PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

DISEÑO DE UNA RED WAN BASADA EN SDN PARA UNA INSTITUCIÓN CON VARIAS SEDES

Tesis para obtener el titulo profesional de Ingeniero Electrónico

ALEXIS PIERRE ESTELA LOZANO

ASESOR: M.Sc.Ing. Gumercindo Bartra Gardini

Lima, agosto de 2022

RESUMEN

El documento de tesis se divide en 4 capítulos que se detallan a continuación:

En el primer capítulo, se realiza la presentación de la red a la cual se realizará el cambio de tecnología en un entorno simulado el cual es la red de la PUCP, se presenta las problemáticas que esta posee en el despliegue actual y cuál fue la evolución de la automatización de las redes hasta llegar a la tecnología SDN.

En el segundo capítulo, se presentan las definiciones sobre la tecnología SDN, como los componentes junto con el funcionamiento de cada uno de ellos, comunicación entre planos, modelos de despliegue y finalmente las áreas de aplicaciones de la tecnología.

En el tercer capítulo, se detallan los requerimientos generales del despliegue y requerimientos de cada plano: datos y control. Luego se realiza la elección de cada componente para la simulación: el controlador, *switch* SDN y entorno de simulación para la red SDN.

Finalmente, en el cuarto capítulo, se muestran las pruebas de los servicios que posee la red PUCP, se obtuvieron estadísticas de telemetría en cada modelo para realizar el análisis y comparación; y el despliegue del nuevo servicio *Video Streaming* en la red SD-WAN PUCP.

DEDICATORIA

A mis padres
A mi asesor
Y principalmente a mi abuelito Celestino

AGRADECIMIENTOS

A Dios, por permitirme terminar esta tesis. A mis padres, por haberme guiado en todo momento de mis 5 años de carrera universitaria. A mi familia, por haberme ayudado en todo momento, aunque no tengan mucha idea de lo que hago. A mis amigos, por todas las vivencias que hemos pasado dentro y fuera de la universidad. A mi asesor, M.Sc. Ing Gumercindo Bartra Gardini, por todo el apoyo brindado durante todo este proceso. Finalmente, un agradecimiento muy especial a mi abuelito Celestino "Papa Tino" que hasta el último momento de su vida estuvo completamente orgulloso de mi, por sus consejos y apoyo incondicional. ¡Este es tu logro Abuelito!

ALEXIS ESTELA LOZANO

ÍNDICE

ÍNDICE DE FIGURA

INDICE DE TABLAS

INTRODUCCIÓN

Las redes actuales que poseen gran tamaño tienen una cantidad considerable de usuarios conectados en diferentes sedes con una única sede central que posee la mayoría de los servicios que se usan en la red como servidor de correos, telefonía, etc. Este es el caso de la red PUCP al concentrar todos los servicios en un punto genera que realicen solicitudes de todo tipo de tráfico desde diferentes orígenes. Estas solicitudes pueden crear cuellos de botellas, congestión de los enlaces, una interrupción de la calidad de servicio brindada a los usuarios. Por lo cual supone un trabajo más complicado para los administradores de red de mantener conectividad, estabilidad y el soporte a la red.

Por estos motivos, las redes WAN deben ser un punto fundamental en los procesos de automatización de las redes, ya que esta red permite las conexiones entre sedes y conectividad con servicios que se ofrecen a los usuarios o servicios en la nube. La tecnología SDN permite tener un control completo de la red centralizado en un elemento llamado controlador que tiene una visibilidad completa de los dispositivos conectados a esta. Los administradores de red podrán realizar los procesos de mantenimiento, soporte y *troubleshooting* de una forma más eficaz en poco tiempo.

El objetivo de la tesis es el diseñar una red que use la tecnología SD-WAN tomando como base los servicios que actualmente brinda la red PUCP. Con el fin de realizar comparaciones de estadísticas de telemetría en ambos modelos para mostrar la efectividad de la nueva tecnología SDN sobre las redes WAN.

1. PROBLEMÁTICA DE LA RED PUCP

Este capítulo se centra en la evolución de las redes hacia el surgimiento de la tecnología SDN para el caso de la red de la Pontificia Universidad Católica del Perú junto con sus interconexiones de sus oficinas externas (WAN). Además, se explica cuáles son las problemáticas de las redes actuales y cuáles son los beneficios que brinda el uso de una arquitectura SDN en las redes LAN y las redes WAN.

1.1 Red PUCP actual

1.1.1. Red LAN

En la actualidad, la red LAN de la PUCP es usada por una gran cantidad de dispositivos entre propios de la universidad y por parte de los alumnos, que constantemente usan servicios de la red como el servicio de correos, plataforma del Campus Virtual, la plataforma PAIDEIA que por la temporada del presente año se usan para el desarrollo de las clases en forma virtual, etc. El despliegue de esta red se rige bajo la topología común para campus: capa de acceso, capa de distribución y capa core como se puede apreciar en la figura 1.

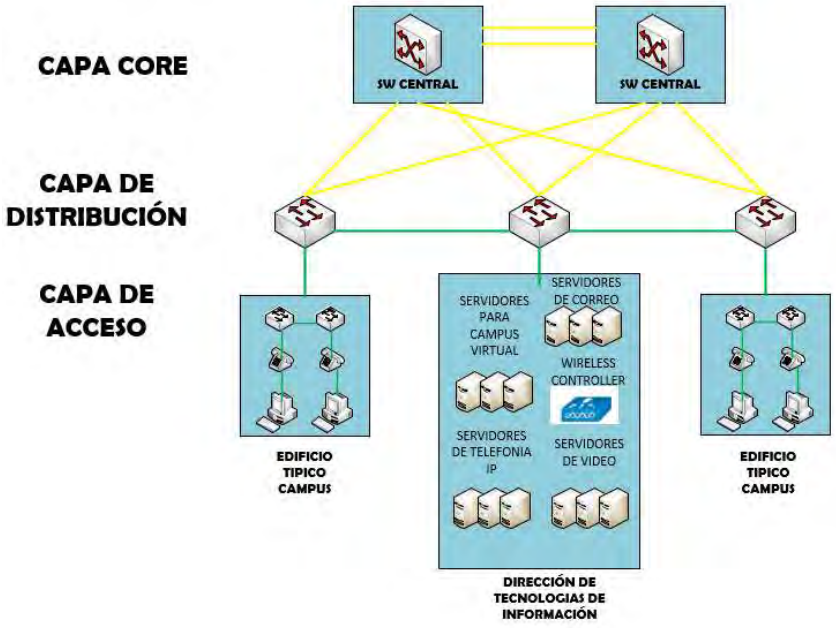

Figura 1: Estructura de la red PUCP
Fuente: Elaboración propia (2020)

1.1.1.1 Capa Core

La red está conformada por dos *Switches* capa 3 Cisco Catalyst 6500 que cumplen la función de core [1]. Estos equipos son de alto rendimiento para redes empresariales, ya que poseen un ancho de banda 32 Gbps que se puede ampliar hasta 720 Gbps; y permiten el uso de aplicaciones multimedia y voz. La ventaja de estos equipos es la cantidad de puertos que varía de 48 a 576 puertos, debido a que es modular, es decir posee ranuras para diferentes configuraciones: módulo de servicios de *Firewall*, servicios VPN *Ipsec*, Servicios SSL [2].

1.1.1.2 Conexiones superiores a la capa core

El tráfico debe pasar por un *firewall* que cumple la función de defender la red interna PUCP de ataques. Luego para las conexiones a internet se usa fibra óptica con un ancho de banda de 5Gbps que van conectados al *router* del *Service providers* que es la compañía que brinda la conexión a Internet [3]. Por otro lado, también se encuentran otros enlaces que van dirigidos a los edificios externos para las conexiones de redes WAN, se definirán los tipos de enlace en líneas adelante.

1.1.1.3 Capa de distribución

Esta capa consta de *switches* que se interconectan a los *switches* de la capa de acceso, que se encuentran en todos los pisos de los edificios, y los *switches* de la capa core. Los *switches* de la capa de distribución son de modelo Catalyst 2900, de 24 y 48 puertos y cumplen la función de realizar enrutamiento para la publicación de redes a diferentes equipos con el uso de protocolos de enrutamiento como OSPF, RIP, EIGRP, etc [4]. Las conexiones de estos equipos con la capa Core se hace mediante dos tipos de cableado de diferente velocidad de intercambio de información: fibra óptica, que puede ser de 1Gbps, 10Gbps y 40Gbps; y cobre, que son de 10Mbps y 100Mbps. Asimismo, en la universidad estas conexiones están distribuidas por todo el campus y se dividen aproximadamente en 30 Km de fibra óptica y 504 Km de cobre como podemos observar en la Fig. 2.

Figura 2: Despliegue del cableado de fibra óptica en el campus
Fuente: Elaboración propia (2020)

1.1.1.4 Red inalámbrica en el campus

Adicionalmente, la red PUCP posee 1189 *access point* de modelo Cisco AIRONET de la serie 3600 colocados en diferentes lugares en todo el campus que permiten los servicios inalámbricos a los dispositivos móviles, laptops, etc. Estos dispositivos son puntos que están conectados al *router* de salida a internet de forma inalámbrica para que se pueda ampliar la conectividad a internet a parte del área de la Universidad con una velocidad máxima de hasta 1Gbps. Los dispositivos son manejados por el *Wireless Lan Controller* que brinda la autenticación de usuarios inalámbricos. Todos los *Access point* deben estar registrados en este controlador, ya que cuando algún usuario se intente conectar a la red Pucp este tráfico que se origina debe enviarse al WLC y este derogar si permite el acceso o no [5].

1.1.1.5 Capa de Acceso

En esta capa, se encuentran *switches* de 24 o 48 puertos cuya funcionalidad es realizar tráfico de capa 2 y se utiliza para la conexión de la conexión de equipos de usuarios finales como computadoras, *Access points*, etc.

1.1.2. Red WAN

La red WAN (*Wide Area Network*) se refiere a las redes de un gran tamaño de área geográfica, en el caso de la PUCP las redes WAN son las redes que se interconectan con la red central (campus): idiomas católica Pueblo libre, Plaza San Miguel, escuela de música, etc.

Para las conexiones con los edificios externos al campus, se usan dos tipos de conexiones: IP VPN (*Virtual Private Network*) y conexión inalámbrica.

