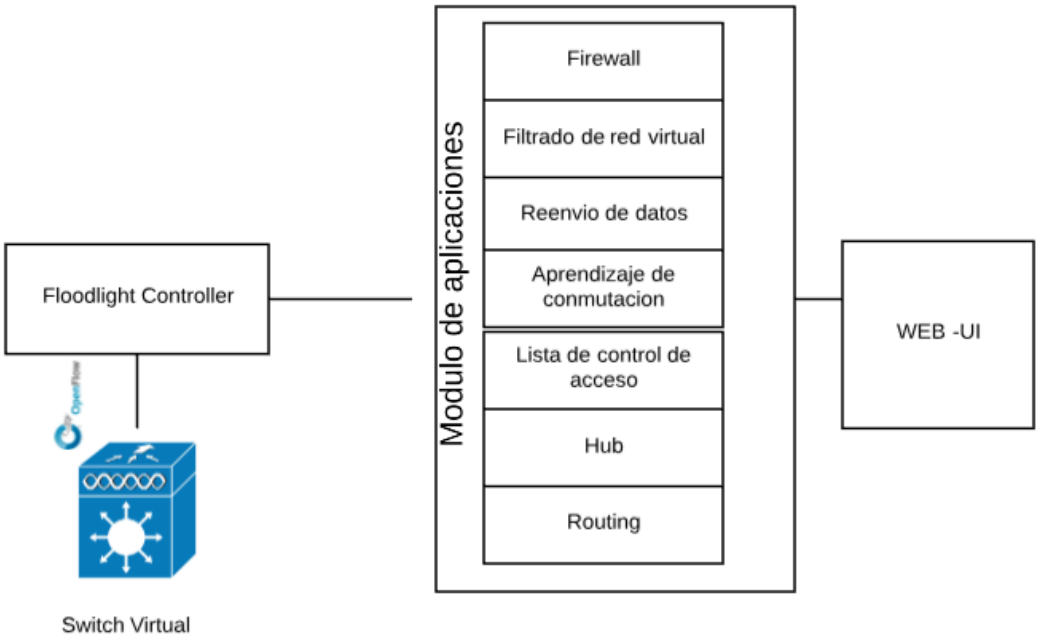


Figura 21. Diseño del controlador Floodlight.

4.2. Diseño de la Red SDN en Mininet

Se estableció 2 segmentos de red SDN Quito y Guayaquil en el emulador Mininet con los dispositivos virtuales que permitieron la comunicación y las políticas de seguridad. Para el diseño de la red en Mininet se realizó la integración de Miniedit siendo esta una interface gráfica de usuario (GUI) permitió incorporar dispositivos virtuales de manera sencilla para la simulación como host, switches, Switch Openflow, routers y links. Observar Figura 23.

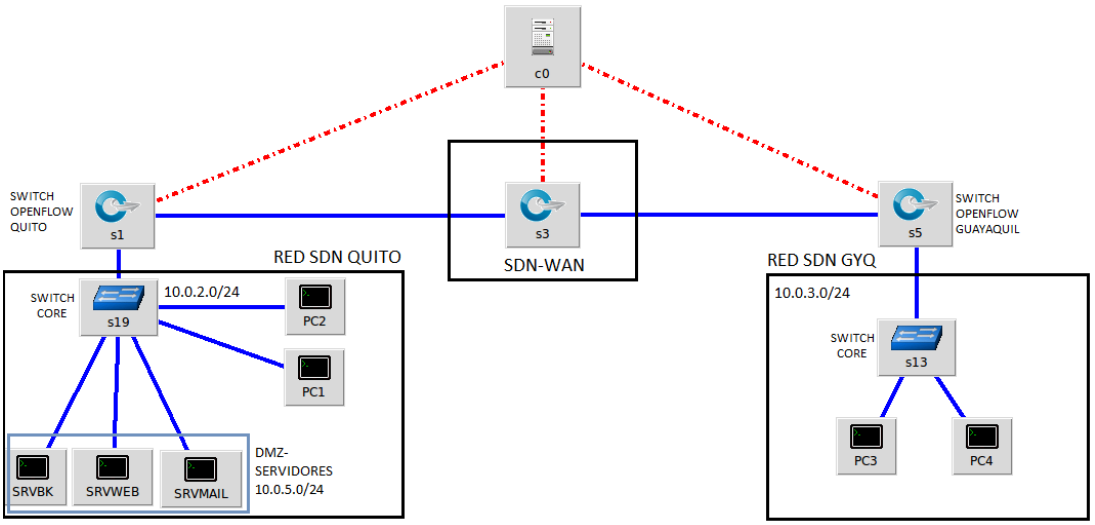


Figura 23. Diseño SDN de la PYMES en Miniedit.

A continuación en la tabla 4 se especifican los componentes de la red SDN simulada con los que se realizó cada una de las emulaciones.

Tabla 4
Componentes de la red simulada SDN.

Componente virtual	**Detalle**
Controlador	Floodligth
1 Switch Virtual	Quito
1 Switch Virtual	Guayaquil
1 Switch MPLS	Conexión entre sucursales
2 PC Virtuales 3 Servidores	Quito
2 PC Virtuales	Guayaquil
Protocolo de comunicación	OpenFlow

4.2.1. Comandos utilizados para la creación de redes en Mininet

En la tabla número 5 se explican la línea de comandos y su descripción que se utilizó para poder trabajar en la creación de redes SDN mediante Mininet.

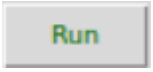
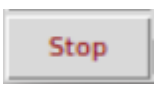
Tabla 5
Comandos Para redes en Mininet.

UTILIDAD	DESCRIPCIÓN
Switch= SWITCH	Este componente fue compuesto para permitir la automatización redes y está diseñado para la distribución en múltiples servidores físicos similares al conmutador virtual distribuido vNetwork de VMWare o al Nexus 1000V de Cisco.
Controller CONTROLLER	Es el controlador que trabaja con el protocolo OpenFlow permite el control de la red.
Topo=TOPO	Permite crear topologías de red virtual.
mac	Permite asignar direcciones MAC a los host de forma automática
Help	Muestra los comandos utilizables para el desarrollo de la red virtual
nodes	Permite ver una lista de los dispositivos virtuales
net	Permite ver todas las conexiones
dump	Permite verificar direccionamiento IP e identificadores
xterm	Abrir una nueva ventana para gestionar directamente con el con el host.

4.2.2. Componentes utilizados en Miniedit

En la tabla 6 se describen los componentes con su respectiva descripción utilizados al trabajar en un entorno grafico en la creación de las redes SDN.

Tabla 6
Componentes MINIEDIT

Elemento de simulación	Funcionalidad
Selección 	Permite mover cada uno de los elementos y conectarlos entre si y ubicarlos en la posición deseada.
Host 	Permite crear nodos de redes conectarlos entre si y se puede configurar parámetros de red.
Switch Open Flow 	Permite crear switches bajo el Open Flow los mismos que deben ir conectados a un controlador, estos Switch son configurables según los parámetros a definir.
Legacy Switch 	Permite crear conexiones de manera independiente son switches capa 2 solo para él envió y recepción de datos.
Legacy Router 	Permite la creación de routers independiente de un controlador en capa 2.
Netlink 	Permite la conexión de enlaces entre nodos, se pueden unir de acuerdo a la topología creada.
Controlador 	Permite crear la conexión a un controlador externo o un interno, se pueden tener varios controladores creados para una misma red, el usuario puede configurar de acuerdo con la necesidad de la red.
Ejecución 	Permite ejecutar la topología de forma lógica.
Detener 	Permite detener la simulación del proyecto en Mininet.

5. CAPITULO IV. SIMULACIÓN DE LA RED SDN

En este capítulo se procederá a ejecutar la simulación de la red, lo cual permitirá visualizar cada una de las características y funciones que se tiene con la red la red SDN.

La simulación está compuesta de 2 elementos de software principales: un simulador MININET y un controlador SDN Floodlight, los cuales se eligieron por licenciamiento gratuito, así mismo por la compatibilidad con ambientes virtuales, lenguaje de programación etc.

Mininet y Floodlight se instalaron en un ambiente virtual con Oracle VM VirtualBox en plataformas Linux, en una versión de Ubuntu 14.0.

5.1. Instalación Mininet

Para proceder con la instalación es necesario descarga el software. Esta descarga se la puede realizar en la página http://mininet.org donde se detallará las siguientes opciones para comenzar con la instalación:

- Opción 1: Instalación de Mininet VM (fácil, recomendado)
- Opción 2: Instalación nativa desde la fuente
- Opción 3: Instalación desde paquetes
- Opción 4. Actualizar una instalación existente de Mininet

Se escogerá la **opción 2**, pues fue la opción que más se acopla al escenario de simulación debido a que admite las versiones más recientes de Open Vswitch.

Para instalar de forma nativa desde la fuente, primero se requiere obtener el código fuente de Mininet.

Dentro de Ubuntu abrimos un terminal y ejecutamos el siguiente comando que permitirá realizar una actualización de software. Se escogerá la última y mejor versión de Mininet y se descargará de la página oficial.

git clone git://github.com/mininet/mininet

Se ejecuta el siguiente comando para poder comprobar que se ha instalado todos los componentes y luego hacemos un check-out a la versión de Mininet 2.2.1.

cd mininet
git tag # lista de versiones disponibles
git checkout -b 2.2.1 2.2.1 # verificar la versión instalada
Una vez que se realiza dicha comprobación procedemos a verificar las opciones de instalación de Mininet bajo el siguiente código de comando.

mininet/útil/install.sh (opciones)
Existen opciones de instalación las cuales son:
Mininet/útil/install.sh -h: Se instala todos los componentes que incluyen las máquinas virtuales, incluye también dependencias como Open Vswitch, así los protocolos OpenFlow y POX.
Mininet/útil/install.sh -nfv: Se instala Mininet solo con el protocolo Openflow y Open vSwitch.
Mininet/útil/install.sh – mydir: Usamos esta opción cuando queremos colocar árboles de origen / compilación en un directorio específico en lugar de en su directorio principal.
Se seleccionó la primera opción
install.sh -h
Para poder comprobar que Mininet está instado se ejecutó el siguiente comando
sudo mn – comprobar que la red esté disponible

El resultado es presentado en la Figura 23. Se puede observar que ejecuta una red por defecto la misma que consta de componentes de red como 2 host virtuales, 1 controlador, 2 Switch virtuales además realiza un ping entre host para comprobar la operatividad de la red.

```
root@mininet-VirtualBox:/home/mininet# sudo mn --test pingall
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Waiting for switches to connect
s1
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
*** Results: 0% dropped (2/2 received)
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 6.550 seconds
```

Figura 22. Red por defecto Mininet

5.1.1. Diseñando la red SDN en Mininet

Existen dos maneras para poder construir la red SDN:

1. Utilizando el lenguaje de programación Python para poder crear cada uno de los componentes.
2. Utilizando el software Miniedit que permite crear los componentes de manera gráfica.

En la simulación se creará la Red bajo la opción 2. Para lo cual se ejecuta la siguiente línea de comando.

root@mininer-VirtualBox:/home/mininet/mininet/examples# ./miniedit.py

Se despliega un panel en el cual se genera la red que se requiere como muestra la Figura 24. Está creada la red a simular con sus respectivos componentes detallados en la tabla 5. Previamente indicada.

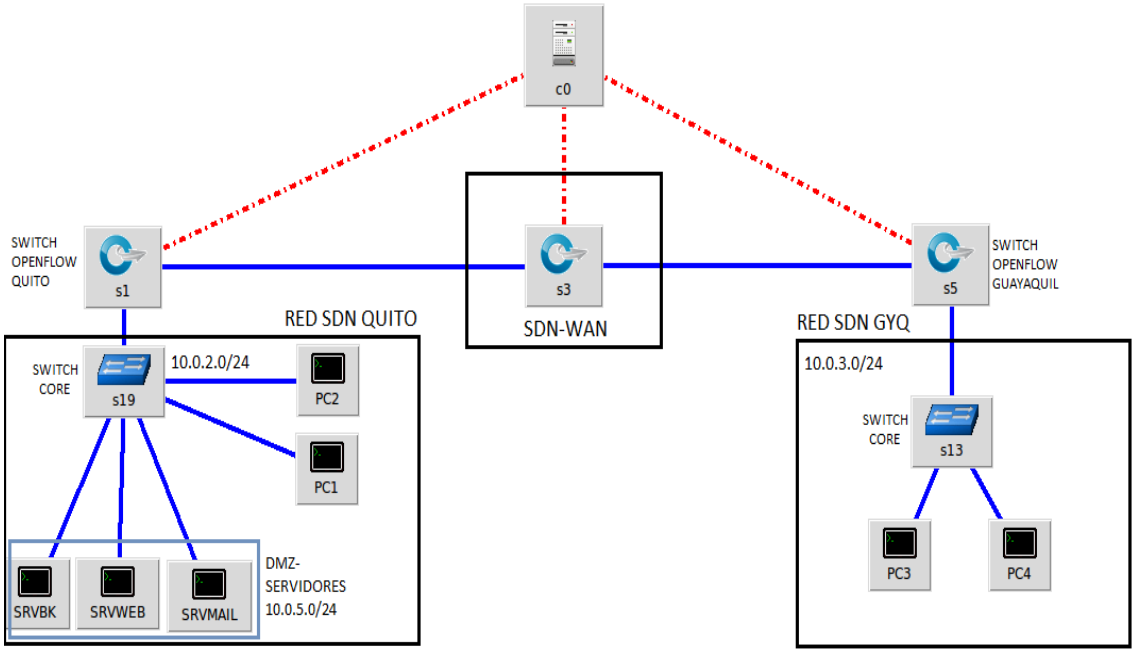


Figura 23. Topología de red SDN en Mininet para simulación

5.1.1.1. Direccionamiento IP de la red simulada

A continuación, en la tabla 7 y 8 se detalla el direccionamiento IP de las sucursales Quito y Guayaquil utilizado para el diseño de la red SDN.

Tabla 7
Direccionamiento IP Sucursal Quito.

DISPOSITIVO VIRTUAL	DIRECCIONAMIENTO IP
Controlador	10.0.2.15
SW UIO 1	10.0.6.1
PC1	10.0.2.21
PC2	10.0.2.22
SRVBK	10.0.5.2
SRVWEB	10.0.5.3
SRVMAIL	10.0.5.4

Tabla 8
Direccionamiento IP sucursal GYE.

DISPOSITIVO VIRTUAL	DIRECCIONAMIENTO IP
Controlador	10.0.2.15
SW GYQ 1	10.0.6.2
PC3	10.0.3.24
PC4	10.0.3.25

5.1.1.2. Configuración controladora Mininet

Se seleccionó 1 controlador dentro de Miniedit, el mismo que permitió configurar el apuntamiento correspondiente a un controlador externo, para esto se escogió el tipo de controlador **"REMOTE CONTROL"** y se apuntó a nuestro controlador Floodlight con la IP 10.0.2.15 mediante el puerto 6653 como muestra la Figura 25.