El primer tipo de conexión es a través de una IP VPN (*Virtual Private Network*), una plataforma de comunicación que redirige el tráfico al edificio que se necesite. Una VPN es la forma de conexión entre un usuario y la red a la que se quiere conectar, utiliza la red pública (INTERNET) a través de túneles virtuales que brindan seguridad al usuario y a la red, ya que no usa la dirección IP propia, sino que todos los paquetes se encapsulan en otro paquete con IP origen y destino diferente (falta aumentar que no son conexiones p2p) [6]. La principal característica de este tipo de conexión es la seguridad, ya que todo el tráfico que se envía o se reciben se encuentran cifrados así en caso de que algún intruso no pueda acceder a la información. La velocidad de conexión del campus a esta plataforma es de 22 Mbps. La velocidad de envió a los edificios externos es de 2, 3 o 4 Mbps como en los edificios Idiomas San Isidro, escuela de música.

El segundo tipo de conexión es a través de conexión inalámbrica con velocidad de 54Mbps, este tipo de conexión solo es usado por 2 edificios externos: el centro comercial Plaza san miguel y el edificio de Idiomas católica de Pueblo libre. El enlace es de tipo *Point*

to point (P2P), es decir son enlaces dedicados únicamente para la comunicación entre los nodos de origen y destino.

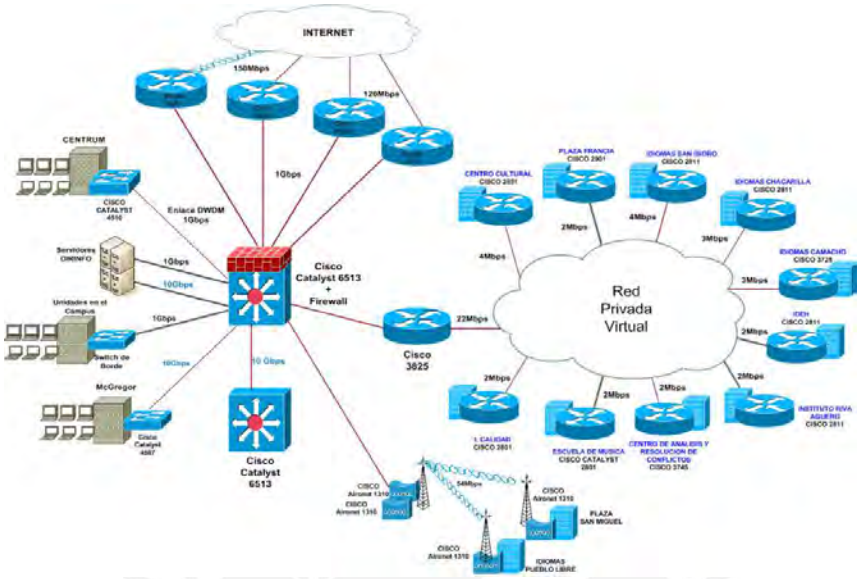

Figura 3: Velocidades de las conexiones WAN de la red PUCP [7]

1.2. Problemática de la red PUCP

La red actual de la PUCP básicamente se conforma por equipos de conmutación como *routers y switches* que se encuentran conectados entre sí bajo una arquitectura basada en el tráfico solicitado por los usuarios, la cual crea conexiones de comunicación hacia diferentes servidores. Según la literatura sobre evolución de las redes hacia SDN, se han creado limitaciones en el desarrollo de las redes y problemas para los usuarios [8] [9] [10].

1.2.1. Incremento de tráfico de datos

Este problema es generado por dos motivos: i) el aumento masivo de dispositivos móviles en las redes actuales, pues el ancho de banda de las arquitecturas actuales no es suficiente para permitir él envió de volúmenes de tráfico grandes por los enlaces donde se crean cuellos de botella o por que los enlaces no soportan dicha cantidad; ii) el incremento

exponencial de contenidos multimedia, este tipo de servicios requiere de trafico de audio y video. Ambos tráficos requieren de un gran ancho de banda y con el aumento de cantidad de aplicaciones de este tipo requerirán un mayor ancho de banda junto con un cambio en la calidad de servicio (QoS) de los dispositivos para la división y criterios de priorización de tipo de tráficos [8].

1.2.2. Poca flexibilidad de la red

Por la arquitectura que las redes actualmente poseen surge una problemática para los usuarios. La red no permite el despliegue de nuevos servicios por la necesidad de mayor ancho de banda, mejor tipo de conexiones entre los dispositivos, etc. Como por ejemplo el tráfico de videoconferencia en alta definición [9].

1.2.3. Gestión compleja de los dispositivos

La gestión de las redes actuales se logra por la configuración del plano de control de la red que se encuentra distribuido en todos los dispositivos de red (*switches, routers*). Esta descentralización de control genera dos problemáticas en el mantenimiento y las operaciones en la red [10]:

i) La configuración manual de los equipos, en caso de que se requiera realizar la modificación o agregar nuevos equipos a la red, es necesario que el personal de TI se acerque al lugar y realice las configuraciones correspondientes. Este trabajo no se puede realizar de forma remota, ya que al ser el plano de control descentralizado se deben modificar la configuración de los equipos que tienen importancia con el nuevo equipo o nueva configuración.

ii) La respuesta lenta de la red ante cambios. Esto es generado por que al ser necesario reconfigurar todos los equipos cuando se realizan cambios en la red, se ralentizando el aprovisionamiento.

1.2.4. Dependencia de *hardware* y *software* propietario

Existen muchas empresas que ofrecen equipos de comunicación como por ejemplo Cisco, Huawei, Fortinet, etc. Pero cuando se requiere desplegar una arquitectura es mucho más eficiente, en las redes actuales, realizarlo con una única compañía, ya que existe una mejor interconexión entre equipos por el uso de los mismos protocolos. Por ejemplo, en el caso de Cisco existe el protocolo CDP (*Cisco Discovery Protocol*) para el descubrimiento de equipos en una red , exclusivamente de esta compañía y esto crea dependencia , lo cual no permite cierta flexibilidad para usar protocolos de otras empresas que puede brindar mejores beneficios a la red [9].

1.3. Estado del arte

1.3.1. Antecedentes de la tecnología SDN

A lo largo de los últimos años, las redes de computadoras han estado en proceso de evolución por las necesidades de los usuarios que han surgido de despliegues de nuevos servicios. Antes de la existencia de la tecnología SDN, ya se percibía conceptos sobre programabilidad en partes de la red como es el caso de *SoftNet*, una red experimental que introdujo la programabilidad en el contenido de los paquetes como comandos que se ejecutaban en el momento que se recibían. Esta fue una primera idea para automatizar y tener un control en tiempo real en pequeña escala de las redes [11].

Luego empezaron a surgir nuevas alternativas de soluciones para controlar la red, aparecieron las Redes Activas. En este método la principal idea era de permitir la programabilidad en los nodos intermedios, es decir conexiones a los *routers o switches*. Fue un gran avance en el control de las redes, ya que los nodos aun con la principal función de realizar el encaminamiento de tráfico a sus destinos realizaban procesamiento de la data que este tráfico contenía. Esta idea fue impulsada en ese momento por la creación de nuevos lenguajes de programación de alto nivel como Java. Este cambio en las redes brindo dos

principales beneficios: i) Aceleración en la evolución de las redes, ya que el tiempo requerido para el despliegue de nuevas aplicaciones se disminuía y la creación de nuevos protocolos sin la necesidad de la estandarización que se requería por parte de la IETF (*Internet Engineering Task Force*). ii) Facilitaba la experimentación de nuevos protocolos para los investigadores, ya que era una plataforma donde se podía experimentar en tiempo real estos protocolos [12].

Luego del surgimiento de las Redes Activas, se mantuvo la posición de la búsqueda de los métodos para la programabilidad de las redes permitiendo mecanismos para la ejecución de comandos en los tráficos, pero con dos importantes bases fundadas por las redes activas: programabilidad de la red y virtualización de las redes [13].

1.3.2. Hacia la tecnología SDN

En el transcurso de la evolución de las redes, se puede observar en la figura 4, la tendencia fue crear nuevas funciones para el plano de datos como la función de *forwarding*, *routing*, etc. Dejando el plano de control a la parte del software del dispositivo.

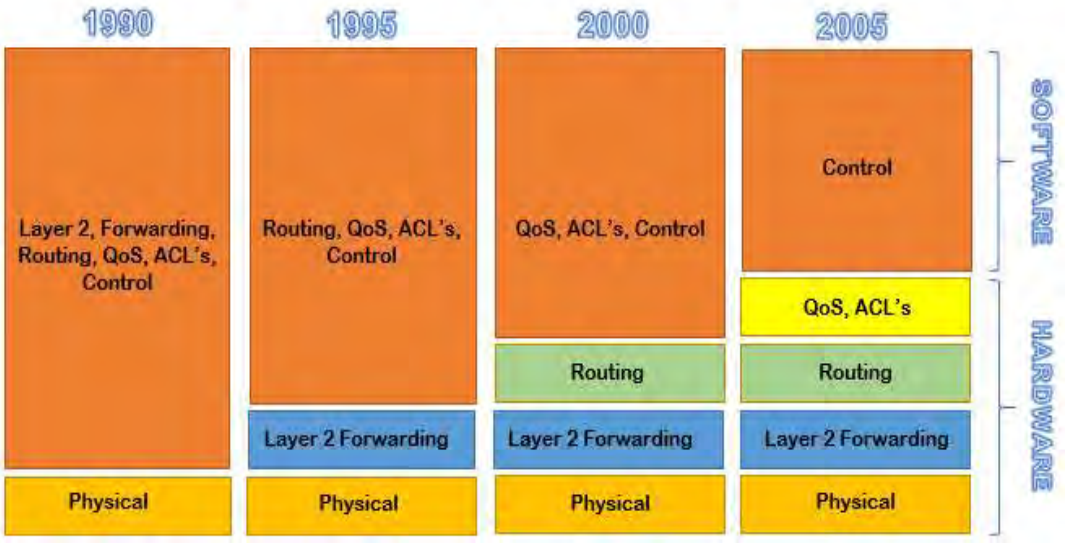

Figura 4: Evolución de las funcionalidades de los equipos de red [9]

Las redes actuales poseen estos planos dentro del dispositivo de red, por lo tanto, los equipos tienen la funcionalidad de realizar sus propias políticas de enrutamiento, *Access list* y luego de encontrar la ruta por la cual se debe enviar dicho tráfico, el plano de datos lo envía.

La tecnología SDN (*Software Defined Network*) brinda la idea de separar el plano de control del dispositivo de red y colocarlo en un controlador que realice todas las funciones de este plano, la principal característica de esta tecnología es la capacidad de poder visualizar toda la red en tiempo real.