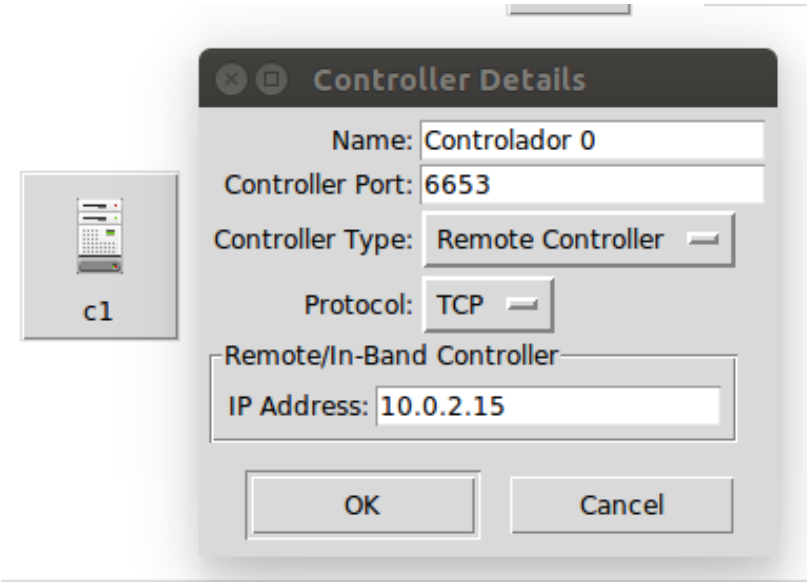
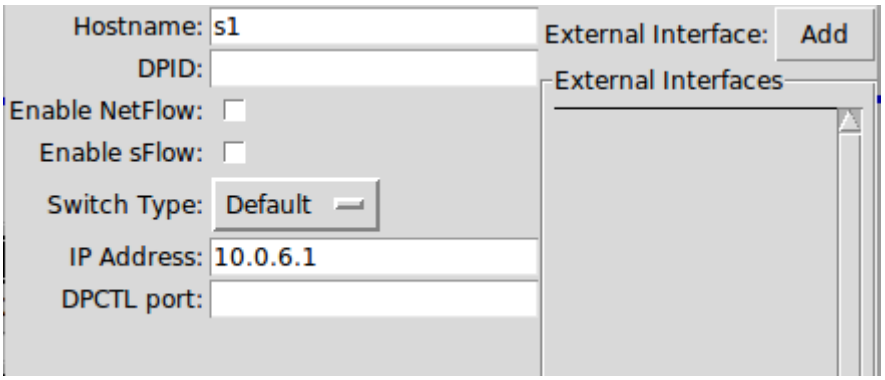
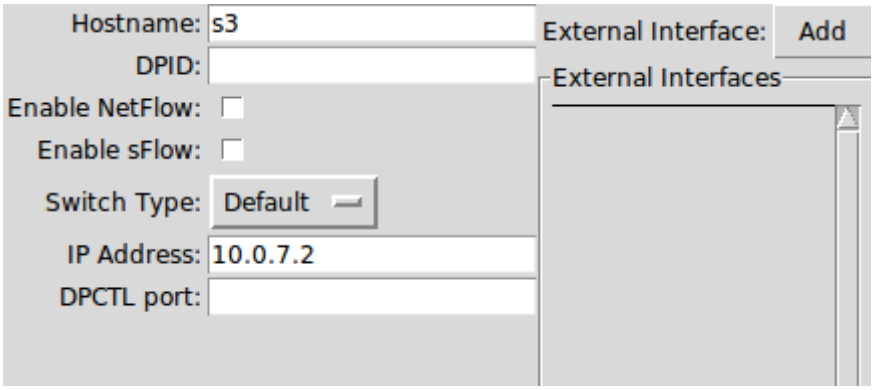
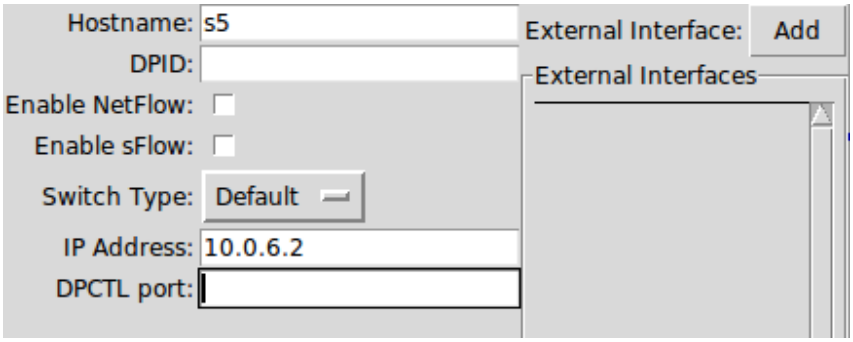


Figura 24. Configuración del controlador

5.1.1.3. Configuración VSwitch

En la Figura 26, se puede ver la creación de los Switches virtuales utilizando OpenFlow como protocolo por defecto, en cuanto a las interfaces se las puedo aumentar en el momento que se necesite y también se especifica el direccionamiento IP para poder generar Vlans o listas de control de acceso, se puede activar el monitoreo mediante Netflow que es un protocolo de red desarrollado por cisco para información sobre el tráfico de red o SFLOW que proporciona un medio para exportar paquetes truncados, junto con contadores de interfaz para el monitoreo de la red.

Tabla 9
Configuraciones de los Switches virtuales

Switch Virtual sucurzal Quito	
Switch Virtual MPLS	
Switch Virtual sucurzal Guayaquil	

5.1.1.4. Configuración de los Host.

En la Figura 27 y 28 se puede observar las configuraciones que permite el emulador Mininet dentro del host, se modificó los siguientes parámetros: Direccionamiento IP, configuración de Vlans. En esta simulación se utilizó la configuración de IPv4 para las conexiones correspondientes de acuerdo al direccionamiento IP de la tabla 6.

Tabla 10

Configuraciones hosts Red SDN Quito

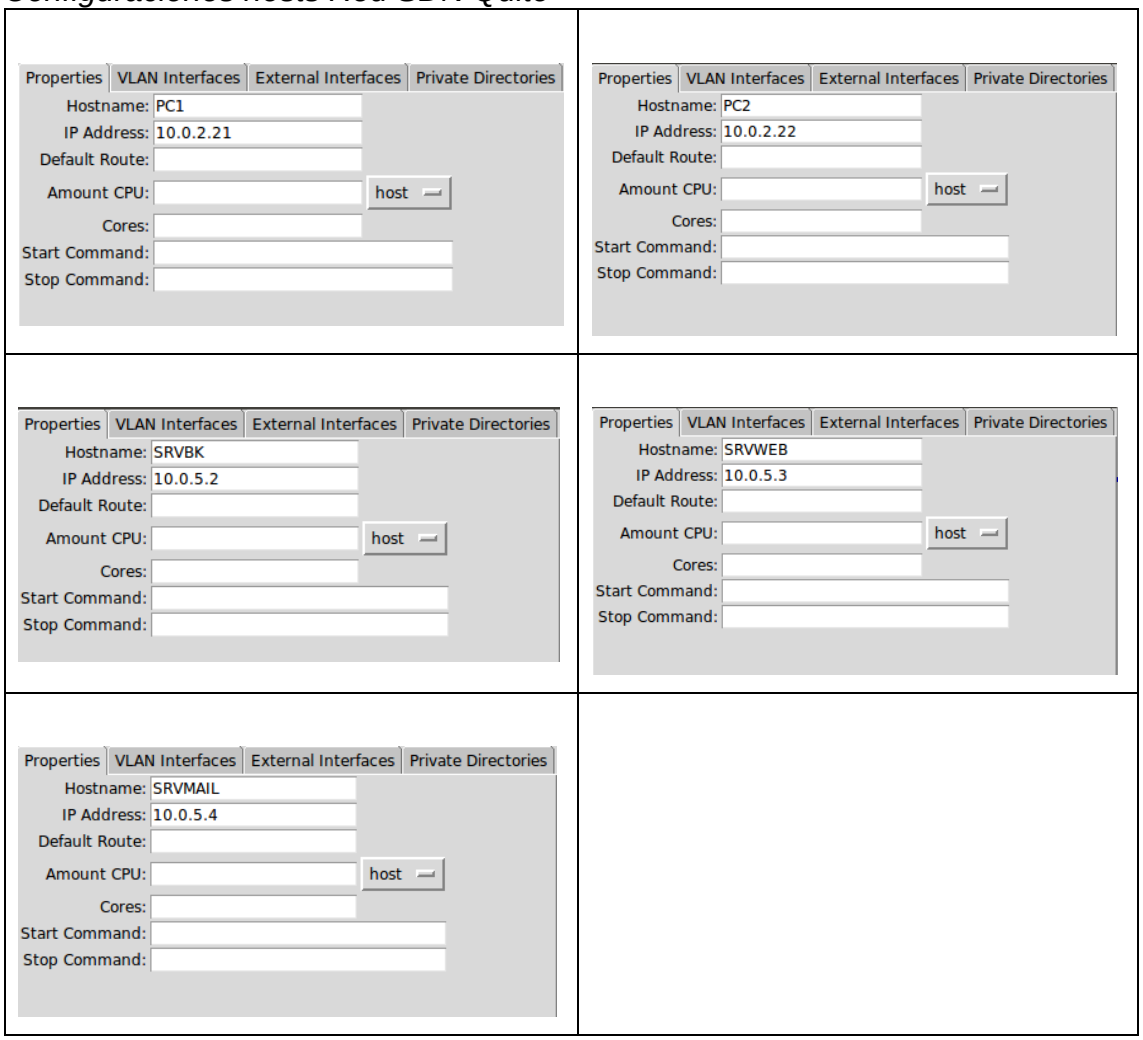
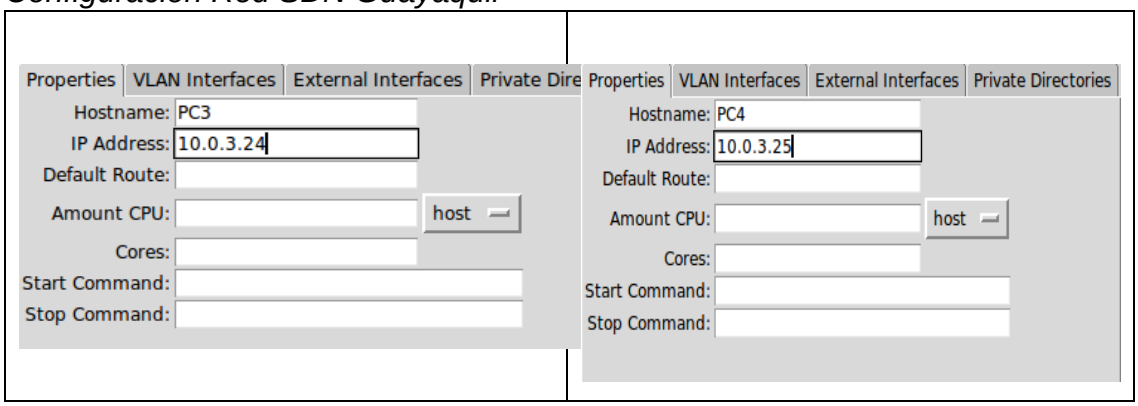


Tabla 11

Configuración Red SDN Guayaquil



5.1.1.5 Programación Python

Aunque la herramienta Mininet crea automáticamente la topología. En el lenguaje de programación Python se modifica el código fuente de la red creada. Se creó nuevos hosts y diferentes conexiones, adicionalmente se exportó el código y se lo transportó a un entorno diferente, el código de la configuración se muestra en la Figura 29.

Para ver el código fuente se ejecutó este comando dentro de nuestro ambiente Mininet.

root@mininer-VirtualBox:/home/mininet/mininet/examples# nano./(nombre del proyecto creado).py

```
#!/usr/bin/python

from mininet.net import Mininet
from mininet.node import Controller, RemoteController, OVSController
from mininet.node import CPULimitedHost, Host, Node
from mininet.node import OVSKernelSwitch, UserSwitch
from mininet.node import IVSSwitch
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.link import TCLink, Intf
from subprocess import call
from mn_wifi.net import Mininet_wifi
from mn_wifi.node import OVSKernelAP, UserAP
from mn_wifi.cli import CLI_wifi

def myNetwork():

    net = Mininet_wifi( topo=None,
                  build=False,
                  ipBase='10.0.0.0/8')

    info( '*** Adding controller\n' )
    c0=net.addController(name='c0',
                      controller=RemoteController,
                      ip='10.0.2.15',
                      protocol='tcp',
                      port=6653)
```

Figura 25. Programación Python

```
info( '*** Add switches/APs\n')
s2 = net.addSwitch('s2', cls=OVSKernelSwitch)
s3 = net.addSwitch('s3', cls=OVSKernelSwitch)
s1 = net.addSwitch('s1', cls=OVSKernelSwitch)

info( '*** Add hosts\n')
PC1 = net.addHost('PC1', cls=Host, ip='10.0.2.21', defaultRoute=None)
SRVWEB = net.addHost('SRVWEB', cls=Host, ip='10.0.5.3', defaultRoute=None)
SRVMAIL = net.addHost('SRVMAIL', cls=Host, ip='10.0.5.4', defaultRoute=None)
PC3 = net.addHost('PC3', cls=Host, ip='10.0.3.24', defaultRoute=None)
PC2 = net.addHost('PC2', cls=Host, ip='10.0.2.22', defaultRoute=None)
PC4 = net.addHost('PC4', cls=Host, ip='10.0.3.25', defaultRoute=None)
SRVBK = net.addHost('SRVBK', cls=Host, ip='10.0.5.2', defaultRoute=None)

net.configureWifiNodes()
info( '*** Add links\n')
net.addLink(s1, SRVBK)
net.addLink(s1, SRVWEB)
net.addLink(s1, SRVMAIL)
net.addLink(s1, PC2)
net.addLink(s1, PC1)
net.addLink(s1, s3)
net.addLink(s3, s2)
net.addLink(s2, PC3)
net.addLink(s2, PC4)

info( '*** Starting network\n')
net.build()
info( '*** Starting controllers\n')
for controller in net.controllers:
    controller.start()

info( '*** Starting switches/APs\n')
net.get('s2').start([c0])
net.get('s3').start([c0])
net.get('s1').start([c0])
    net.addLink(s2, PC3)
    net.addLink(s2, PC4)

    info( '*** Starting network\n')
    net.build()
    info( '*** Starting controllers\n')
    for controller in net.controllers:
        controller.start()

    info( '*** Starting switches/APs\n')
    net.get('s2').start([c0])
    net.get('s3').start([c0])
    net.get('s1').start([c0])

    info( '*** Post configure switches and hosts\n')

    CLI_wifi(net)
    net.stop()

if __name__ == '__main__':
    setLogLevel( 'info' )
    myNetwork()
```

Figura 26. Programación Python

5.1.2. Instalación Controlador Foodligth

El controlador Floodlight permite descargar una máquina virtual que ya contiene el software del controlador, así como Wireshark para lo cual se descargó del siguiente link:

http://www.projectfloodlight.org/download/

Después de la descarga se exportó la máquina virtual en el hipervisor Virtual Box con los requerimientos de hardware explicados en la Tabla 9.

Tabla 12
Requerimiento de Hardware para el controlador Floodlight

Memoria Ram	7168 MB
Procesador	2 núcleos
Almacenamiento	80 GB
RED	Se define una red interna 10.0.2.x

Una vez ejecutada la máquina virtual apareció la siguiente pantalla de inicio y se ingresó con las siguientes credenciales, tal como se indica en la Figura 30.
User: floodlight
Pass: floodlight

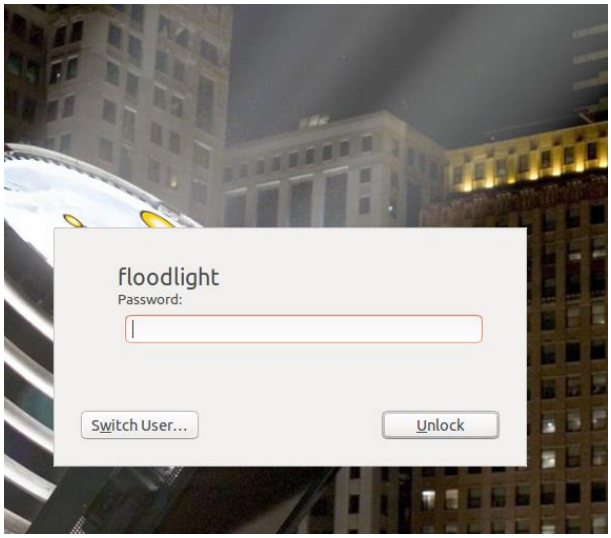


Figura 27. Panel de inicio controlador Floodlight.

Este controlador funciona con versión de Java 8 y para iniciarlo fue necesario ejecutar la siguiente línea de comando:

$ java -jar target/floodlight.jar

Al ejecutar este comando se levantó automáticamente las librerías y módulos del controlador, así como el modo de visualización Web mediante la IP 10.0.2.15 por el puerto de conexión 8080. Observar Figura 31.

```
; Linux x86_64; rv:38.0) Gecko/20100101 Firefox/38.0    http://10.0.2.15:8080/ui
/pages/index.html
16:04:02.877 DEBUG [LogService:Dispatcher: Thread-22] Processing request to: "ht
tp://10.0.2.15:8080/wm/core/module/all/json"
16:04:02.877 DEBUG [LogService:Dispatcher: Thread-29] Processing request to: "ht
tp://10.0.2.15:8080/wm/core/module/loaded/json"
16:04:02.924 INFO [LogService:Dispatcher: Thread-29] 2018-11-08 16:04:02       1
0.0.2.15                -       8080    GET     /wm/core/module/loaded/json    -
200     -       0       47      http://10.0.2.15:8080   Mozilla/5.0 (X11; Ubuntu
```

Figura 28. Imagen del Log de inicio del controlador

5.1.2.1. Panel principal del Controlador Foodlight Open Flow.

En la Figura 32 se evidencia el panel principal del controlador, cuya función principal es controlar toda la red SDN diseñada en Mininet. Además, se estableció políticas de seguridad.

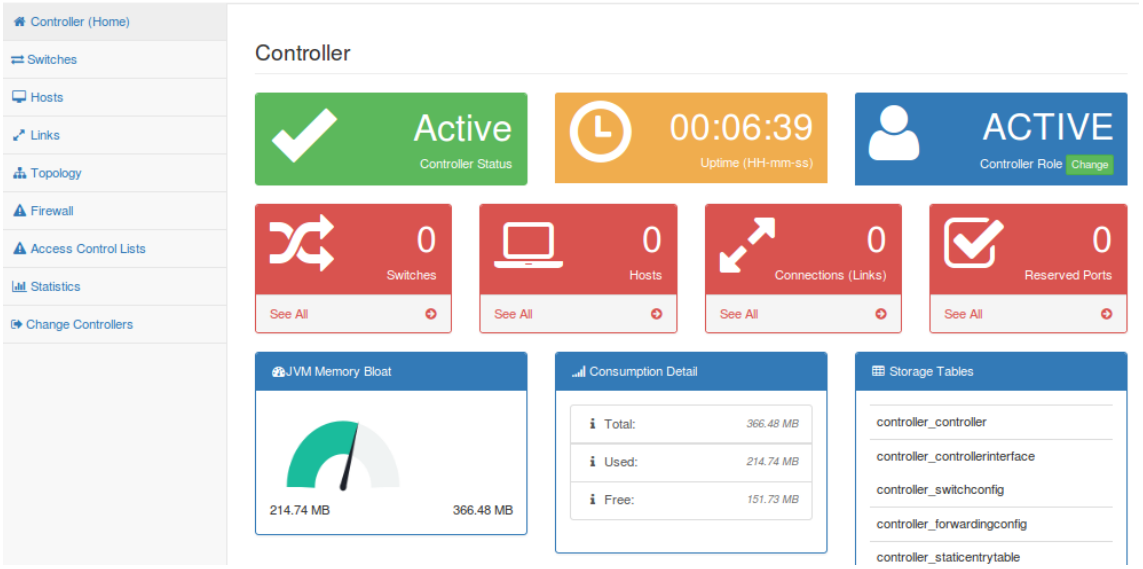


Figura 29. Panel principal Floodlight

5.1.3. Escenario de emulación conectividad de la red

El primer escenario fue evaluar la conectividad mediante la red SDN simulada en MININET y la conexión hacia el controlador Floodlight. Además, se comprobó la configuración de Vlans y la conectividad entre Host bajo la arquitectura de la red definida por software según la figura 27.

5.1.3.1. Conectividad Red interna Quito

En la Figura 33 se comprobó conectividad entre host de la red interna Quito hacia los servidores y entre si host de la red interna, entre las estadísticas de ping se puede observar que los tiempos de conexión son mucho más rápidos en poder llegar a su destino que con el protocolo TCP, con SDN el tiempo de conexión es de 0.75 ms.

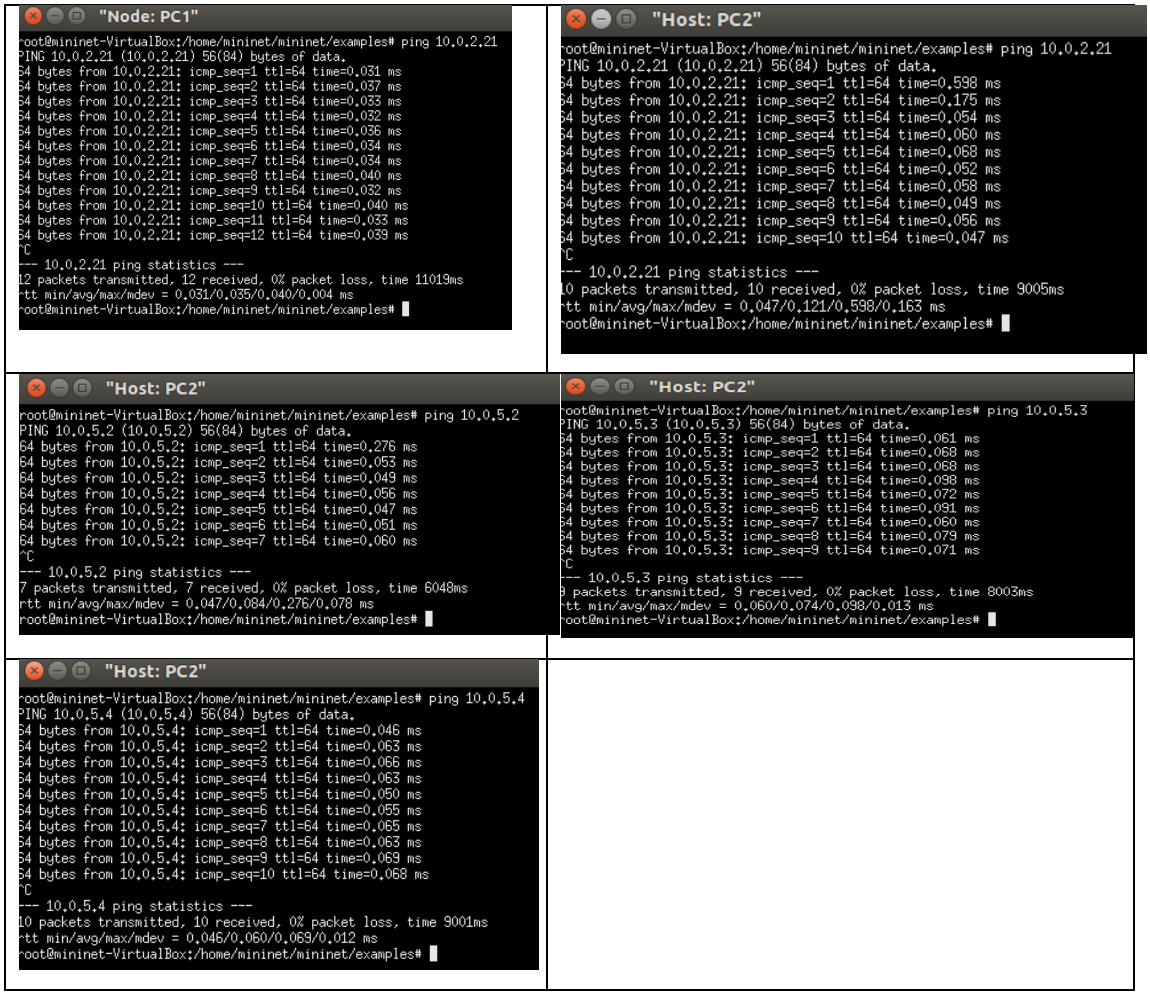


Figura 30. Conectividad red interna Quito

5.1.3.2. Conectividad Red interna Guayaquil

Como se observa en la Figura 34, la conectividad entre los hosts de la red de Guayaquil mantuvo tiempo de respuestas muy pequeñas.

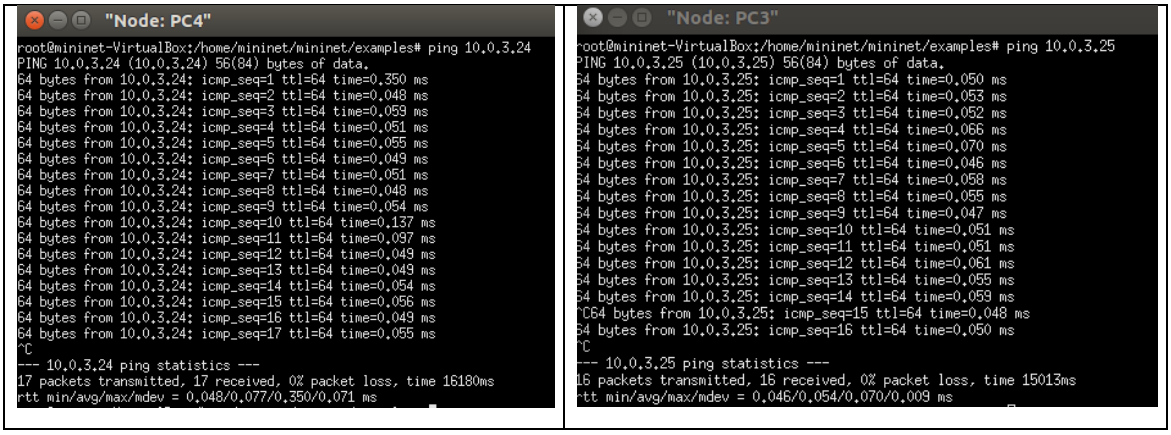


Figura 31. Conectividad red interna Guayaquil

5.1.3.3. Conectividad de Servidores

La empresa mantiene servidores tanto para Web, Mail y un servidor de respaldo, para lo cual se diseñó una única red en la cual trabajarían por mantener la protección en una red aislada. En la Figura 35. Se puede verificar la conectividad de los servidores entre sí.

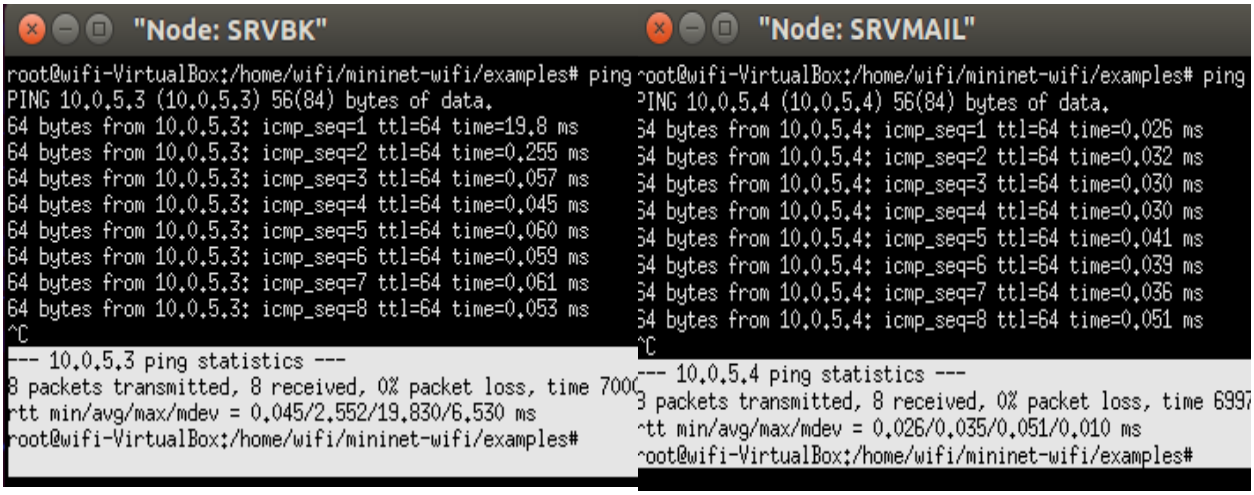


Figura 32. Conectividad de servidores

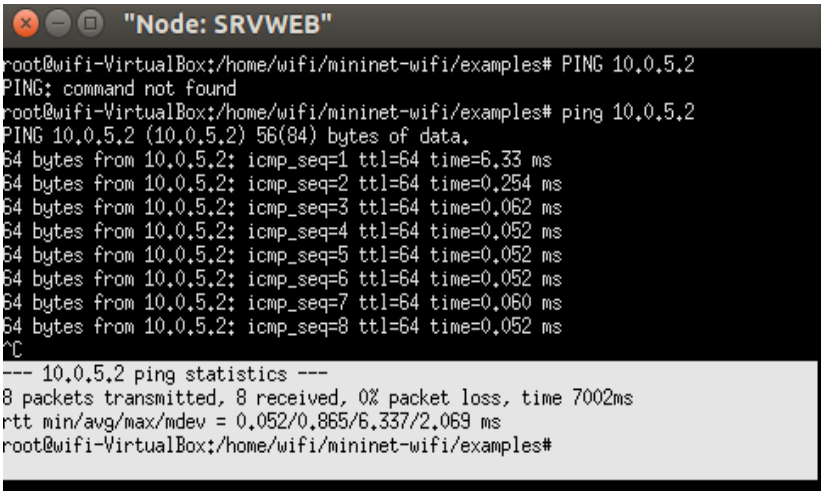


Figura 33. Conectividad de servidores

5.1.3.4. Creación de Vlans

Bajo la estructura SDN se configuró Vlans, las mismas que sirvieron para segmentar las redes y generar más escalabilidad para el crecimiento de las redes de datos, con Mininet se puede emular Vlans dependiendo la necesidad que se tenga.

Tabla 13
Vlans para configurar.