1.3.3. Presentación de la tecnología SDN

Progresivamente las redes se han expandido con la creación de nuevos protocolos por las necesidades que surgen de *routing*, seguridad, administración de la red, etc. Sin embargo, esto en un futuro puede quedar obsoleto, ya que, en la actualidad, se busca la automatización de procesos y este concepto no debe ser ajeno a las redes. Por estos motivos se requiere de redes que sean ágiles en la implementación de nuevos servicios, flexibles, eficaces, con mejor visibilidad para implantar políticas de seguridad, mejor implementación de calidad de servicio, adaptables a las necesidades de los usuarios. Ante estas necesidades se desarrolló la tecnología SDN, que puede realizar una separación física de la red en tres planos: plano de control, el plano de datos y plano de aplicaciones, ver figura 6 y 7.

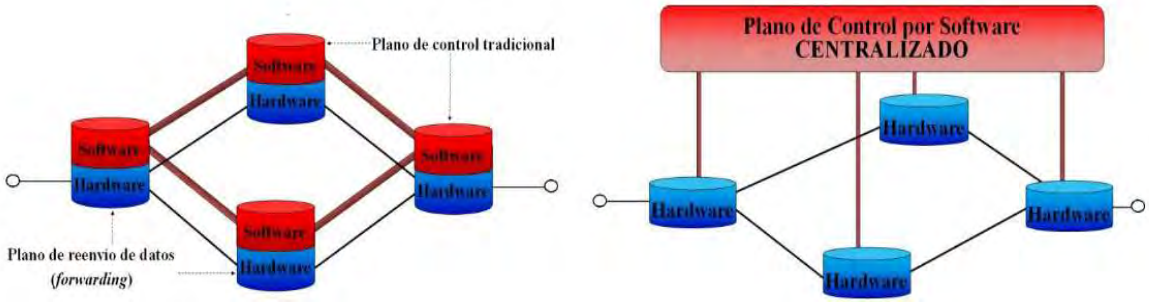

Figura 5: Separación de planos en SDN [8]

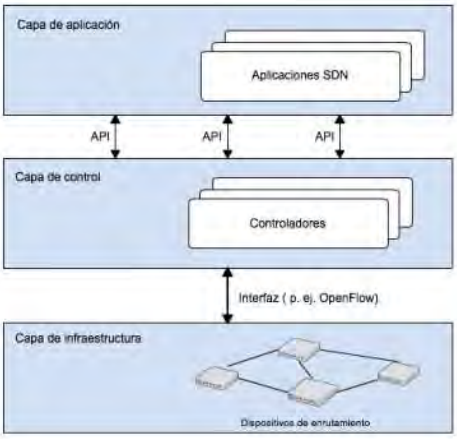

Figura 6: Arquitectura SDN [14]

Todo dispositivo de red posee un plano de control que se encarga de realizar todo el procesamiento de solicitudes de políticas de enrutamiento. Todas estas funciones están designadas a un controlador SDN como *OpenDaylight, NOX, Floodlight, Beacon*, etc que tiene una gestión de la red centralizada, envía todas las políticas y administra todo el flujo de datos; un plano de datos o plano de reenvió que realiza él envió y la recepción del flujo de datos; y finalmente, un plano de aplicaciones donde el administrador anuncia los requisitos de la red a través de una API (*Application Protocol Interfac*e) hacia el plano de control. Tras la separación de planos el único plano que se mantendrá en los dispositivos de red es el de datos [10].

Esta tecnología se puede aplicar en tres principales áreas de la red: red LAN (para campus), *DATA CENTERS* y red WAN, la aplicación de la tecnología SDN a las redes WAN se le conoce como SD-WAN (*Software Defined Wide Area Network*). En esta última se centrará el diseño de este trabajo de tesis.

1.4. Objetivos

1.4.1. Objetivo general

Esta tesis tiene como objetivo principal el diseño de una red SD-WAN en la Pontificia Universidad Católica del Perú basado en el modelo actual de su red para la mejorar el provisionamiento de nuevos servicios.

1.4.2 Objetivos específicos

1. Analizar los servicios actuales y realizar una estrategia de provisionamiento de nuevos servicios en la red PUCP.
2. Establecer redundancias entre los enlaces críticos para asegurar la supervivencia de la red en caso de fallos y permitir rutas alternativas.
3. Brindar un control masivo de todos los dispositivos conectados a la red.
4. Realizar la simulación de la red en una plataforma virtualizada.
5. Realizar pruebas para la medición de la calidad de servicio (QoS) y contrastar con la de la red actual.

2.TECNOLOGÍA SDN

En el presente capitulo, se explica la definición de la tecnología SDN como la solución a las problemáticas expuestas en el capítulo anterior, se presentan los componentes del despliegue de la tecnología y las áreas donde se aplican.

2.1. SDN

Es una arquitectura de redes agiles que se implementan con la finalidad de tener la capacidad de programabilidad dinámica a los equipos de red individualmente. Esto brinda un mayor control y visión total de los equipos; y facilita los procesos en la gestión de los servicios de red [16]. Se realiza por medio de la separación de los planos de control y de datos. Se designa todas las funciones del plano de control a un controlador, mientras que las funciones del plano de datos se realizan únicamente en el dispositivo de red.

2.1.1 Arquitectura SDN

Según el documento RFC 7426, la arquitectura de las redes SDN presentan 5 planos las cuales se presentan a continuación.

2.1.1.1 Application Plane

Este plano contiene todas las aplicaciones y servicios que la red ofrece su función es realizar la comunicación de las necesidades que posee la red al plano de control por medio del uso de una API *Northbound*.

2.1.1.2 Control Plane

Este plano se encuentra el controlador y tiene la función de controlar los dispositivos de red, para lo cual realiza comunicaciones hacia el *forwarding plane* por medio de API *Southbound*. Las instrucciones que envía permiten saber cómo se debe procesar, cual es el destino y por donde se debe enviar el tráfico.

2.1.1.3 Forwarding Plane

En este plano se encuentran los dispositivos de red, es el encargado de realizar las acciones sobre el tráfico entrante y saliente del dispositivo bajo las instrucciones enviadas desde el plano de control.

2.1.1.4 Management Plane

Es el plano que cumple la función de monitorear, realizar configuraciones y mantenimiento de los dispositivos de red por medio de instrucciones específicas. Principalmente este plano se relaciona con el *operational plane*, ya que le brinda las instrucciones correspondientes.

2.1.1.5 Operational Plane

Es el plano que cumple la función de verificar el correcto funcionamiento de los dispositivos de red y es el que se relaciona con los componentes de estos equipos como memoria, puertos, etc.

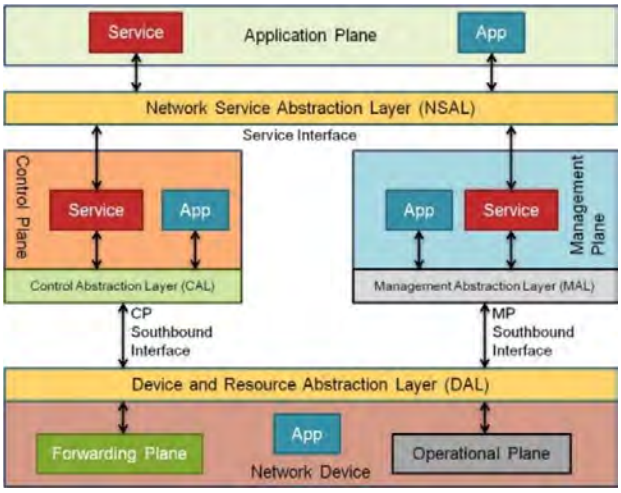

Figura 7: Arquitectura SDN [17]

2.1.2 Interfaces de comunicación

La arquitectura SDN como ya se explicó en el capítulo uno tiene como una característica la independencia de proveedores, ya que se usan interfaces de código abiertas para la comunicación entre planos.

Este método de comunicación se realiza a través de las API's (*Application programming Interface*) que son protocolos que permiten la interacción de aplicaciones o servicios sin la necesidad de conocer como fueron implementados [18]. Para la arquitectura SDN se usan tres tipos API's que se explicaran a continuación.

2.1.2.1 API Northbound

Es la interfaz que permite la comunicación del plano de control y el plano de aplicaciones. Esta API le permite, a este último plano, enviar instrucciones o políticas al controlador en base a las necesidades de la red. Para esto el proveedor proporciona una lista de funciones que esta interfaz puede usar. El controlador recibe las instrucciones interpretando en un lenguaje de alto nivel para cada nodo del plano de datos. Un ejemplo de esta interfaz es REST API [19].

2.1.2.2 API Southbound

Es la interfaz que permite la comunicación entre los planos de datos y control. La importancia de esta API es que cumple la función de realizar él envió , por parte del controlador, de instrucciones por medio de un único protocolo para el comportamiento del flujo de datos en los dispositivos de red y realizar cambios en tiempo real en base a las necesidades que se presentan [19]. Un ejemplo de esta interfaz es el protocolo *OpenFlow*.

2.1.2.3 API Eastbound

Esta interfaz permite la comunicación entre controladores en el plano de control, en caso existan más de uno, para establecer interoperabilidad entre estos. La principal utilidad de esta API es el intercambio de datos entre controladores, permitir el monitorio de la red en caso algún nuevo equipo se une a la topología o la verificación de la actividad los controladores [20].

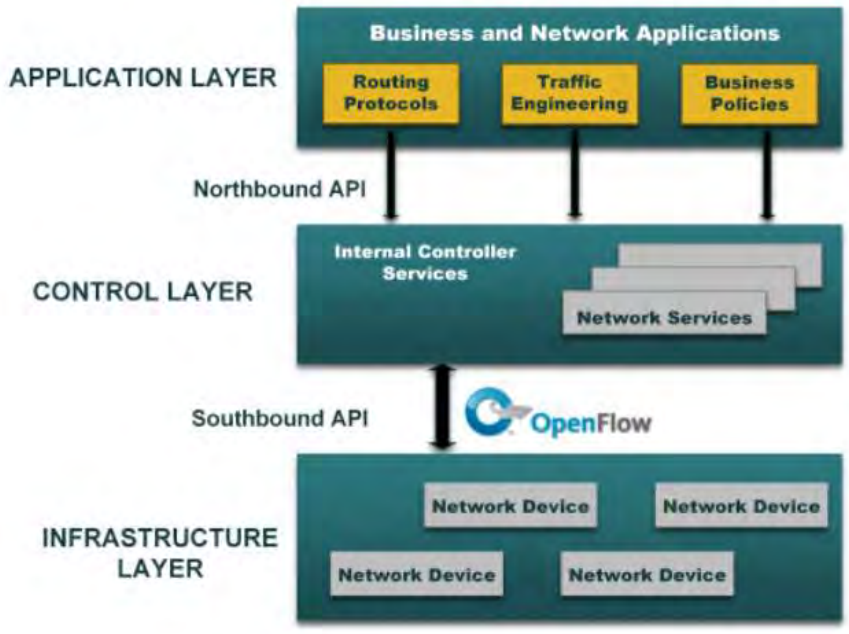

Figura 8: Arquitectura SDN con las interfaces de comunicación [21]

2.1.3. Modelos de despliegue

El despliegue de una red SDN se puede realizar en tres modelos [10]:

2.1.3.1 Modelo SDN basados en dispositivos

Este modelo presenta una serie de equipos de red físicos que operan bajo las instrucciones de un mismo controlador en el *Control Plane*. Como se observa en la figura 10.