VLAN ID	DISTRIBUCIÓN	RED
VLAN 100	SERVIDORES	10.0.5.0/24
VLAN 200	ADMINISTRADORES	10.0.2.0/24
VLAN 300	VENTAS	10.0.3.0/24

5.1.3.4.1. Configuración Vlans Switch virtual Quito

En la Figura 36 se evidenció cada una de la las Vlans de Servidores y Administradores que tendrá el Switch virtual de Quito con los puertos virtuales y conexiones correspondientes. Dentro del Switch virtual Quito se puede crear las Vlans necesarias.

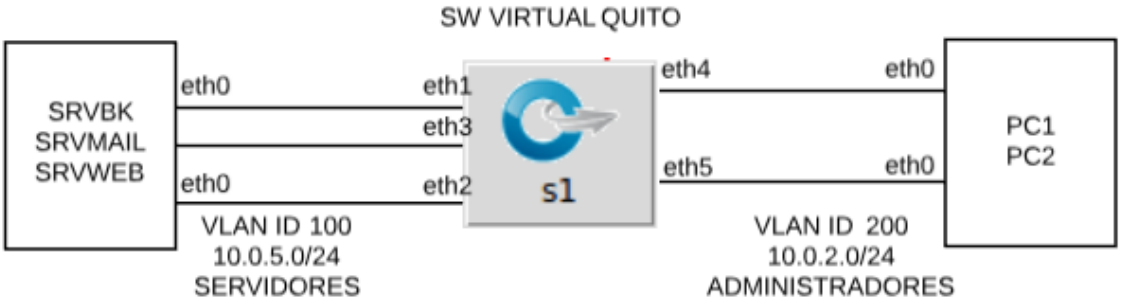


Figura 34. Switch Virtual Quito.

Se ejecutó la red SDN Mininet con la siguiente línea de comando:

root@mininer-VirtualBox:/home/mininet/mininet/examples# nano./(nombre del proyecto creado).py

Las líneas de este código crean la Vlan y añade al host en la Figura 35 se evidenció como cada uno de los hosts son parte de la Vlan creada.

mininet> **SRVBK** vconfig add SRVBK-eth0 100
mininet> **SRVWEB** vconfig add SRVWEB-eth0 100
mininet> **SRVMAIL** vconfig add SRVMAIL-eth0 100
mininet> **PC1** vconfig add PC1-eth0 200
mininet> **PC2** vconfig add PC2-eth0 200

```
mininet-wifi> SRVBK vconfig add SRVBK-eth0 100
Added VLAN with VID == 100 to IF -:SRVBK-eth0:-
mininet-wifi> SRVWEB vconfig add SRVWEB-eth0 100
Added VLAN with VID == 100 to IF -:SRVWEB-eth0:-
mininet-wifi> SRVMAIL vconfig add SRVMAIL-eth0 100
Added VLAN with VID == 100 to IF -:SRVMAIL-eth0:-
```

Figura 35. Host y Vlans creadas.

Se crearon *wildcard*, las cuales están diseñadas para filtrar direcciones de host individuales o rangos, o incluso se pueden filtrar direcciones de red para la conexión de las Vlans.

mininet> SRVBK route del –net 10.0.0.0 netmask 255.0.0.0
mininet> SRVMAIL route del –net 10.0.0.0 netmask 255.0.0.0
mininet> SRVWEB route del –net 10.0.0.0 netmask 255.0.0.0
mininet> PC1 route del –net 10.0.0.0 netmask 255.0.0.0
mininet> PC2 route del –net 10.0.0.0 netmask 255.0.0.0

Se estableció las direcciones IP con el enrutamiento así el Switch virtual.

mininet> SRVBK ifconfig SRVBK-eth0.100 10.0.5.2
mininet> SRVWEB ifconfig SRVWEB-eth0.100 10.0.5.3
mininet> SRVMAIL ifconfig SRVMAIL-eth0.100 10.0.5.4
mininet> PC1 ifconfig PC1-eth0.100 10.0.2.22
mininet> PC2 ifconfig PC2-eth0.100 10.0.2.23

En la Figura 36 se encuentra detallada la configuración de cada host y se evidencia la creación de las Vlans.

Figura 36. Detalle del Direccionamiento IP y evidencia de asignación de Vlans.

5.2.4.1.2 Configuración Vlans Switch virtual Guayaquil

En la Figura 39, se evidencia la estructura de la Vlans de Ventas, la misma que tiene restricción limitada a los servidores y funcionó como una red

independiente. Dentro del Switch virtual Guayaquil se creó las Vlans necesarias.

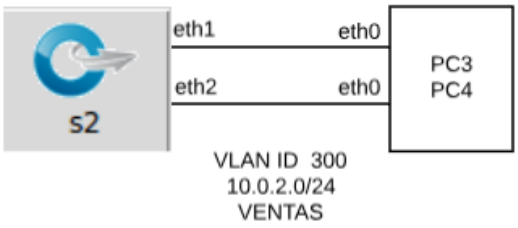


Figura 37. Vlans Ventas.

Las líneas de este código crean la Vlans y añaden al host dentro de la Vlans.

mininet> PC3 vconfig add PC3-eth0 300

mininet> PC4 vconfig add PC2-eth0 300

Se creó unas *wilcard* que están diseñadas para filtrar direcciones de host individuales o rangos e incluso pueden filtrar direcciones de red para la conexión de las Vlans.

mininet> PC3 route del –net 10.0.0.0 netmask 255.0.0.0

mininet> PC4 route del –net 10.0.0.0 netmask 255.0.0.0

Se estableció las direcciones IP con el enrutamiento hacia el Switch virtual.

mininet> PC3 ifconfig PC3-eth0.300 10.0.2.24

mininet> PC4 ifconfig PC4-eth0.300 10.0.2.25

Todo queda evidenciado en la Figura 40.

Figura 38. Hosts PC3 y PC4 añadidos a Vlans Ventas

5.1.4. Escenario de simulación de Seguridades

En este escenario se configuró la seguridad que permite al controlador Floodlight limitar accesos y permitir tráfico de un lugar a otro.

5.1.4.1. Listas de control de acceso

En la simulación se evidenció como trabajan las listas de control de acceso estándar. Se realizó el siguiente control en la red de Quito; tal como indica la Figura 39.

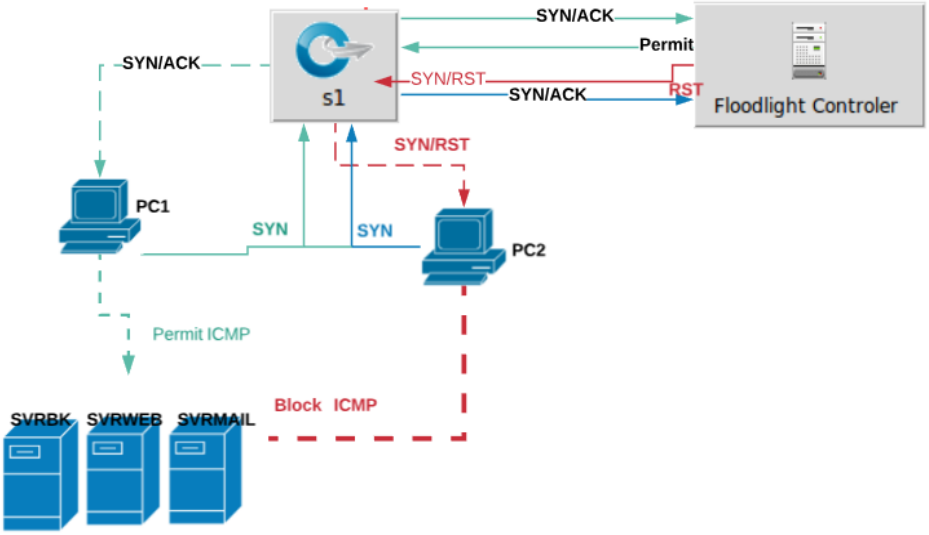


Figura 39. Diagrama de control de acceso.

Se observó que para esta configuración en Redes SDN no se requiere un Router o un Switch para lograrlo. Más bien desde la API integrada en el controlador Floodlight se realizó la programación correspondiente para controlar accesos.

En la Figura 40, se puede observar que la PC2 debe tener únicamente el bloqueo para los accesos a los servidores mientras que en las máquinas restantes se podrá acceder normalmente.

Para poder configurar las listas de control de acceso fue necesario ingresar al controlador vía terminal y se ejecutó los siguientes comandos para bloquear el ingreso de la PC2 a la granja de servidores.

Lista de control de acceso PC2 hacia SRVBK

curl -X POST -d '{"src-ip":"**10.0.2.22/32**","dst-ip":"**10.0.5.2/32**","action":"deny"}' **http://10.0.2.15:8080/wm/acl/rules/json**

curl -X POST -d '{"src-ip":"**10.0.2.22/32**","dst-ip":"**10.0.5.3/32**","action":"deny"}' **http://10.0.2.15:8080/wm/acl/rules/json**

curl -X POST -d '{"src-ip":"**10.0.2.22/32**","dst-ip":"**10.0.5.4/32**","action":"deny"}' **http://10.0.2.15:8080/wm/acl/rules/json**

La línea comando descrita anteriormente especifica la IP de origen y la IP destino. Además, la acción que tuvo es *Deny* (denegado). Luego se realizó un apuntamiento hacia el controlador. Como se indica en la Figura 40.

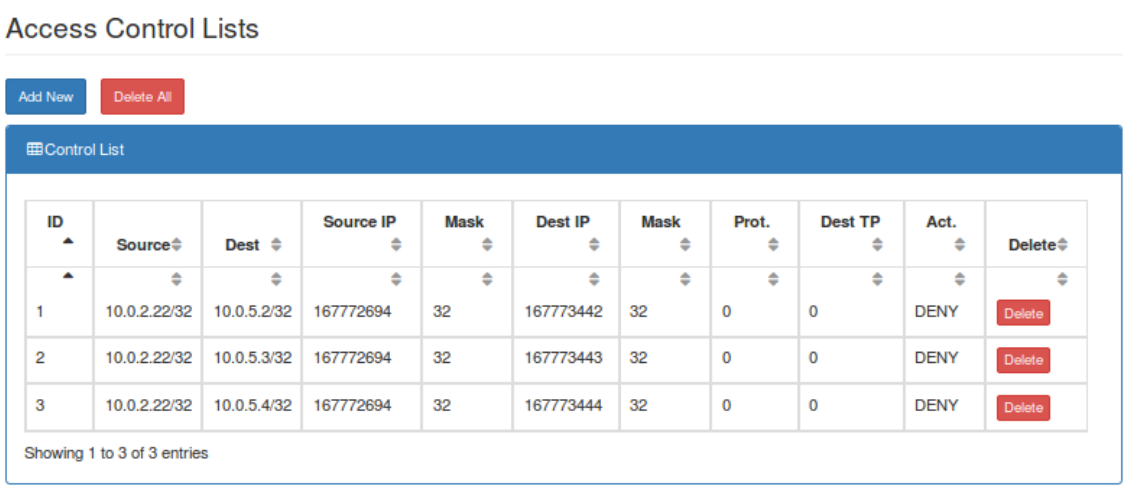


Figura 40. Listas de control de acceso configuradas.

Para comprobar el acceso al servidor de backup mediante Ping desde la PC2 se evidenció el bloqueo correspondiente a la lista de acceso Nº1; tal como se indica en la Figura 43.

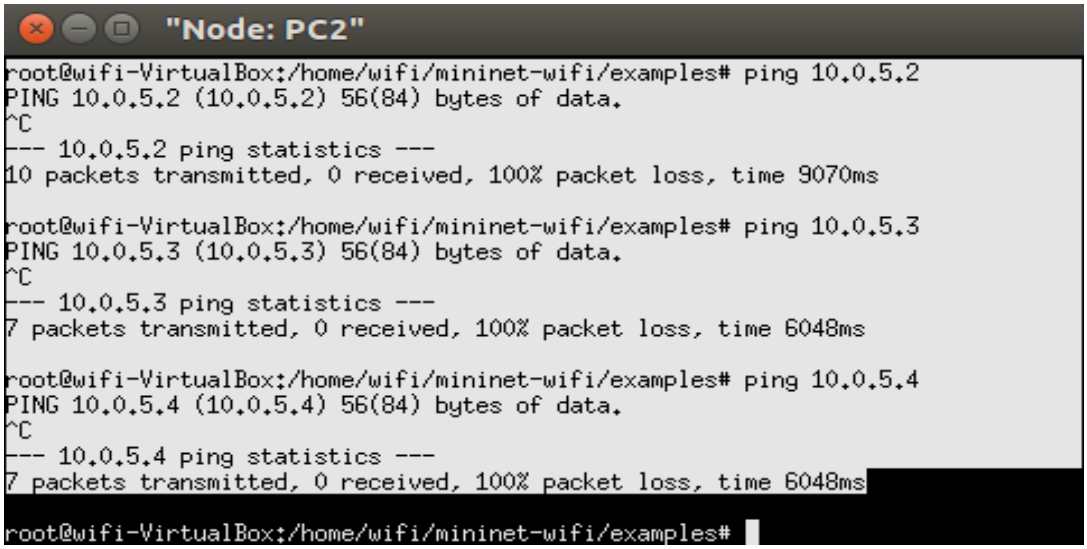


Figura 41. Evidencia Bloqueo PC2 a SRVBK

Mientras que se puede observar en la Figura 42, que la PC1 puede acceder sin problemas ya que no está aplicada ninguna regla para el bloqueo a dicha IP.

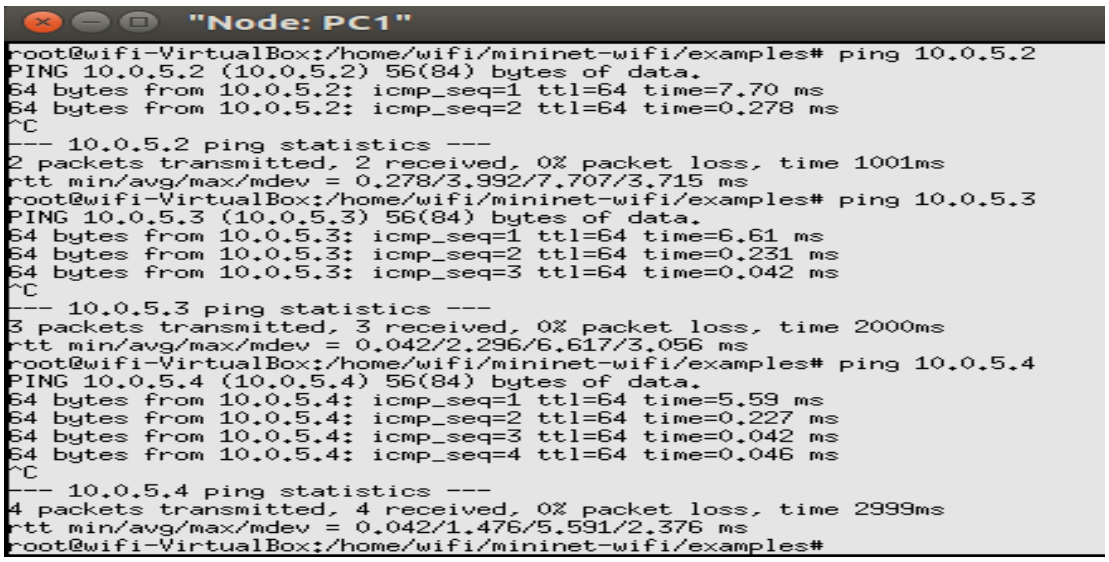


Figura 42. Evidencia acceso sin bloqueo PC1 a SRVBK

En la Figura 43 se observa el resultado del tráfico de Wireshark donde se evidenció que el paquete ICMP cuyo origen es la IP 10.0.2.22/32 y el destino la IP 10.0.2/32 es rechazado.

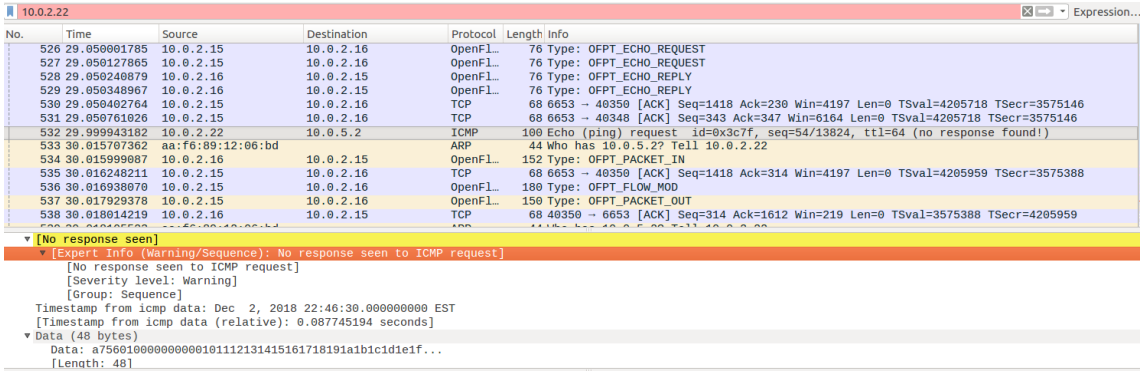


Figura 43. Resultado Wireshark tráfico bloqueado

En la Figura 44 se observa el resultado del tráfico de wireshark donde se envidencia que el paquete ICMP cuyo origen es la IP 10.0.2.22/32 y el destino la IP 10.0.5.3./32 es rechazado.

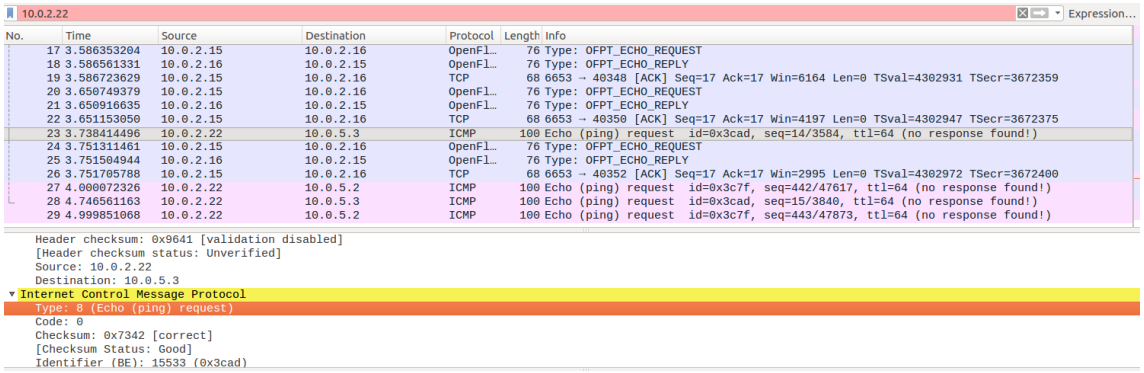


Figura 44. Resultado Wireshark tráfico bloqueado

La Figura 45 es el resultado del tráfico de wireshark donde se envidencia que el paquete ICMP cuyo origen es la IP 10.0.2.22/32 y el destino la IP 10.0.5.4/32 es rechazado.

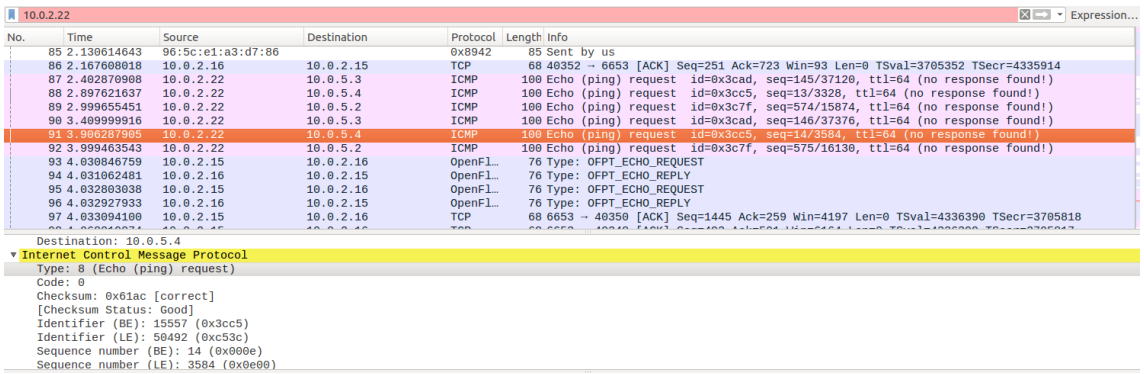


Figura 45. Resultado Wireshark tráfico bloqueado

5.1.4.2. Reglas de Firewall

Para la simulación se utilizó las entradas de firewall las mismas que se aplicararon desde la red de Guayaquil hacia la red de Quito bloqueando todo y dejando solo los puertos necesarios abiertos que en el caso de esta simulación es ICMP Y ARP.

La regla de firewall al momento de activar desde el controlador todos los paquetes tanto TCP como UDP son rechazados.

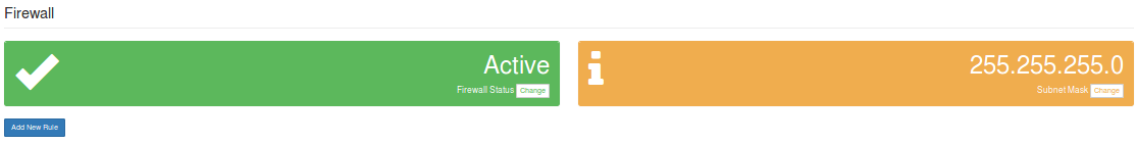


Figura 46. API Firewall Floodlight.

Para poder generar las reglas fue necesario especificar las rutas que se requieren. Se permitió el tráfico solo de la red de Guayaquil 10.0.3.0/24 hacia la red de Quito 10.0.2.0/24.
Para esto se ejecutó el siguiente codigo:

Permitimos el acceso entre redes por ARP.

curl -X POST -d '{"src-ip": "10.0.2.0/24", "dst-ip": "10.0.3.0/24", "dl-type":"ARP" }' http://10.0.2.15:8080/wm/firewall/rules/json

curl -X POST -d '{"src-ip": "10.0.3.0/24", "dst-ip": "10.0.2.0/24", "dl-type":"ARP" }' http://10.0.2.15:8080/wm/firewall/rules/json

Permirimos el acceso entre redes por ICMP

curl -X POST -d '{"src-ip": "10.0.2.0/24", "dst-ip": "10.0.3.0/24", "nw-proto":"ICMP" }' http://10.0.2.15:8080/wm/firewall/rules/json

curl -X POST -d '{"dst-ip": "10.0.2.0/24", "dst-ip": "10.0.3.0/24", "nw-proto":"ICMP" }' http://10.0.2.15:8080/wm/firewall/rules/json

En la Figura 47 y 48 se puede observar como el tráfico de Quito es aceptado por el firewall y es redirigido a su destino la red Guayaquil.

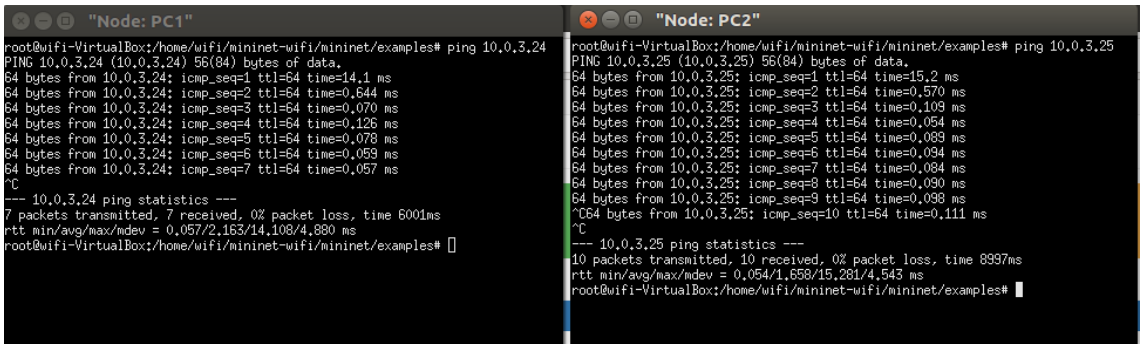


Figura 47. Tráfico de Ping desde Quito a la red de Guayaquil

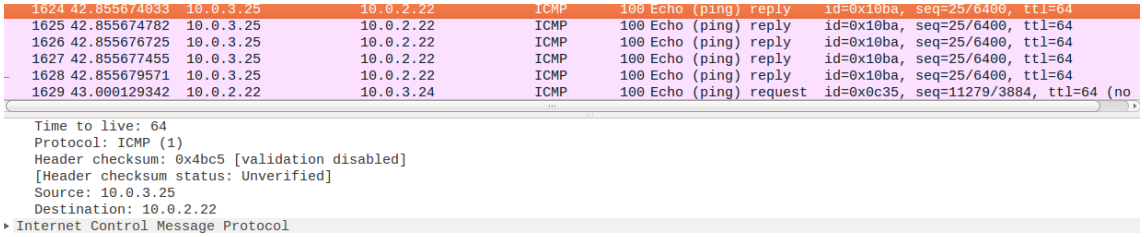


Figura 48. Wireshark tráfico permitido.

Mediante el controlador via web se verificó la creación de las reglas para permitir el tráfico de las redes de Quito y Guayaquil.

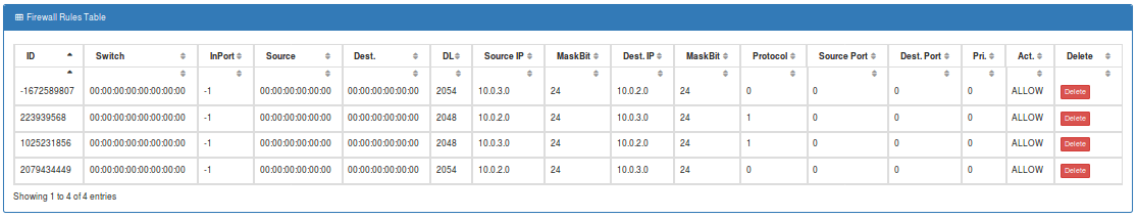


Figura 49. Listado de regla de firewall.

5.1.5. Escenario de simulación SDN-WIFI

Los desarrolladores de Mininet generaron nuevas funcionalidades, entre ellas SD-WIFI y puntos de acceso basados en controladores inalámbricos estándar y controladores para simulación basados en el protocolo de red inalámbrica 80211_hwsim.

En esta simulación se mostró como ejecutar un esenario simple de y captura de tráfico inalámbrico mediante SDN.

Miniedit en su entorno gráfico permite la creación de redes Wifi con el fin de poder emular esenarios completos de redes SDN en este caso según la Figura 50 se simuló una conexión WIFI en mininet.

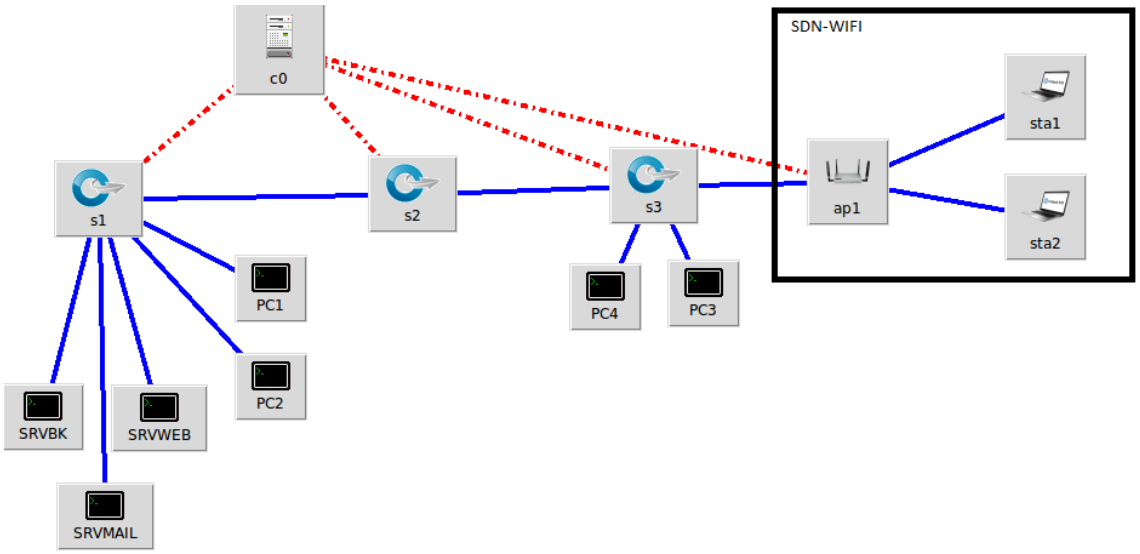


Figura 50. Diseño de la red SDN-WIFI

Para la configuracion se escogió un controlador de access point el mismo que permitió la conexión de host sta1 y sta2. En la Figura 53 se observa la configuración, donde se especifica el canal en el que trabajó la frecuencia en los estándares b, g y n que **funcionan sobre la banda de 2.4 GHz**, esta banda permanece en todos los paises universalmente y mantiene velocidades hasta de **11 Mbit/s, 54 Mbit/s, y 300 Mbit/s**.