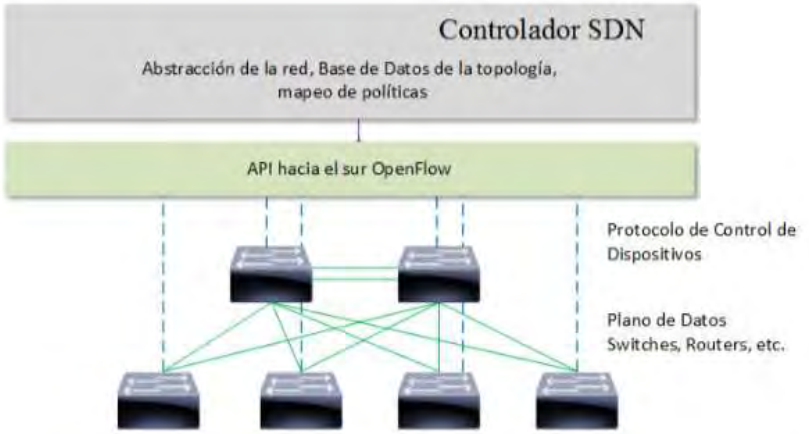

Figura 9: Arquitectura SDN con las interfaces de comunicación [10]

2.1.3.2 Modelo SDN Overlay

Este modelo se centra en la superposición de redes, es decir se crean capas en la red física para obtener conexiones entre puntos virtuales o puntos físicos separados y transparentes de la red física como se puede observar en la figura 11. La diferencia con el primer modelo es el uso de *switchs* virtuales que están establecidos en hipervisores en un ambiente de virtualización de servidores. Estos equipos son los nodos finales del *Forwarding Plane*, el flujo entrante y saliente correspondiente lo maneja el controlador sin afectar el plano físico ni el plano adyacente.

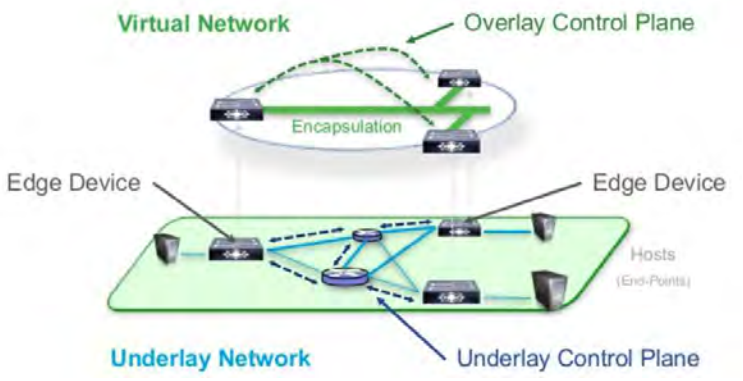

Figura 10: Modelo SDN Overlay [22]

La forma de crear este tipo de conexión es a través de túneles llamados *Overlay* con protocolos como VxLan (*Virtual Extensible LAN*), NVGRE (*Network Virtualization using Genering Routing Encapsulation*), etc. Estos túneles normalmente tienen como destino *switchs* virtuales dentro de los Hipervisores, el factor que se debe tomar en cuenta es que se pierde visibilidad por parte de controlador por la dificultad de operar ambas redes (física y virtual) [15] [23] [10].

2.1.3.3 Modelo SDN hibrido

Este modelo se basa en la inclusión de las tecnologías tradicionales y tecnologías SDN en una misma red, debido a esta unión, los componentes(controlador, protocolos de comunicación, etc) deben ser híbridos, es decir deben interactuar con equipos *Legacy* (tecnología actual) y componentes SDN [24].

2.2. Arquitectura SDN con protocolo OpenFlow

Esta arquitectura presenta cuatro componentes: el *Switch OpenFlow*, el controlador, el canal *switch*-controlador y el protocolo *OpenFlow*. En la figura 12 se muestra una imagen donde se puede apreciar todos los componentes de dicha arquitectura.

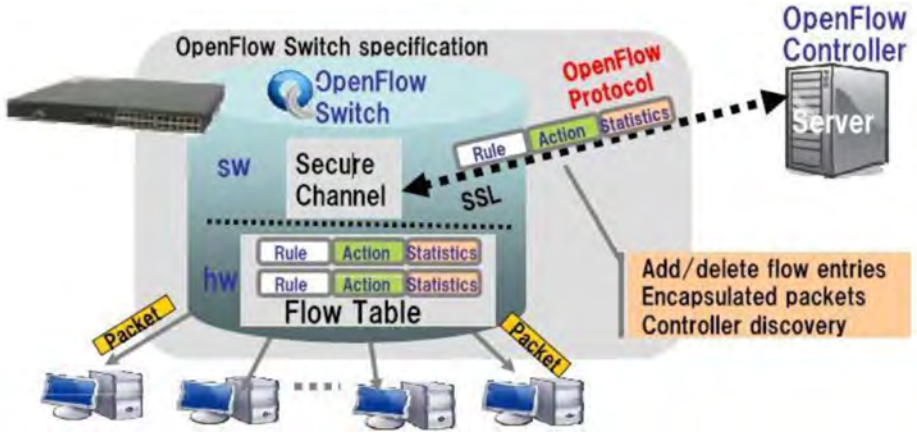

Figura 11: Arquitectura SDN/OpenFlow [1]

La entidad encargada de realizar la estandarización de los requerimientos del *Switch OpenFlow* es la ONF (*Open Networking Foundation*). Estas especificaciones incluyen los componentes, el funcionamiento lógico del *switch* y la manera en que se controla la red desde el controlador.

2.2.1. Switch OpenFlow

El *switch OpenFlow* está conformado por uno o más *Flow tables* y *Group tables*, que realizan la búsqueda y él envió de los paquetes, y uno o más canales *OpenFlow* que dirigen hacia el controlador como se puede observar en la figura 12. A continuación se especificaran estos componentes el *switch.*

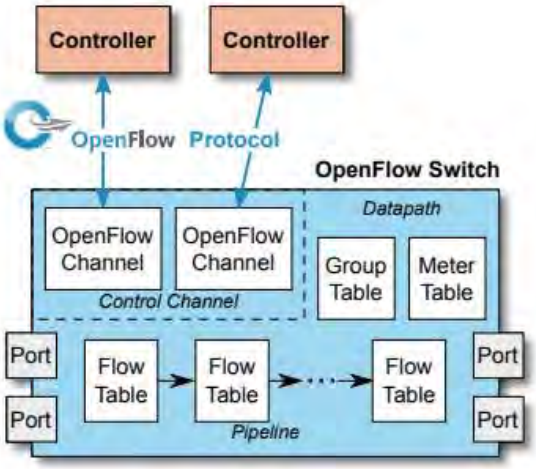

Figura 12: Componentes del Switch OpenFlow [25]

2.2.1.1 Flow Table

Una Flow table es la forma que presenta el *switch OpenFlow* una tabla de *Forwarding* en un *switch* tradicional, es decir muestran los destinos de los puertos. Un *switch* debe contener al menos una *Flow Table* y a su vez esta consiste en uno o más *Flow Entries* [25].

2.2.1.2 Flow Entry

Una *Flow entry* es un elemento de la *Flow table*. Es una regla de envío que posee el *switch OpenFlow*, esta contiene campos que se usan para la comparación con las cabeceras de los paquetes que llegan al *switch*. Estos campos son seis en el *switch* versión 1.5.1 [25] [1] :

- Match Field: en este campo contienen los puertos de ingreso, dirección IP destino y origen, etc.
- Priority: en este campo se encuentra el orden de prioridad que posee el *Flow entry*.
- Counters: este campo es un contador que aumenta la cuenta cuando ocurre una coincidencia del *Flow entry*.
- Instructions: en este campo se encuentra el conjunto de acciones que se aplicaran sobre el paquete. Estas acciones pueden ser eliminar el paquete, enviar por algún puerto del *switch* o enviarlo al controlador.
- Timeouts: este campo se divide en dos valores de tiempos para la *Flow entry*: el primero es el *Idle Timeout*, es el tiempo que debe pasar desde la última coincidencia para que este *Flow entry* sea eliminado de la *Flow table*. El segundo es el *Hard Timeout*, este tiempo es el que debe pasar desde el momento que se instaló el *Flow entry* en el *switch* para que sea eliminado, independientemente si ocurre o no una coincidencia.
- Cookies: campo cuyo contenido lo determina el controlador. Comúnmente se usa para realizar la cuenta de algún parámetro relacionado con los paquetes recibidos por *Flow entry*.

A continuación se muestra en la figura 13 los campos de una *Flow entry*.

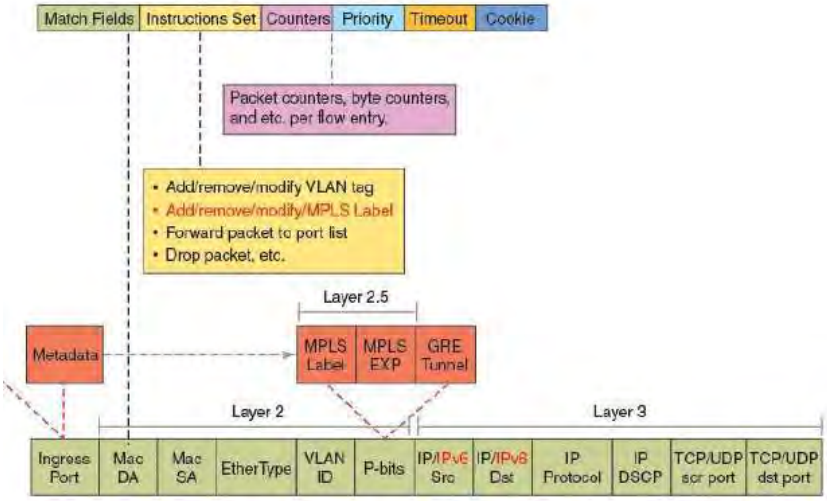

Figura 13: Campos de una Flow entry [26]

2.2.1.3. Group Table

Un *Group Table* consiste en un Group *entries*. La función de estas es de brindar métodos adicionales de *Forwarding* por medio de un grupo de *Flow entries* que apuntan hacia dicho *Group table* [25].

2.2.1.4 OpenFlow Ports

Los *OpenFlow Ports* son interfaces de red que permiten el paso de paquetes hacia el procesamiento interno de los *switchs OpenFlow*, además sirven de entrada y salida en él envió de paquetes entre switchs. Este debe admitir 3 tipos de *OpenFlow Ports* [25]:

- Puertos físicos: son puertos que corresponden a una interfaz de la parte del hardware del *switch*.
- Puertos lógicos: son puertos que se encuentran definidos en el *switch* por métodos que no son de *OpenFlow*. Además, no necesariamente corresponden a una interfaz del hardware.
- Puertos reservados: son puertos lógicos que se encuentran reservados por defecto para funciones específicas de reenvío como por ejemplo hacia el controlador.

2.2.2 Controlador OpenFlow

El controlador reside entre en el *Control plane*. Este cumple la función de ser el cerebro de la arquitectura, ya que es el que traslada las especificaciones del plano de aplicación hacia el plano de control por medio de *Flow entries*. A continuación, se brindará una comparación de los controladores SDN más representativos.

2.2.2.1 Controlador Nox/Pox

NOX fue el primer controlador de código abierto, diseñado específicamente para investigación de componentes y protocolos *OpenFlow* en código C++. Luego de uso de este controlador por años, estas las limitaciones impulsaron la creación de otro controlador llamado POX. El controlador POX, administrado por *Open Networking Lab*, de la misma manera usado para las mismas tareas que NOX usa un lenguaje de programación Python y C++, pero con mejores aplicaciones para dicha tarea como por ejemplo brinda una interfaz gráfica, soporte para descubrimiento de topologías [27][28].

2.2.2.2 Controlador Beacon

Controlador desarrollado en lenguaje de programación Java, de la misma forma es de código abierto y como principal característica es que es de desarrollo rápido.

2.2.2.3 Controlador Floodlight

Este controlador fue una actualización del controlador Beacon de la misma manera es de código abierto, fue diseñado para realizar investigaciones en SDN en lenguaje de programación Java. Soporta *Switch OpenFlow* físicos como virtuales desde la versión 1.0 hasta 1.3. además, tiene la característica de poder usarse sobre entornos diferentes a *OpenFlow* [27].

2.2.2.4 Controlador OpenDayLight

Este controlador desarrollador principalmente por Linux *Foundation*. Es de código abierto y el lenguaje de programación que se usa es Java, cabe destacar también la compatibilidad con el uso de la herramienta Mininet además que tiene una amplia lista de compañías que le brindan soportes entre ellas esta Cisco, Huawei, IBM, Microsoft. Una característica resaltante de este controlador es que es multiprotocolo y modular, es decir permite que los usuarios puedan elegir un solo protocolo o múltiples protocolos para la solución de problemas [29][30].

2.2.3 Diagrama de bloques

El controlador presenta diagrama de bloques internamente divididos en 3 niveles con bloques que cumplen diferentes funciones relacionadas al plano de aplicación y al plano de datos como se puede observar en la figura 14. A continuación, se explicará más a detalle estos bloques.

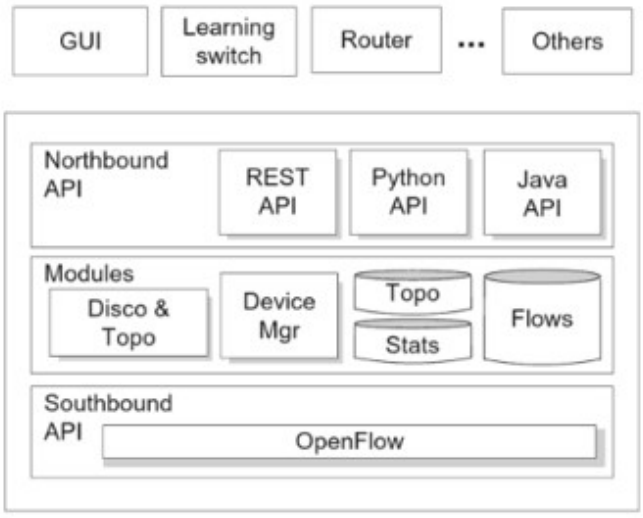

Figura 14: Diagrama de bloques controlador [31]

2.2.3.1 Módulos Core y de servicios

El controlador por medio de estos módulos aprende de cómo se encuentra en tiempo real la topología, es decir de la existencia de los *switchs* y las conexiones con los usuarios finales. Este módulo cumple las siguientes funciones [31]:

- Descubrimiento de usuarios finales como por ejemplo laptops, celulares, impresoras, etc.
- Descubrimiento de equipos de red, se entiende como equipos que cumplen una función de la red como por ejemplo *routers*, *switchs*, *Access point*, etc.
- Descubrimiento de la topología de red, esta función permite que el controlador mantenga información sobre interconexiones de los equipos de red con otros equipos sean usuarios finales o dispositivos de red.
- Gestión de flujo, esta función permite mantener una base de datos de los flujos que realiza el controlador y las instrucciones que envía a los dispositivos en el plano de datos.

2.2.3.2 Modulo de aplicaciones

Este módulo cumple la función de realizar los cambios necesarios en el plano de datos por medio del uso de los módulos de *core* y de servicios. Estos cambios se realizan programando el plano de datos cuando se inserta, se quita o se modifican *Flow entries* en dicho plano. Los principales módulos que se usan son los siguientes [1]:

- Forwarding:

En caso de la llegada de algún paquete, este módulo busca el camino más corto al destino y para realizar su reenvío se debe instalar *Flow entries* en el *switch*.

- Firewall:

Brinda seguridad mediante la filtración de paquetes según dirección IP y puerto de servicios como http, tcp, udp.

- Access list

Permite el bloque de conexiones a usuarios según su nivel de acceso.

- Port down reconciliation:

En caso de caída de algún puerto del *switch,* se eliminan los *Flow entries* que contenían en algún aspecto a este puerto.

2.2.3.3. Módulos de interfaces

El uso de las API en el controlador permite el acceso del plano de aplicaciones a la red, específicamente la API *northbound*, el controlador usa este método para enviar paquetes con información de eventos en la red hacia el plano de aplicación y a través de mensajes de respuesta hacia el controlador envía las acciones correspondientes [31][1] como se puede observar en la figura 15. El módulo API que se usa principalmente se denomina REST API continuación se explicara.

Este tipo de modulo usan únicamente API's del tipo REST (*Representational State Tranfer*), se usa principalmente para obtener información o generar operaciones, por medio de HTTP, de datos en formatos JSON o XML.

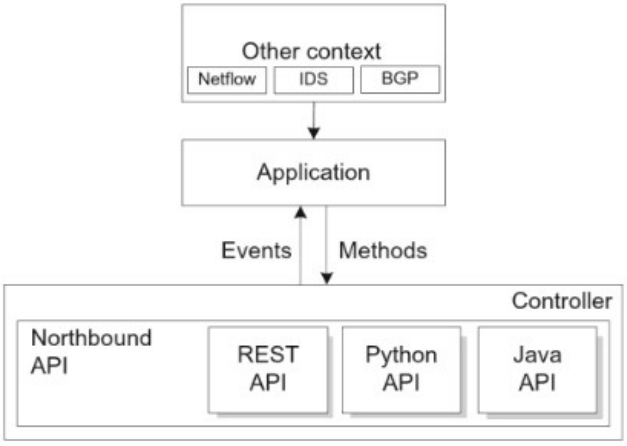

Figura 15: Mensajes entre *control plane* y *application plane* [31]

2.2.4 Protocolo Openflow

Este protocolo consiste en una serie de mensajes que se envían desde el controlador hacia el *switch* y de la misma manera una serie de mensajes en respuesta en sentido contrario. Estos mensajes permiten programar a los equipos según la necesidad que presente la red por medio de creación, eliminación de los *Flow entries* en el *switch*. A continuación, se especifican cuáles son estos tipos de mensajes.

2.2.4.1 Mensajes Controlador – *Switch*

Este tipo de comunicación las inicia el controlador y no necesariamente requiere de una respuesta por parte del *switch*. Los principales mensajes de este tipo se muestran a continuación:

- Features: el controlador solicita una identificación del *switch* junto con sus características como por ejemplo modelo, dirección IP, etc. El *switch* debe enviar una respuesta, este mensaje es el que se envía cuando se establece el protocolo [25].
- Configuration: el controlador envía este tipo de mensajes para establecer parámetros en el *switch* [25].
- Modify-state: este mensaje se envía para realizar modificación, aumentar o eliminar alguna *Flow entry* [25].
- Packet-out: este tipo de mensajes es usado por el controlador para realizar él envió de paquetes previamente enviados desde el *switch* por medio del mensaje *Packet-in*.

En este mensaje se encuentran las acciones a tomar sobre el paquete como por ejemplo por cual puerto del *switch* se debe enviar o descartar dicho paquete [25].

2.2.4.2 Mensajes *Switch* – Controlador

Este tipo de mensajes son enviados desde el *switch* sin solicitud del controlador, también se les conoce como mensajes asíncronos. Este tipo de mensaje se usa para notificar al controlador de la llegada de un paquete al *switch*. A continuación, se explica los principales mensajes de este tipo que se usan.

- Packet-in: este tipo de mensaje le brinda el control de un paquete, que llego a un puerto del *switch*, al controlador para el análisis de las acciones que se deben tomar con dicho paquete [25].
- Flow – removed: este tipo de mensaje notifica al controlador sobre la eliminación de una *Flow entry* en alguna *Flow table*. Este mensaje se envía en respuesta a un pedido de eliminación de dicha *Flow entry* por parte del controlador [25].
- Port – status: este mensaje notifica al controlador sobre el estado del puerto en caso de algún cambio en la configuración por ejemplo en caso de alguna caída del enlace que conduce a dicho puerto [25].