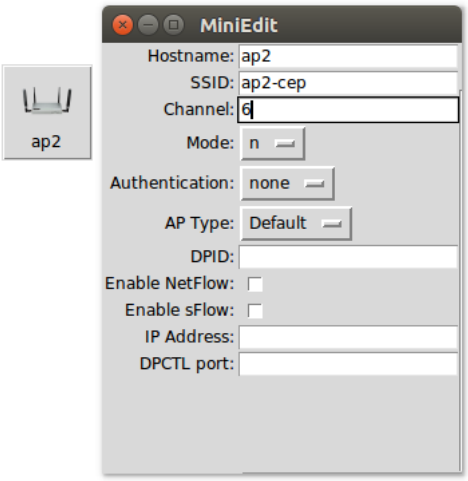


Figura 51. Configuración Miniedit wifi controlador WIFI

Una vez creado el controlador y añadido los hots como muestra la Figura 51. Se ejecutó el siguiente comando dentro de Mininet y se desplegó la red wifi creada

root@mininer-VirtualBox:/home/mininet/mininet/examples#Python ./(nombre del proyecto creado).py

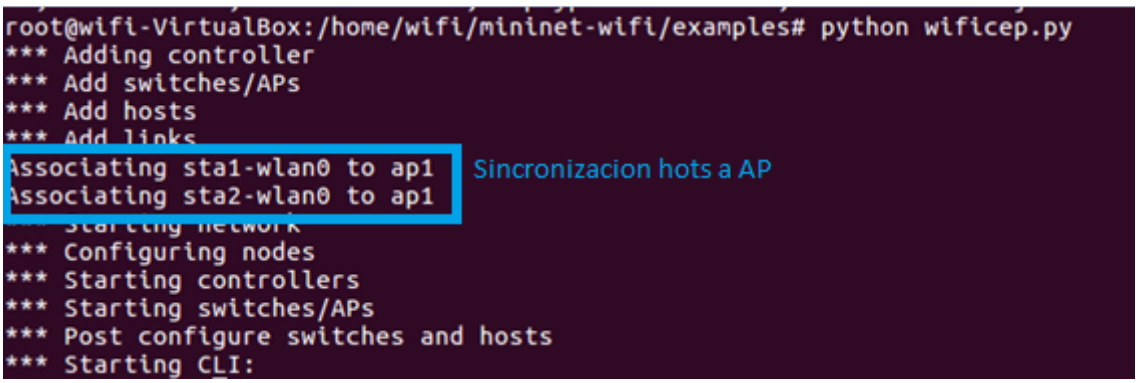


Figura 52. Asociación de los host a los *Acces Point*.

Para tener una visión más clara de la configuración generada en el lenguaje de programacion Python se ejecutó el siguiente comando:

root@mininer-VirtualBox:/home/mininet/mininet/examples# nano ./(nombre del proyecto creado).py

Se reflejo cada un de los host, link e intregaciones de la red wifi así como también las frecuencias y seguridades de dicha red con opción a poder editar.

```
info( '*** Adding controller\n' )
c0=net.addController(name='c0',
                     controller=Controller,
                     protocol='tcp',
                     port=6633)

info( '*** Add switches/APs\n')
ap1 = net.addAccessPoint('ap1', cls=OVSKernelAP, ssid='ap1-ssid', channel='6', mode='n')
s1 = net.addSwitch('s1', cls=OVSKernelSwitch)

info( '*** Add hosts\n')
h1 = net.addHost('h1', cls=Host, ip='10.0.0.1', defaultRoute=None)
sta2 = net.addStation('sta2', cls=Station, ip='10.0.0.2', defaultRoute=None)
sta1 = net.addStation('sta1', cls=Station, ip='10.0.0.1', defaultRoute=None)

net.configureWifiNodes()
info( '*** Add links\n')
net.addLink(h1, s1)
net.addLink(ap1, sta1)
net.addLink(ap1, sta2)
net.addLink(ap1, s1)

info( '*** Starting network\n')
net.build()
info( '*** Starting controllers\n')
for controller in net.controllers:
    controller.start()

info( '*** Starting switches/APs\n')
net.get('ap1').start([c0])
net.get('s1').start([c0])
```

Figura 53. Red Wifi Python

Dentro de la configuración del host sta1,sta2 se observó que existe ya la conexión hacia el controlador y también tiene asignada una dirección IP 10.0.0.1/24, 10.0.0.2/24.

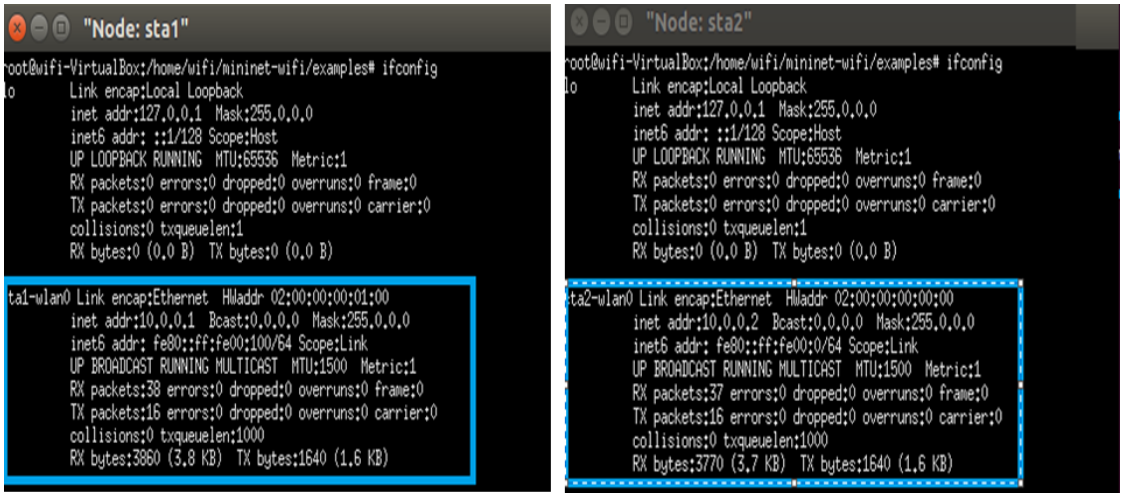


Figura 54. Integración de los hosts al controlador WIFI

En las pruebas realizas podemos tener conectividad entre los hosts conectados a la red Wifi entre sí como muestra la Figura 55.

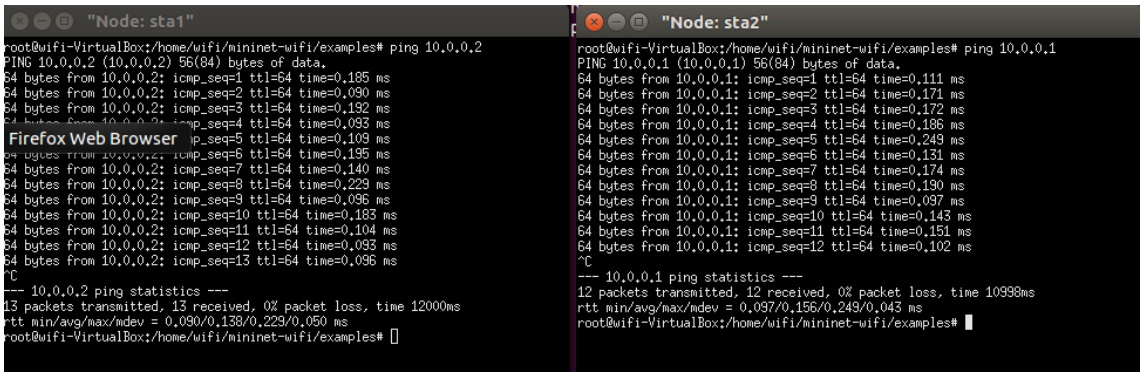


Figura 55. Conectividad de host a controlador WIFI.

Para poder observar qué hosts están conectados y estadísticas de conexión la recepción de la señal (RX) y trasmisión de la señal (TX) y el período de respuesta, se ejecutó la siguiente línea de comando:

Mininet-wifi> sta1 iw dev sta1-wlan0 link
Mininet-wifi> sta2 iw dev sta2-wlan0 link

Según la Figura 56 se puede observar que los tiempos de trasmisión y recepción son estables y se puede tener conexión sin pérdida de paquetes.

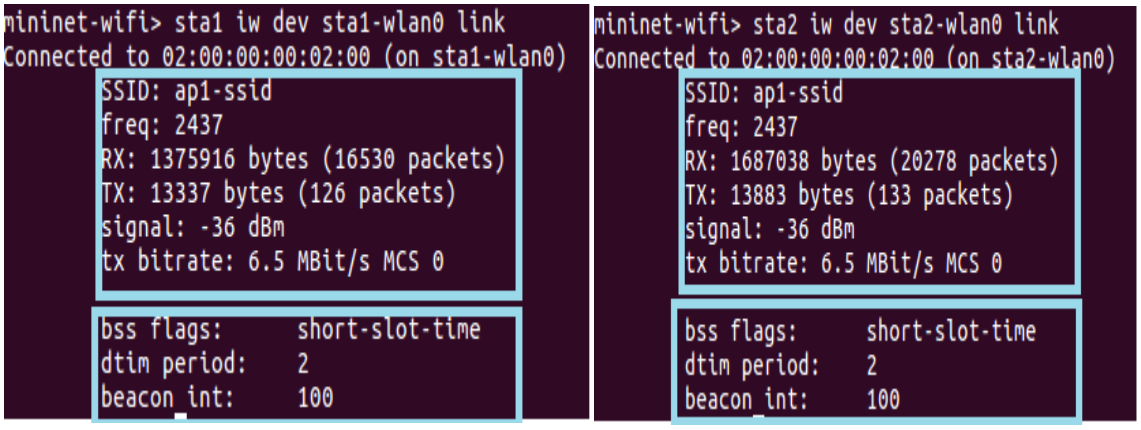


Figura 56. Tiempo de trasmisión y recepción de host conectado a WIFI.

Para poder saber el espectro que irradia la antena virtual emulada se ejecutó la siguiente línea de comando.

Mininet-wifi> sta1 iwconfig

Mininet-wifi> sta2 iwconfig

En la Figura 57 se puede observar que la frecuencia con la cual se conectó el host es de 2.437 GHZ dentro del estándar N de Wireless, además, de la velocidad 6.5 Mbps.

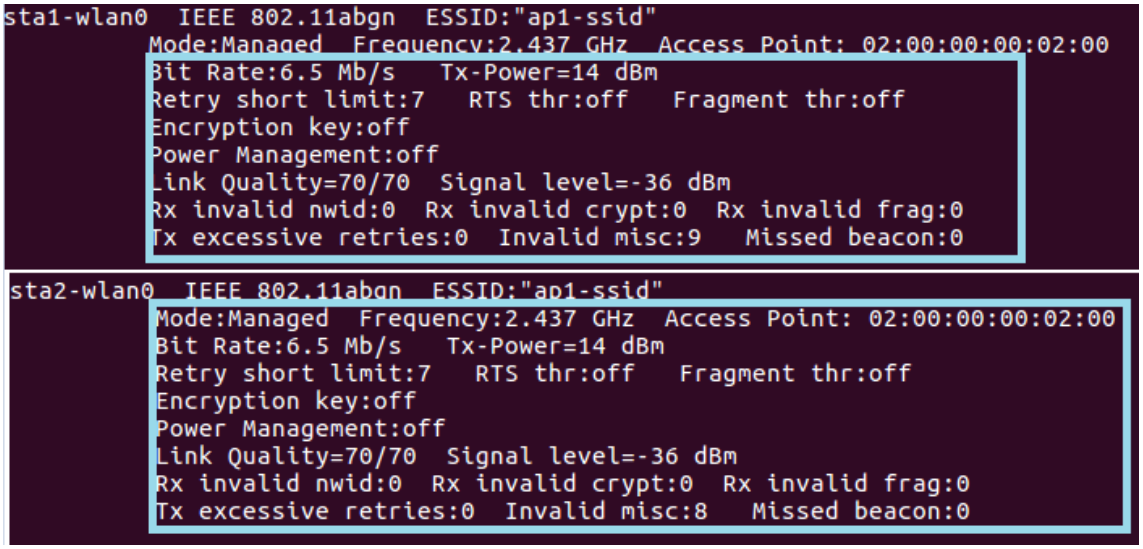


Figura 57. Estadística de conexiones WIFI.

Es posible ver como los Host pueden trasladarse de un lugar a otro o verificar en qué posición se encuentran mediante un diagrama en tiempo real proporcionada por Miniedit donde se observó la cobertura que se tiene y como el host se puede desconectar mientas se alejan de la cobertura. Se indica en la Figura 58.

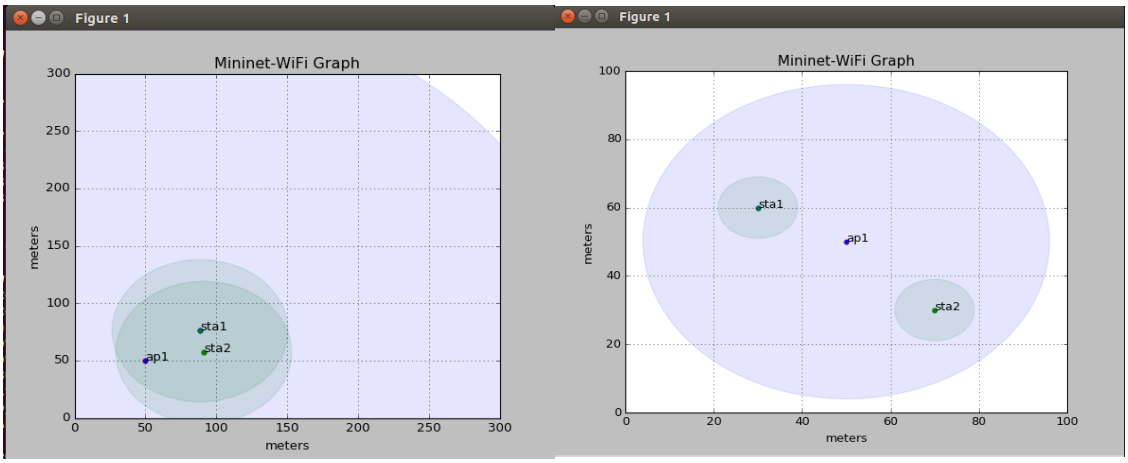


Figura 58. Cobertura que tienen los AP y como es la movilidad.

6. CAPITULO VI. EVALUACIÓN DE LA RED SDN

Este capítulo contiene un breve análisis de los resultados obtenidos en la simulación antes realizada.

De acuerdo con la arquitectura de SDN es necesario estudiar al controlador Floodlight el mismo que tuvo todo el control de la red mediante sus APP generadas con código abierto.

En la Figura 61 se observa una estadística de la red, su actividad, el número de host conectados y los links de conexión, las pruebas duraron alrededor de 37 minutos en la simulación, también podemos observar que el controlador este activo, así como también el panel de reglas.

Podemos observar que el controlador ya detecta 3 switches virtuales con 28 host (estos host es una estadística de cuantos se han conectado durante el periodo de pruebas) también existe una grafica en la cual nos indica el procesamiento del controlador por si algún momento llega a colapsar.

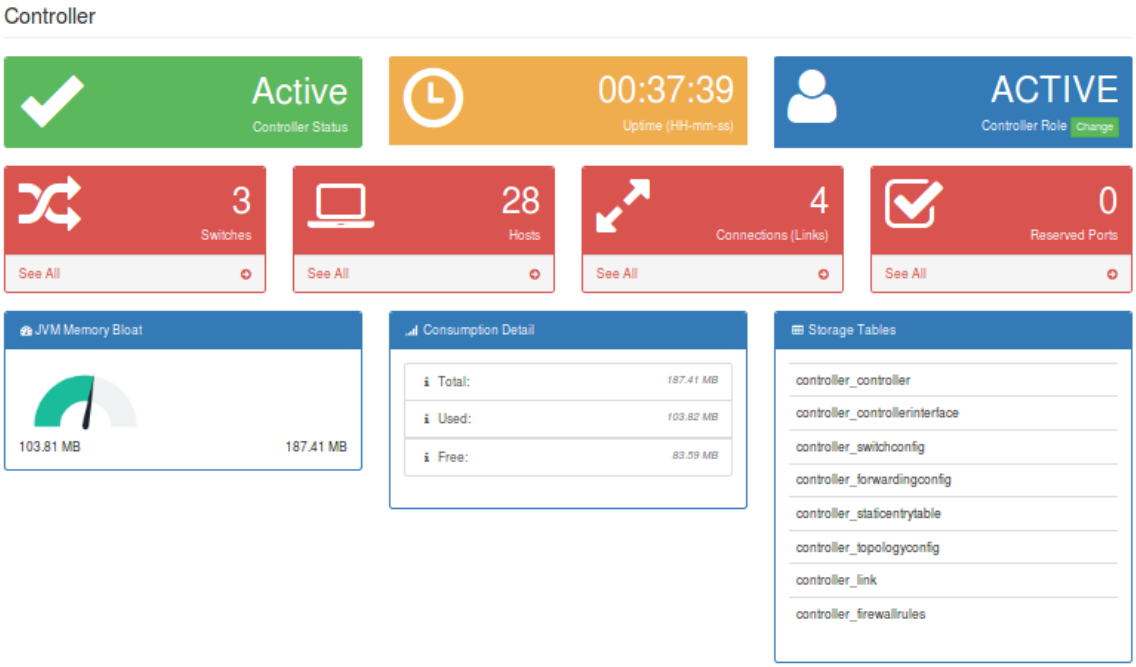


Figura 59. Controlador Floodlight

Floodlight muestra la topología creada bajo SDN-Mininet como se puede observar en la Figura 60.

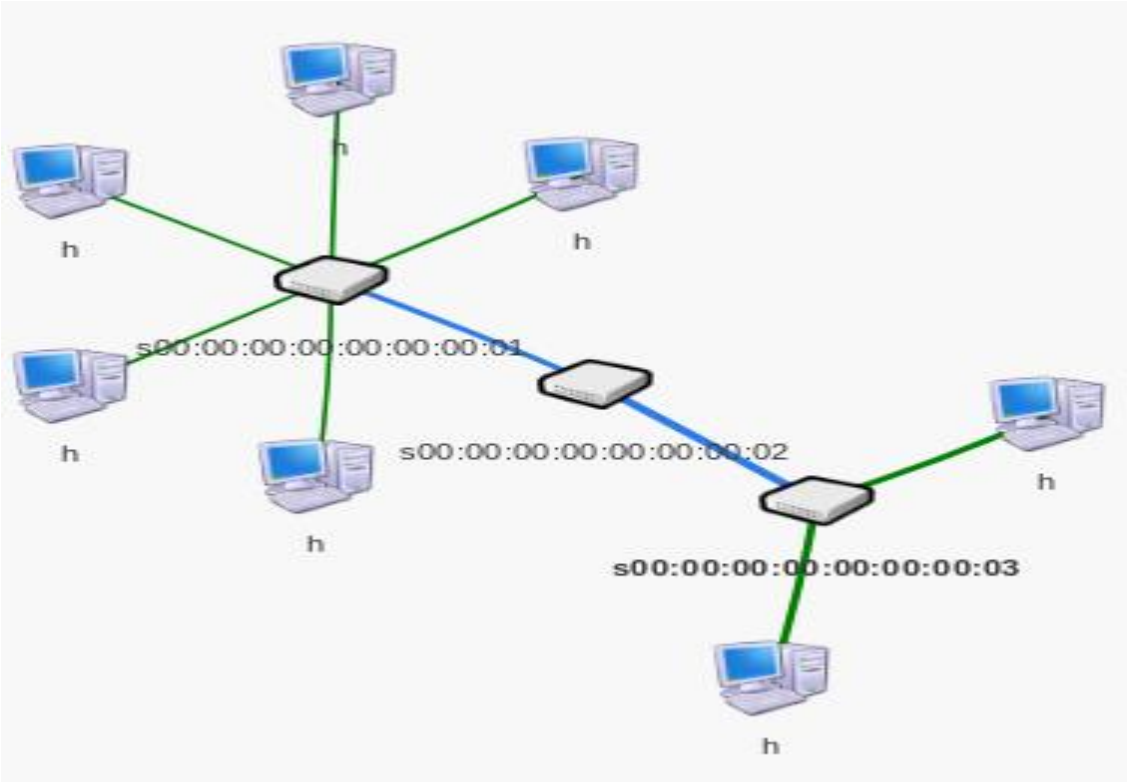


Figura 60. Topología SDN FLODLIGTH

En el controlador se estableció políticas de firewall, donde después de la configuración generada, se evidenció dichas reglas, las cuales se pueden modificar vía comandos y también eliminar, según la Figura 61.

Firewall Rules Table

ID	Switch	InPort	Source	Dest.	DL	Source IP	MaskBit	Dest. IP	MaskBit	Protocol	Source Port	Dest. Port	Pri.	Act.	Delete
-1672589807	00:00:00:00:00:00:00:00	-1	00:00:00:00:00:00	00:00:00:00:00:00	2054	10.0.3.0	24	10.0.2.0	24	0	0	0	0	ALLOW	Delete
223939568	00:00:00:00:00:00:00:00	-1	00:00:00:00:00:00	00:00:00:00:00:00	2048	10.0.2.0	24	10.0.3.0	24	1	0	0	0	ALLOW	Delete
1025231856	00:00:00:00:00:00:00:00	-1	00:00:00:00:00:00	00:00:00:00:00:00	2048	10.0.3.0	24	10.0.2.0	24	1	0	0	0	ALLOW	Delete
2079434449	00:00:00:00:00:00:00:00	-1	00:00:00:00:00:00	00:00:00:00:00:00	2054	10.0.2.0	24	10.0.3.0	24	0	0	0	0	ALLOW	Delete

Showing 1 to 4 of 4 entries

Figura 61. Políticas de Firewall.

En la figura 62 se puede observar las listas de control de acceso donde se puede evidenciar que se crean en orden ascendente según como se creó, además de puede observar que en todas las reglas tiene la acción de denegar el tráfico de una segmento a otro.

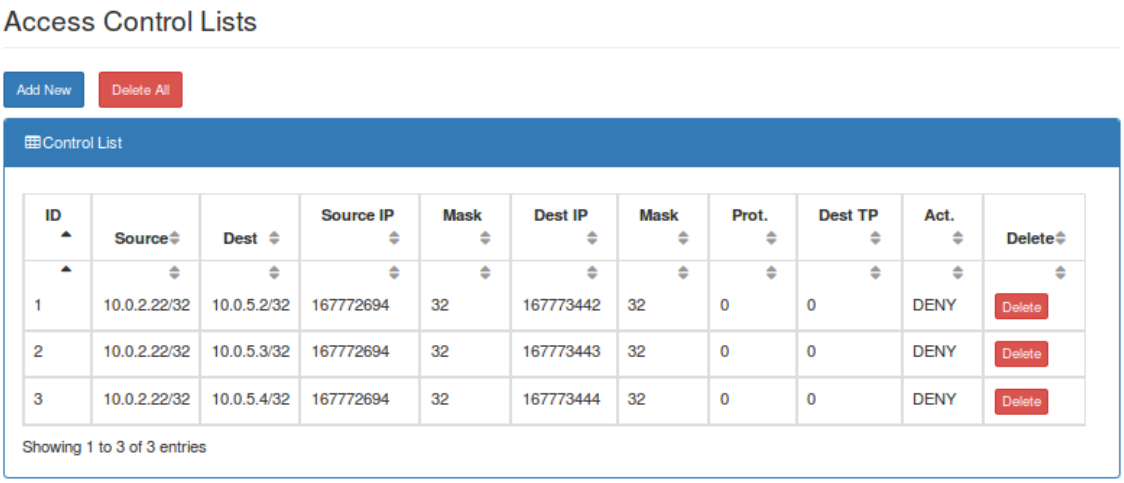


Figura 62. Access Control

Los switches son el componente principal en las redes SDN especialmente el Switch Openflow, los mismos que reciben cada uno de los paquetes y los distribuyen hacia el controlador mediante el protocolo estudiado Openflow.

Según la Figura 63 se observa el tráfico generado es de 13619 Bytes y los paquetes enviados son de 175 ms, así mismo se puede evidenciar un detalle completo del Switch como versión del protocolo y fabricante etc.

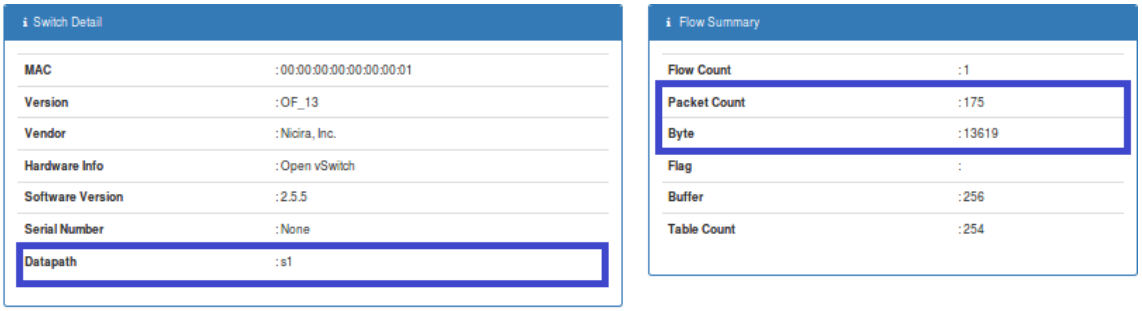


Figura 63. Detalle OpenSwitch S1.

La Figura 64 muestra el detalle del Switch OpenFlow S2, un Switch que simula ser de borde para las comunicaciones Quito-Guayaquil. Este Switch se encargó de compartir información de las 2 sucursales es por eso que sus paquetes y el ancho de banda son mayores en comparación a los otros Switches de acceso.

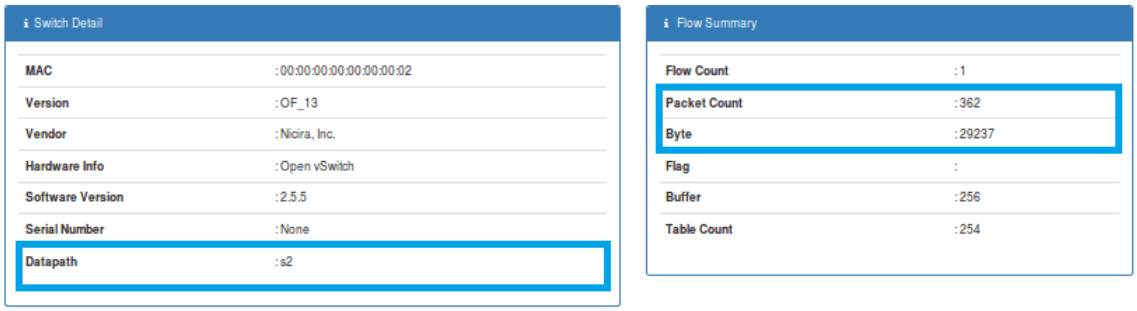


Figura 64. Detalle Openswitch S2

De acuerdo a las políticas de firewall y control de acceso la sucursal de Guayaquil tiene únicamente 1 acceso a la red de quito es por eso que el Switch que interactúa con esta sucursal el Switch S3 mantiene poco tráfico y ancho de banda generado para la comunicación de acuerdo a la Figura 65.

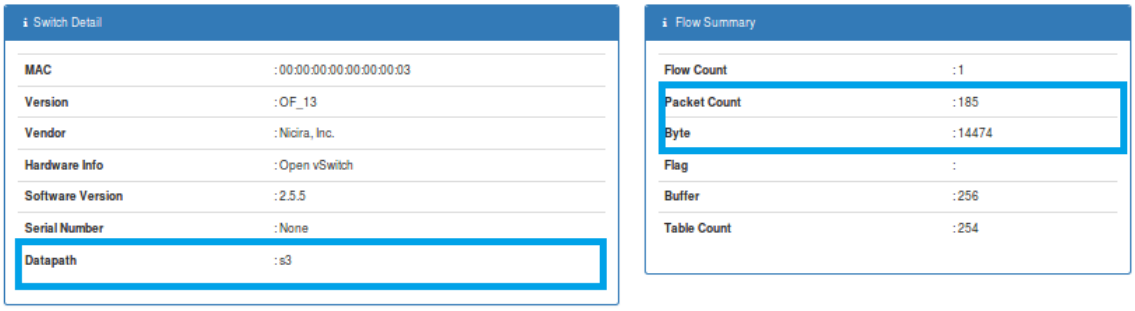


Figura 65. Detalle OpenSwitch S3

7. CAPITULO VIII. CONCLUSIONES Y RECOMENDACIONES

7.1. Conclusiones

El presente proyecto planteó una solución de una red SDN utilizando Mininet y un controlador Floodligth para esta pymes, se investigó y se estudió que las redes SDN flexibilizan toda administración y el despliegue de la red, permite gestionar de manera centralizada bajo un controlador todos los flujos de datos optimizando el tiempo de respuesta y reduciendo los costos de operación. Un controlador SDN puede administrar toda la red definida por software mediante diferentes funcionalidades que nos ofrece esta tecnología, como seguridades, routing, segmentación de red, este enfoque permite conservar la integridad de las API de código abierto permitida por el controlador SDN de la misma forma refuerza la orquestación del centro de datos la automatización y la gestión de toda la red en un mismo panel de control.

Cuando las redes sufren fallos se puede actuar de manera rápida ya que se puede aplicar redundancia del controlador y de los switches virtuales que optimizará cualquier tipo de recuperación ante un desastre.

El crecimiento actual de las compañías requiere grandes trasformaciones en las infraestructuras actuales y las redes futuras de las telecomunicaciones. Con SDN las redes son más escalables, gestionables y seguras.

Los resultados evidenciados en este trabajo evidencian las funcionalidades de SDN bajo la perspectiva de la escalabilidad y la adaptación ante el crecimiento de las redes, además en la simulación se demostró el comportamiento de las redes SDN, bajo el código abierto que nos proporciona Mininet cuyo objetivo es la creación de este tipo de redes. Con la tecnología SDN, se podrá filtrar el tráfico de red este-oeste. También, el tráfico vertical norte-sur con el fin de conocer qué tipo de protocolo circula por la Red. Además, generará políticas donde puedan descartar o redirigir el tráfico. Un beneficio como medida de seguridad será el establecer políticas de firewall y ACL.

Por otro lado, es importante recordar que SDN es una tecnología en desarrollo, sin embargo, a pesar de las grandes ofertas de los fabricantes y de la multitud de tecnologías de software libre, pensar en una migración de las redes actuales a una red SDN no deja de ser una utopía. Se puede mencionar casos de éxito como la tecnología SDN que adoptó el protocolo OpenFlow en la WAN de la empresa Google. Grandes marcas como VMware, NSX, Juniper, Fortinet ya están en la capacidad para poder instalar o diseñar este tipo de redes, sin embargo, existen desarrolladores que están creando nuevas APIs para diferentes funcionalidades como seguridades, routing, ext.

7.2. Recomendaciones

En caso de implementar está tecnología SDN en esta Pymes, para lo cual se tiene que aprovisionar el software y hardware adecuado para poder implementar la arquitectura planteada considerado número de hosts, gestión de comunicación para la red de Quito y Guayaquil, convergencia de red, así como recursos físicos que serán reemplazados por la gestión del controlador como: Firewall, Switches, etc.