2.2.4.3 Mensajes simétricos

Este tipo de mensajes se envían en cualquier dirección sin necesidad de alguna solicitud de alguna de las partes. A continuación, se muestran cuáles son estos mensajes[25].

- Hello: estos mensajes se envían por ambas partes (*switch* y controlador) para iniciar la comunicación entre ambos.
- Echo: estos mensajes se envían por ambas partes para notificar que se encuentran en funcionamiento.
- Error: mensaje que se envía como notificación de alguna falla.

2.2.5 Canal seguro *Switch* – Controlador

Este canal es usado para el intercambio de mensajes entre el controlador y el *switch*, usualmente el protocolo *Openflow* usan más de un canal *OpenFlow* para cada diferente *switch* que posee la red, pero también puede existir más de un canal entre el controlador y el *switch* para que funcione como respaldo en caso de fallas del primero. Se usa el protocolo TLS (*Transport Layer Security*) para brindar la seguridad en el canal. Este tipo de conexión puede realizarse usando 2 métodos [31]:

El primero es el método *in-band*, en este caso se usan puertos de este *switch OpenFlow*, el controlador envía paquetes a través de un puerto especifico con la seguridad necesaria usando *Flow entries* como se puede observar en la figura 16 las líneas punteadas, en ese caso se usa el puerto K.

El segundo es el método *out-of-band*, en este caso se usan conexiones dedicadas externas al plano de datos del *switch OpenFlow*, principalmente se usa este tipo de conexión cuando la red es de configuración hibrida. Como se puede ver en la figura 16 la línea continua se usa el puerto Z que no se encuentro como puerto del plano de datos.

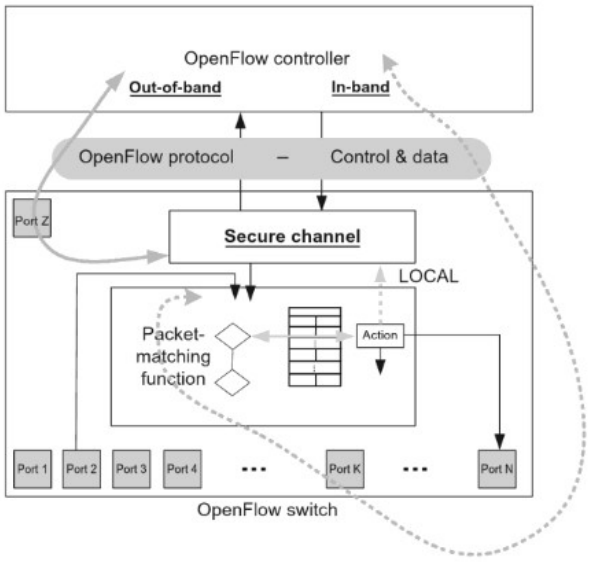

Figura 16: Canal seguro controlador- switch [31]

2.3. Funcionamiento de la arquitectura SDN

El funcionamiento se divide en dos procesos: proceso de ingreso y el proceso de egreso.

El primero inicia con la llegada de un paquete por algún puerto del *switch*, este realiza la extracción de la cabecera de dicho paquete y empieza la comparación del campo *match field* del *Flow entry* con mayor prioridad según el campo *priority* que posea con la cabecera del paquete, en caso exista alguna coincidencia entre estos campos se ejecuta el campo *instruction set* (se actualizan el campo *counters*) de la *Flow entry* que se obtuvo conciencia. En caso de que dicha instrucción dirija a otra *Flow entry* se realiza todo el proceso anterior, caso contrario se ejecuta el *action set* sobre el paquete para finaliza con el *packet-out* si no tuviese una *Flow table* de egreso [25].

Luego inicia el segundo proceso que es el de egreso, en caso posea una *Flow table* de egreso se realiza las acciones sobre el paquete de la misma manera que el proceso de ingreso, pero con la diferencia que la prioridad de las *Flow entries* que se usan ya no puede ser menor a la prioridad de la *Flow entry* que se usó para el proceso de ingreso. Caso contrario que en el proceso de ingreso no se encuentre coincidencia con alguna *Flow entry*, el controlador debe colocar una nueva *Flow entry* para determinar la acción que se tomara sobre ese paquete. Estas acciones pueden ser descartar el paquete, enviar por algún puerto por defecto o enviárselo al controlador por el canal seguro *switch*-controlador para el respectivo análisis. A continuación, se muestra en la figura 17 el diagrama de flujo del funcionamiento de esta arquitectura [25].

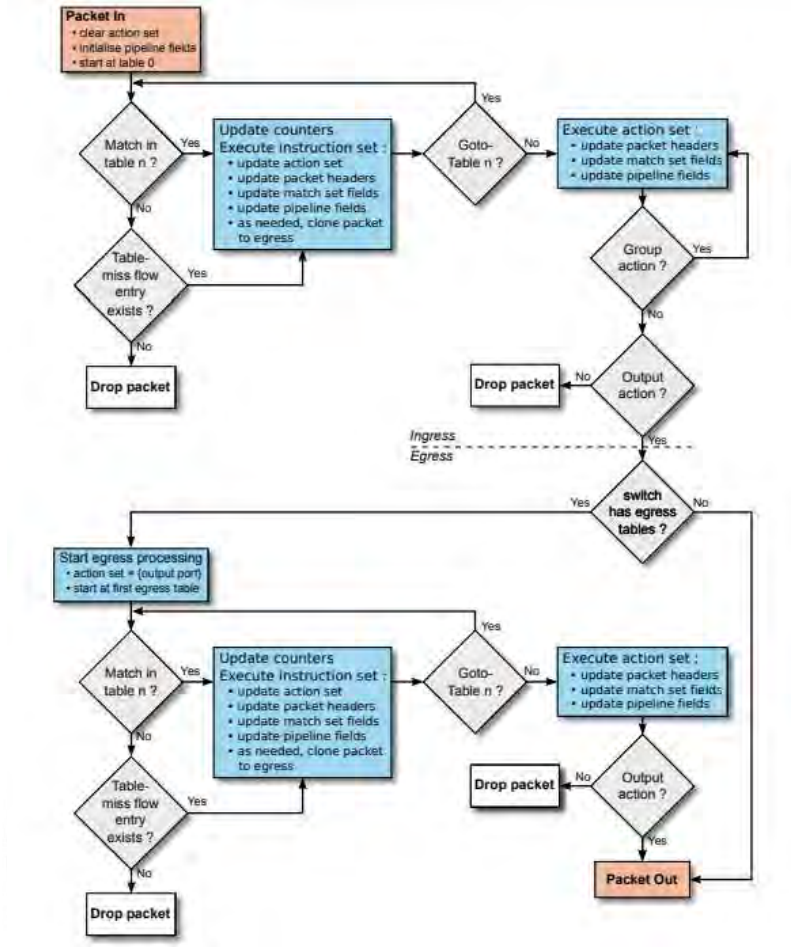

Figura 17: Diagrama de bloques del funcionamiento de la arquitectura SDN [25]

2.4 Escenarios de aplicación de SDN

Los escenarios donde se han implementado la tecnología SDN han sido tres: redes de campus, redes WAN y *DATA CENTERS*.

2.4.1 Redes de campus

Los motivos de la implementación de esta tecnología son dos. Primero es por la variedad de dispositivos que posee y los servicios que brinda con diferentes tipos de tráfico como por ejemplo voz, video y videoconferencia. Segundo es la identificación de usuarios y los controles de acceso que les brinda [25].

2.4.2. Red WAN

Las deficiencias en esta red se centran en 2: la primera es por la caída de enlaces de servicios críticos por saturación o fallas en dichos enlaces y la segunda es por el alto costo de las conexiones a las oficinas externas, ya que actualmente se usan conexiones dedicadas como VPN, MPLS, etc [25]. Un ejemplo de esta implementación es la red de GOOGLE.

2.4.3. DATA CENTERS

Las deficiencias en este tipo de red es la gran cantidad de volumen de tráfico que se puede manejar, la escalabilidad de la red, la latencia debe ser baja y finalmente la necesidad de un mayor ancho de banda en los enlaces entre equipos [25].

2.5. Presentación de la implementación SD-WAN en la red PUCP

Como se explicó en la sección anterior una de las áreas donde se aplica la tecnología SDN es en la red WAN la cual se denomina SD-WAN, la función tradicional que tenía este tipo de red es brindar conexión a sucursales externas con las aplicaciones que se encontraban en los servidores, principalmente estas conexiones eran dedicadas por medio de MPLS o VPN, ya que es necesario garantizar la confiabilidad y seguridad de la conexión.

Actualmente la red PUCP tiene 14 edificios externos cuyas conexiones hacia la red central, la cual posee todos los servicios que son a través de una VPN como se puede observar en la figura 18, donde solo se presentan dos conexiones a edificios externos (la escuela de música y el centro de idiomas católica de Pueblo libre).

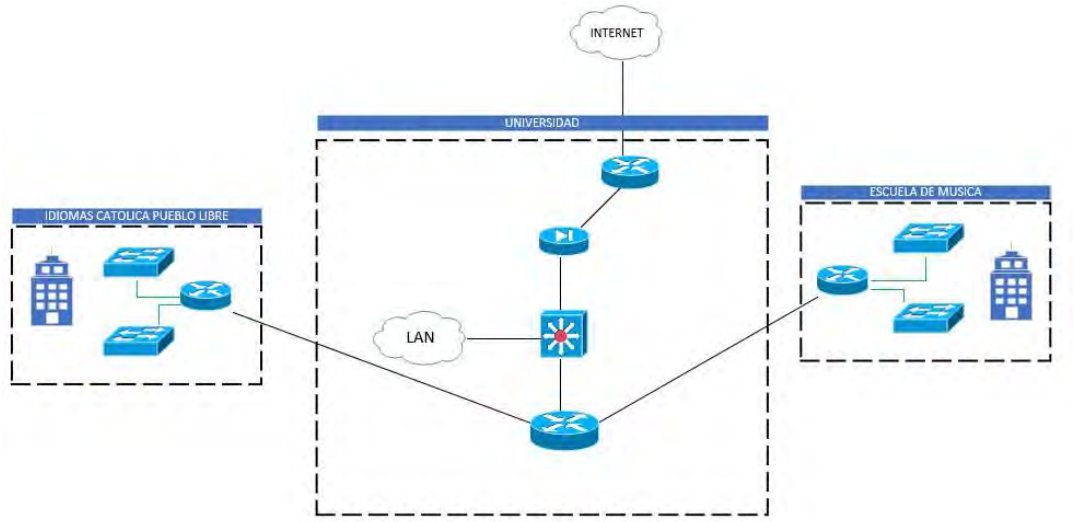

Figura 18: Conexiones físicas red WAN PUCP
Fuente: Elaboración propia (2020)

Este tipo de red posee las siguientes limitaciones:

- Alto costo de las conexiones: estas conexiones de MPLS o VPN, como se puede observar en la figura 18 las conexiones de negro son costosas por todos los servicios que brindan.
- En caso de la red PUCP que posee diferentes tipos de servicios, estas conexiones deben ser confiables, ya que transmite tráfico de diferente tipo (voz, video, etc) y con un mejor ancho de banda para el despliegue de nuevos servicios.
- La gestión de red es complicada, ya que no se tiene una visibilidad completa de la red.
- La necesidad de redundancia de enlaces en caso de fallas o caídas en las conexiones.