Se recomienda utilizar SDN para poder filtrar el tráfico para los usuarios específicos desde y hacia el controlador SDN ya que esta funcionalidad es de gran importancia y de utilización así mismo por que podríamos ver todos los paquetes enviados y poder detectar cada uno de los protocolos que circulan en la red en tiempo real.

Para la implementación se recomienda utilizar el controlador Floodlight el mismo que según la investigación es de gran utilidad por su software libre y las APIS que se pueden integrar en las que se pueden tener varias funcionalidades que permiten más gestión sobre la red.

También se recomienda segmentar las redes mediante VLANs para poder optimizar los recursos de las redes y permitan tener mayor seguridad y crecimiento.

Para poder realizar la implementación se recomienda generar una ventana de trabajo y un plan de migración para el despliegue del controlador SDN y las APIS correspondientes.

El lenguaje de programación utilizado para este tipo de tecnología puede ser Python, las líneas de código permiten crear nuevas reglas para el control o la creación de redes, los desarrolladores pueden crear plataformas donde se puede agregar cualquier tipo de regla mediante una interfaz gráfica la cual se recomienda utilizar para cualquier configuración o despliegue de la red SDN.

Se recomienda utilizar software libre para la implementación ya que facilita la creación de aplicaciones para cada una de la gestión de la red que necesitemos, sin embargo, si existen recursos económicos trabajar con tecnologías como VMware NSX, CISCO, JUNIPER que realizan virtualización de redes bajo un esquema SDN sería lo más apropiado.

REFERENCIAS

Akram Hakiri, A. G. (2014). *Software-defined Networking: Challenges and Research Opportunities for Future.* Recuperado el 9 de septiembre de 2018 de : https://www.researchgate.net/publication/267339360_Software-Defined_Networking_Challenges_and_research_opportunities_for_Future_Internet

Bermudes, G. I (2018). *DISEÑO DE UNA RED SDN EN LA DIRECCIÓN PROVINCIAL DE PICHINCHA Quito:* Universidad de las Americas.

Ciena Experience. Outcomes. (2018). *Ciena*. Recuperdo el 03 de enero de 2019 de: https://www.ciena.com.mx/insights/what-is/What-is-Network-Functions-Virtualization_es_LA.html

Contact, M. (2016). *Diferencias entre SDN y NFV*. Recuperado el 7 de septiembre de: https://mundocontact.com/diferencias-entre-sdn-y-nfv/

Corporación de Estudios y Publicaciones. (2018). *Corporación de Estudios y Publicaciones*. Recuperado el 20 de agosto de: https://www.cep.org.ec/newceporgec/index.php?page=quienesSomos

Floodlight. (2017). *API REST de ACL (lista de control de acceso)*.Recuperado el 13 de septiembre de 2018 de: https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/4882438/ACL+Access+Control+List+REST+API

Floodlight, P. (2018). *Software de código abierto para construir redes definidas por software*. Recuperado el 13 de septiembre de 2018, de:http://www.projectfloodlight.org/

Gonzalez, C. A. *Despliegue de una maquete de Red basada en Open Flow.* Cantabria: Universidad de Cantabria.

GROUP, O. C. (2017). *¿PARA QUÉ SIRVE LA VIRTUALIZACIÓN DE REDES Y SDN (SOFTWARE DEFINED NETWORK)?* Recuperado el 13 de septimebre de 2018, de: https://www.orbit.es/virtualizacion-de-redes-y-sdn/

Jain, R. (2014). *www.cse.wustl.edu*. Recuperado el 7 de septiembre de 2018 de: https://www.cse.wustl.edu/~jain/tutorials/ftp/sd_hs14es.pdf

Kreutz D., Ramos,F.M,Verissimo P.,Rothenberg C.,Azodolmolky S., Uhlig S. (2014). Software-Defined Networking A Comprehensive Survey . Recuperado el 26 de diciembre de 2018 ,de: https://arxiv.org/pdf/1406.0440.pdf

López, L. I. (2013). propuesta de escenarios virtuales con la herramienta vnx para pruebas del protocolo openflow. Recuperado el 6 de septiembre de: http://www.dit.upm.es/~posgrado/doc/TFM/TFMs2012-2013/TFM_Lorena_Barona_2013.pdf

Muro, Y. A. (2016). *Plataforma de pruebas para evaluar el desempeño de las redes definidas por software basadas en el protocolo OPENFLOW.* Cuba: Universidad Central "Marta Abreu" de las Villas.

Norberto Figuerola. (2014). SDN Redes definidad por Software. Recuperado el 6 de septiembre del 2018, de: https://articulosit.files.wordpress.com/2013/10/sdn.pdf

opennetworking. (s.f.). *https://www.opennetworking.org/projects/mininet/*. Recuperado el 23 de octubre de 2018, de:https://www.opennetworking.org/projects/mininet/.

OpenWebinars.net. (2015). *Curso de Cloud con OpenStack*. Recuperado el 13 de septiembre de 2018, de http://iesgn.github.io/ow1/curso/u6/intro.

Pardo, C. *Implementación de un openflow controller para el manejo de OPENFLOW.* Bogotá:Universidad Politécnica Javeriana.

Punt Informátic. (2018). *Punt Informatic*. Recuperado el 03 de enero de 2018, de: https://puntinformatic.com/beneficios-de-la-red-definida-por-software-sdn/

Rouse, M. (2015). *searchsdn.techtarget.com.* Recuperado el 15 de octubre de 2018 de: https://searchsdn.techtarget.com

Sicrom Technology. (s.f.). *Sicrom*. Recuperado el 03 de enero de 2018, de https://sicrom.com/blog/las-caracteristicas-especiales-de-las-sdn/

Spera, C. (2013). *Software Defined Network.* Recuperado el 7 de septiembre del 2018 de: https://www.la.logicalis.com/globalassets/latin-america/...20/lnow20-nota- 42-45.pdf

Tejedor, R. J. (2014). *SDN: el futuro de las redes inteligentes.* Recuperado el 10 de septiembre del 2018: el futuro de las redes inteligentes: http://www.academia.edu/18932217/SDN_el_futuro_de_las_redes_inteligentes

ANEXOS

ANEXO 1. Código de creación de la red SDN en leguaje de programación Python

```
#!/usr/bin/python

from mininet.net import Mininet
from mininet.node import Controller, RemoteController, OVSController
from mininet.node import CPULimitedHost, Host, Node
from mininet.node import OVSKernelSwitch, UserSwitch
from mininet.node import IVSSwitch
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.link import TCLink, Intf
from subprocess import call
from mn_wifi.net import Mininet_wifi
from mn_wifi.node import OVSKernelAP, UserAP
from mn_wifi.cli import CLI_wifi

def myNetwork():

    net = Mininet_wifi( topo=None,
                        build=False,
                        ipBase='10.0.0.0/8')

    info( '*** Adding controller\n' )
    c0=net.addController(name='c0',
                         controller=RemoteController,
                         ip='10.0.2.15',
                         protocol='tcp',
                         port=6653)
```

```
info( '*** Add switches/APs\n')
s2 = net.addSwitch('s2', cls=OVSKernelSwitch)
s3 = net.addSwitch('s3', cls=OVSKernelSwitch)
s1 = net.addSwitch('s1', cls=OVSKernelSwitch)

info( '*** Add hosts\n')
PC1 = net.addHost('PC1', cls=Host, ip='10.0.2.21', defaultRoute=None)
SRVWEB = net.addHost('SRVWEB', cls=Host, ip='10.0.5.3', defaultRoute=None)
SRVMAIL = net.addHost('SRVMAIL', cls=Host, ip='10.0.5.4', defaultRoute=None)
PC3 = net.addHost('PC3', cls=Host, ip='10.0.3.24', defaultRoute=None)
PC2 = net.addHost('PC2', cls=Host, ip='10.0.2.22', defaultRoute=None)
PC4 = net.addHost('PC4', cls=Host, ip='10.0.3.25', defaultRoute=None)
SRVBK = net.addHost('SRVBK', cls=Host, ip='10.0.5.2', defaultRoute=None)

net.configureWifiNodes()
info( '*** Add links\n')
net.addLink(s1, SRVBK)
net.addLink(s1, SRVWEB)
net.addLink(s1, SRVMAIL)
net.addLink(s1, PC2)
net.addLink(s1, PC1)
```

```
net.addLink(s1, s3)
net.addLink(s3, s2)
net.addLink(s2, PC3)
net.addLink(s2, PC4)

info( '*** Starting network\n')
net.build()
info( '*** Starting controllers\n')
for controller in net.controllers:
    controller.start()

info( '*** Starting switches/APs\n')
net.get('s2').start([c0])
net.get('s3').start([c0])
net.get('s1').start([c0])
```

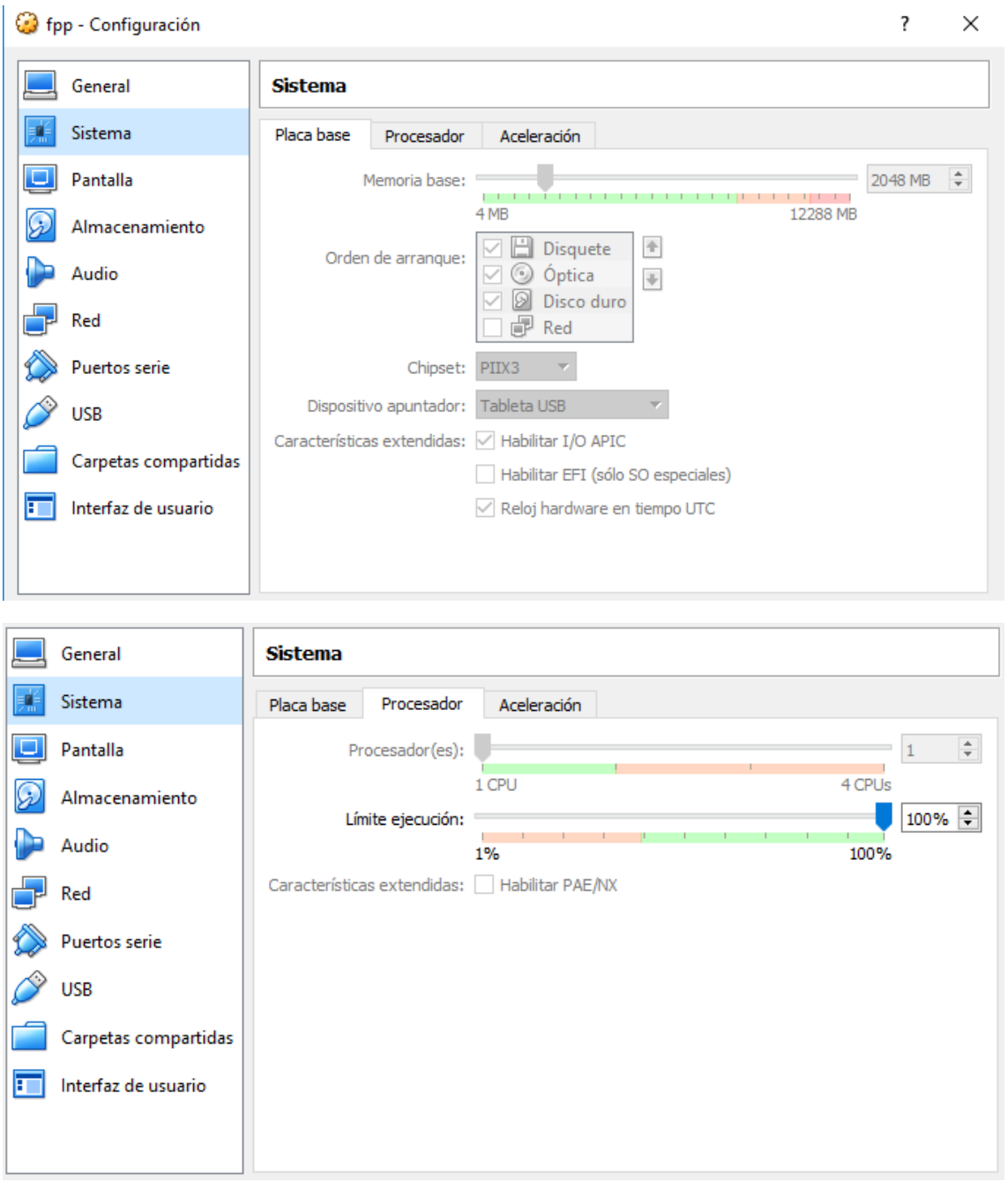
```
    net.addLink(s2, PC3)
    net.addLink(s2, PC4)

    info( '*** Starting network\n')
    net.build()
    info( '*** Starting controllers\n')
    for controller in net.controllers:
        controller.start()

    info( '*** Starting switches/APs\n')
    net.get('s2').start([c0])
    net.get('s3').start([c0])
    net.get('s1').start([c0])

    info( '*** Post configure switches and hosts\n')

    CLI_wifi(net)
    net.stop()

if __name__ == '__main__':
    setLogLevel( 'info' )
    myNetwork()
```

```
info( '*** Add switches/APs\n')
s2 = net.addSwitch('s2', cls=OVSKernelSwitch)
s3 = net.addSwitch('s3', cls=OVSKernelSwitch)
s1 = net.addSwitch('s1', cls=OVSKernelSwitch)

info( '*** Add hosts\n')
PC1 = net.addHost('PC1', cls=Host, ip='10.0.2.21', defaultRoute=None)
SRVWEB = net.addHost('SRVWEB', cls=Host, ip='10.0.5.3', defaultRoute=None)
SRVMAIL = net.addHost('SRVMAIL', cls=Host, ip='10.0.5.4', defaultRoute=None)
PC3 = net.addHost('PC3', cls=Host, ip='10.0.3.24', defaultRoute=None)
PC2 = net.addHost('PC2', cls=Host, ip='10.0.2.22', defaultRoute=None)
PC4 = net.addHost('PC4', cls=Host, ip='10.0.3.25', defaultRoute=None)
SRVBK = net.addHost('SRVBK', cls=Host, ip='10.0.5.2', defaultRoute=None)

net.configureWifiNodes()
info( '*** Add links\n')
net.addLink(s1, SRVBK)
net.addLink(s1, SRVWEB)
net.addLink(s1, SRVMAIL)
net.addLink(s1, PC2)
net.addLink(s1, PC1)
```

```
net.addLink(s1, s3)
net.addLink(s3, s2)
net.addLink(s2, PC3)
net.addLink(s2, PC4)

info( '*** Starting network\n')
net.build()
info( '*** Starting controllers\n')
for controller in net.controllers:
    controller.start()

info( '*** Starting switches/APs\n')
net.get('s2').start([c0])
net.get('s3').start([c0])
net.get('s1').start([c0])
```

```
    net.addLink(s2, PC3)
    net.addLink(s2, PC4)

    info( '*** Starting network\n')
    net.build()
    info( '*** Starting controllers\n')
    for controller in net.controllers:
        controller.start()

    info( '*** Starting switches/APs\n')
    net.get('s2').start([c0])
    net.get('s3').start([c0])
    net.get('s1').start([c0])

    info( '*** Post configure switches and hosts\n')

    CLI_wifi(net)
    net.stop()

if __name__ == '__main__':
    setLogLevel( 'info' )
    myNetwork()
```

Anexo 2.- Recursos de máquina virtual para la simulación del controlador



Anexo 3.- Recursos de máquina virtual para la simulación de Mininet