La red que se propone con el uso de la tecnología SDN tendrá las siguientes características:

- Se usará una conexión de internet de ancho de banda como conexiones entre los *routers* SD-WAN de los edificios externos, ya que son menos costosas que las conexiones dedicadas, las características de la conexión de fiabilidad y seguridad la brindara el controlador.

- El controlador tendrá visibilidad completa de la red, ya que todos los dispositivos de red se encuentran conectados por medio de canales seguros *switch*-controlador y por lo tanto tendrá una gestión centralizada como se puede observar en la figura 20.
- Gestión basada en políticas para mejorar la calidad de servicio y disminuir la latencia del tráfico como por ejemplo voz, video, etc.
- El *troubleshooting* será más eficaz por el control generalizado que tendrá el controlador.

A continuación, se muestran las topologías de la red propuesta en la figura 19 y 20: lógica y física.

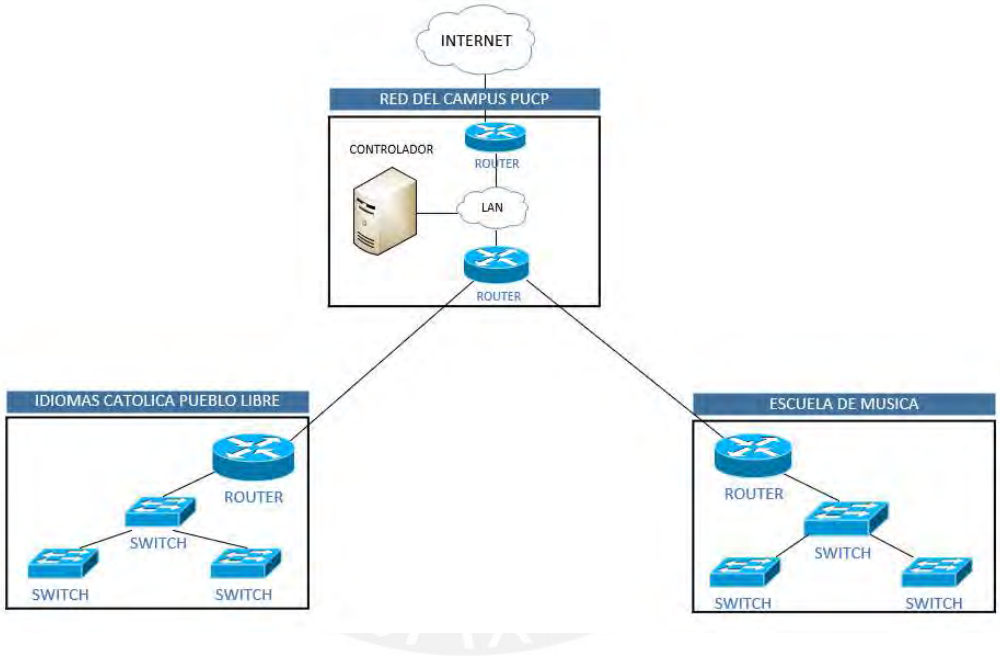

Figura 19: Topología física SD-WAN PUCP
Fuente: Elaboración propia (2020)

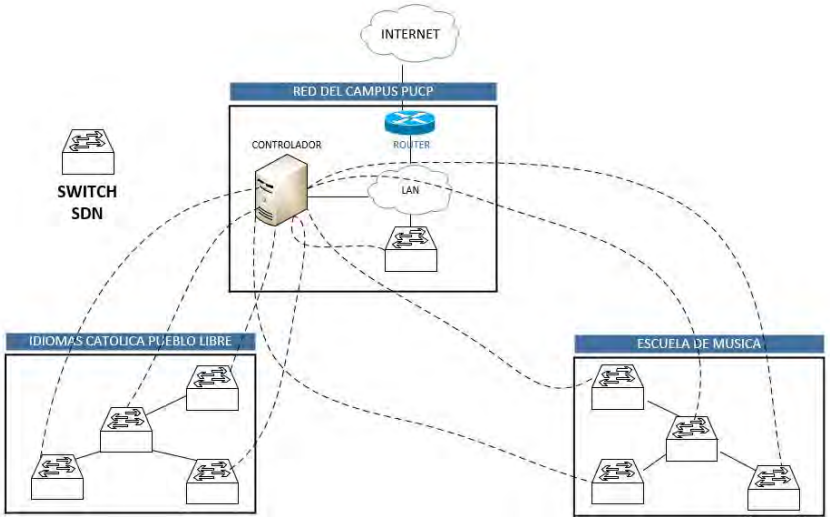

Figura 20: Topología lógica SD-WAN PUCP
Fuente: Elaboración propia (2020)

Los parámetros medibles en la red propuesta:

- Calidad de servicio: es el modo que usa una red para brindar fluidez por medio de la designación de prioridades en los diferentes tipos de tráficos.
- Latencia de red: es el tiempo que tarda los paquetes en llegar desde el nodo de origen hasta el nodo de destino.
- Costo de implementación
- Nivel de *troubleshooting*: es el nivel de complejidad que posee la red para la solución de un problema en la red.

2.6 Comparación de soluciones SD-WAN actuales

Se realizará una comparación entre distintas soluciones actuales de SD-WAN más relevantes con la que se presentará en este documento, en consideración de los requerimientos mínimos que se puedan necesitar, en la tabla mostrada a continuación.

Tabla 1. Tabla de soluciones, modificado de [40][41][42]

TABLA DE SOLUCIONES			
Requerimientos	**CISCO**	**FORTINET**	**PRESENTADA**
Equipos	6	3	6
Controladores	3	0	2
Licenciamiento	6	0	0
Almacenamiento	1000GB	Integrado	32GB

Fuente: Elaboración propia (2020)

En la opción del proveedor Cisco, se considera seis equipos por la redundancia (dos equipos por cada sede) en las opciones se podrían contemplar equipos vEdge o cEdge que poseen funcionalidades específicas para una solución SD-WAN, en la parte de los controladores se deben contemplar tres: vManage que mantendrá el plano de administración, vBond que tendrá el plano de orquestación y vSmart que tendrá el plano de control. Todos los controladores mencionados deberán ser desplegados *On Premise,* es decir almacenados en un servidor interno en la sede central. En relación con el almacenamiento y licenciamiento, se debería utilizar un servidor con capacidad de 1000GB para el despliegue junto con la adquisición de 6 licencias para subscripción de DNA Routing para cada equipo mencionado.

En la opción del proveedor Fortinet, se encuentra orientado a la seguridad de las redes, se considera tres equipos (un equipo por cada sede) en las opciones que se pueden desplegar Fortigate VM (01V,02V,04V,08V y 16V) y FortiManager. No se requiere ningún controlador porque ya los equipos vienen integrados con la funcionalidad de SD-WAN junto con las licencias necesarias.

Tomando en cuenta ambos proveedores, los costos son elevados en comparación con un software libre que es la solución propuesta y en consecuencia a continuación en el capítulo 3 se realizara el diseño de la red SD-WAN en base a los requerimientos generales de cada componente necesario.

3. DISEÑO DE UNA RED SD-WAN

En el presente capitulo, se mostrarán los requerimientos para cada plano de la red SDN que se deben considerar. Además, se realizará la selección de equipos y software necesarios para implementar este trabajo en una simulación de una red SD-WAN para la red PUCP.

3.1 Componentes a seleccionar

Como ya se explicó en el capítulo dos, los componentes necesarios en una red SDN son los siguientes:

- Versión de protocolo OpenFlow
- Controlador
- *Switch* SDN

A continuación, se presentan los requerimientos del diseño.

3.2 Requerimientos generales

3.2.1 Open Source

Es necesario que la solución sea *Open Source* para evitar la dependencia de los proveedores y permite la interoperabilidad del sistema a diferentes fabricantes. Además de permitir componentes de la red de diferentes orígenes.

3.2.2 Escalabilidad de la red

La red deberá permitir el aumento masivo de dispositivos móviles a la red y que pueda permitir sin problemas el manejo de todo el tráfico. Aproximadamente deberá permitir un crecimiento de 20 000 usuarios y 160 000 conexiones de forma simultánea [1].

3.3 Requerimientos en el plano de datos

3.3.1 TCAM en los equipos SDN

Es necesario que el *Switch* SDN posea un numero alto de cantidad de *Flow entries* que se puedan colocar en una TCAM (*Ternary content-addressable memory*), actualmente los equipos de alto performance tienen una cantidad de aproximadamente 8000. Estas memorias son costosas por tal motivo esta cantidad mencionada es muchas veces baja en equipos regulares.

3.3.2 Redundancia y Tolerancia del Switch

En ámbitos actuales, es necesario que los equipos de borde tengan una protección ante fallas del propio equipo o tener tolerancia ante caídas. Esto se logra por medio de la operabilidad

de dos equipos del mismo modelo en Alta disponibilidad e interoperabilidad de protocolos como HSRP y VRRP, o en el caso más básico el uso de la estrategia de *Stacking*. De la misma manera las conexiones por túneles o físicas usados hacia el controlador deben tener una protección ante algún fallo del enlace por lo tanto se usará la estrategia de *Link Agregation* que permite usar hasta 8 conexiones físicas como un único enlace lógico para proporcionar tolerancia a fallos y agregar un mayor ancho de banda en la comunicación.

3.4 Requerimientos en el plano de control

3.4.1 Ancho de banda del canal seguro *switch*-controlador

Dependiendo del tipo de conexión que se usara para el canal seguro *switch*-controlador: *in band* o *out of band*. En el primer caso, como ya se explicó en el capítulo dos, se usan puertos de este *switch*, por lo tanto, la cantidad de puertos usados no deben sobrepasar de un máximo colocado por el administrador para asegurar la confiabilidad del ancho de banda para el tráfico de paquetes de los usuarios. Por otro lado, para el segundo caso esto es indistinto, ya que los enlaces que se usan para el tráfico de datos y el tráfico de control son distintos. En la figura 21 se puede observar la diferencia entre estos dos enlaces.

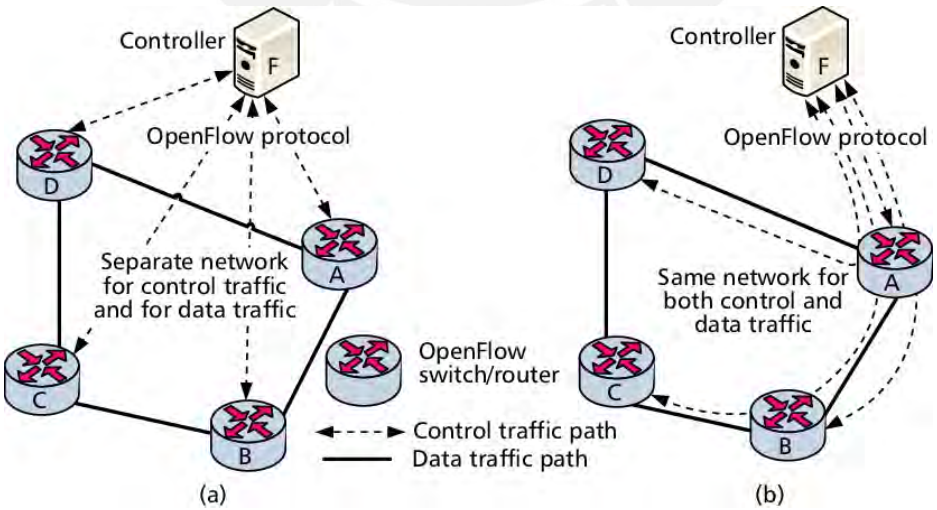

Figura 21: Conexiones del canal seguro *Switch*- controlador a) *Out of band* y b) *In band* [32]

3.4.2 Redundancia y Tolerancia del controlador

La redundancia del controlador nos proporcionara un control efectivo ante eventos premeditados y no premeditados como interrupciones para mantenimiento o caída del controlador. Además, brinda una mayor escalabilidad para un futuro crecimiento de la red. La estrategia de adopción de redundancia que se utilizara es la de alta disponibilidad que permite operación la red bajo dos controladores; el primario que posee toda la configuración y la comunicación con los equipos directamente por los caminos creados y un secundario que debe mantener comunicación constante con el primario para funcionar como respaldo y este tome el control de la red cuando ocurran eventos. Se considerarán dos controladores Floodlight en *Active/Standby*.

3.5 Selección de componentes

3.5.1 Elección de la versión de protocolo OpenFlow

Con el paso de los años, el protocolo ha sido mejorado con nuevas funcionalidades para satisfacer las necesidades que las redes tradicionales brindan y se realicen los cambios respectivos a esta tecnología nueva de SDN, a continuación, se detallan cuáles son las versiones OpenFlow existentes [26]:

- OpenFlow 1.0: fue la primera versión de este protocolo creado el año 2009, la comunicación de los componentes (*switch* y el controlador) por medio del protocolo TLS11, solo se limita a realizar las acciones del campo *instruction set* y actualizar los contadores.
- OpenFlow 1.1: la versión fue creada el año 2011 soporta MPLS, trafico *Multicast* con baja eficiencia, *Vlan's* y los puertos virtuales.
- OpenFlow 1.2: la versión fue creada el año 2011 el mes de diciembre, brinda soporte para direcciones IPv6.

- OpenFlow 1.3: la versión fue creada el año 2012, permite usar cabeceras extendidas para tráfico de IPv6 y permite la interoperabilidad de dos o más controladores en la misma red.
- OpenFlow 1.4: la versión fue creada el año 2013, permite usar TLV 15, notificación si la tabla *Flow entries* está llena.

Se considera la versión más completa la 1.3 además es la que posee más documentación. Por lo tanto, se escogerá la versión 1.3.

3.5.2 Elección del controlador

Para la selección del controlador se consideran las siguientes características:

- Lenguaje de programación

  Es un factor importante que varía por dos principales factores: el primero es la facilidad que posee el implementador con cada lenguaje y el segundo es el nivel de performance que posee el controlador que use dicho lenguaje, en el presente caso se escogerá el lenguaje de programación JAVA.

- Soporte de versión de protocolo OpenFlow

  Es necesario que el controlador a escoger posea soporte para la versión de protocolo OpenFlow que se usara en la simulación para el caso de esta tesis se usara la versión 1.3.

- Soporte de Sistema Operativo

  De la misma manera el controlador se debe poder implementar sobre el sistema operativo que se usara para la simulación en el caso de la tesis se usara el sistema operativo LINUX.

- Cantidad de documentación

  Los controladores actuales poseen gran cantidad de documentación, entonces la implementación de la simulación será más factible en base a otras simulaciones hechas

en otros trabajos. Por lo tanto, se debe escoger un controlador que posea abundante documentación.

- Performance

Algunos controladores poseen librerías que permites agilizar la simulación por ejemplo los controladores Floodlight y OpenDaylight poseen librerías que permiten realizar tareas de manejos de hosts [1].

Por estas características explicadas se decide escoger el controlador Floodlight, porque cumple con las expectativas de cada característica ya mencionada. A continuación, se muestra, en una tabla, todas las características de los controladores comparados.

Tabla 2. características de los controladores, modificado de [1][31][33]

Características	NOX	POX	Beacon	Floodlight	OpenDayLight
Cantidad de documentación	Poca	Poca	Abundante	Abundante	Media
Lenguaje de desarrollo	C++	Python	Java	Java	Java
Soporte de versiones de protocolo OpenFlow	1.0	1.0	1.0	1.0 - 1.5	1.0 - 1.3
Interfaz grafica	No	Python+ y Web	Web	Web	Web
Performance	Alta	Baja	Alta	Alta	Alta
Soporte de plataformas	Linux	Linux, Mac Os, Windows	Linux, Mac OS, Windows	Linux, Mac OS, Windows	Linux, Mac OS, Windows
Tipo de software	abierto	abierto	abierto	abierto	abierto

El controlador Floodlight brinda beneficios de facilidad de implementación por las interfaces de programa (API REST), brinda en su sitio web una variedad de ejemplos de codificación para diferentes simulaciones de redes SDN [34]. Finalmente, el controlador permite el control masivo de todos los dispositivos de red, ya que todos estarán conectados por medio del canal seguro.

3.5.3 Selección del modelo de Switch SDN

Para la selección del modelo de *Switch* SDN se toma en consideración las siguientes características.

- Compatibilidad con la versión de protocolo OpenFlow

De la misma manera que con el software y el controlador, el *Switch* debe ser compatible con la versión 1.3 del protocolo OpenFlow.

- Latencia

Es un factor importante en un *Switch*, ya que determina el tiempo que demora un paquete estándar de 64 Bytes hacia un destino, por lo tanto, a menor latencia el dispositivo brinda una mejor eficacia de direccionamiento de tráfico.

- Cantidad de puertos

La cantidad de puertos que posee un *Switch* determina la posibilidad de escalabilidad de la red. Por lo tanto, a mayor cantidad de puertos se podrán conectar más dispositivos y permitirá el crecimiento de la red.

- Cantidad de *Flow entries* en la TCAM (como se explicó en el punto 3.3.1).

Se realizo la comparación de 3 modelos de equipos: Pica8-P3297 de la marca Pica8, Summit X440-48t de la marca Xtreme y Intel FM6000 de la marca Intel. En base a las características mencionadas se escogió el *Switch* Pica8-P3297. A continuación, se muestra la tabla comparativa de los 3 equipos.

Tabla 3. Comparación de *Switchs* SDN [35][36][37]

Características	Pica8-P3297	Summit X440-48t	Intel FM6000
Version de protocolo OpenFlow compatibles	1.3	1.3	1.3 - 1.5
Entradas en una TCAM	8192	4096	entre 2K - 8K
Latencia	300ns	500ns	400ns
# Mac Addresses	32K	16K	64K
# Puertos	48 x 10/100/1000 Base-T RJ45 Y 4 puertos de 10 GbE SFP	48 x 10/100/1000 Base-T RJ45 Y 4 puertos de 100/1000BASE-X SFP	24 x 10GbE

El *Switch* pica8-P3297 fue la elección principalmente por la cantidad de *Flow entries* que admite su TCAM, ya que como ya se mencionó, estas cantidades son muy reducidos para equipos de bajo performance [35].

3.5.4 Selección del entorno de la simulación

Para la selección del software de simulación donde se armará la arquitectura se tomarán en consideración las siguientes características.

- Compatibilidad de versión de protocolo OpenFlow

  El simulador por escoger debe tener compatibilidad con la versión del protocolo OpenFlow a usar que en este caso como ya se mencionó será la versión 1.3.

- Tipo de licencia

  Para mayor facilidad del implementador el tipo de licencia de la plataforma debe ser libre para que el proceso de instalación sobre el sistema operativo sea más rápido.

- Compatibilidad con controladores

  El software debe ser compatible con el controlador a usar, es decir que se pueda instalar sobre dicho software.

Se realizo la comparación de tres plataformas de simulación en base a las características mencionadas, obteniendo como elección al simulador MININET, ya que cumple con todas expectativas de las características. Se muestra a continuación la siguiente tabla con la comparación realizada.

Tabla 4. Comparación de software de simulación [33][38]

Características	OPNET	VNX	MININET
Documentación	Baja	Media	Alta
Version de protocolo OpenFlow compartibles	1.3	1.0 - 1.3	1.0 - 1.3
Tipo de licencia	Libre	Libre para usuarios registrados	Libre
Plataformas	Mac OS , Unix y Windows	Mac OS , Linux y Windows	Windows, Mac y Linux
Compatibilidad con controladores	NOX, POX ,Floodlight y OpendayLight	NOX,POX,Beacon ,Floodlight	NOX, POX, Floodlight,Beacon, OpendayLight

La plataforma MININET es un simulador para el despliegue de redes SDN que permite instalarse sobre una máquina virtual con el sistema operativo Linux. Brinda la opción para acceder a controladores, armar topologías, etc [39].

3.6 Topología de solución

En base a los requerimientos de cada plano y las elecciones hechas en las secciones anteriores se muestra la topología de la solución propuesta en el simulador MININET en la figura 22.

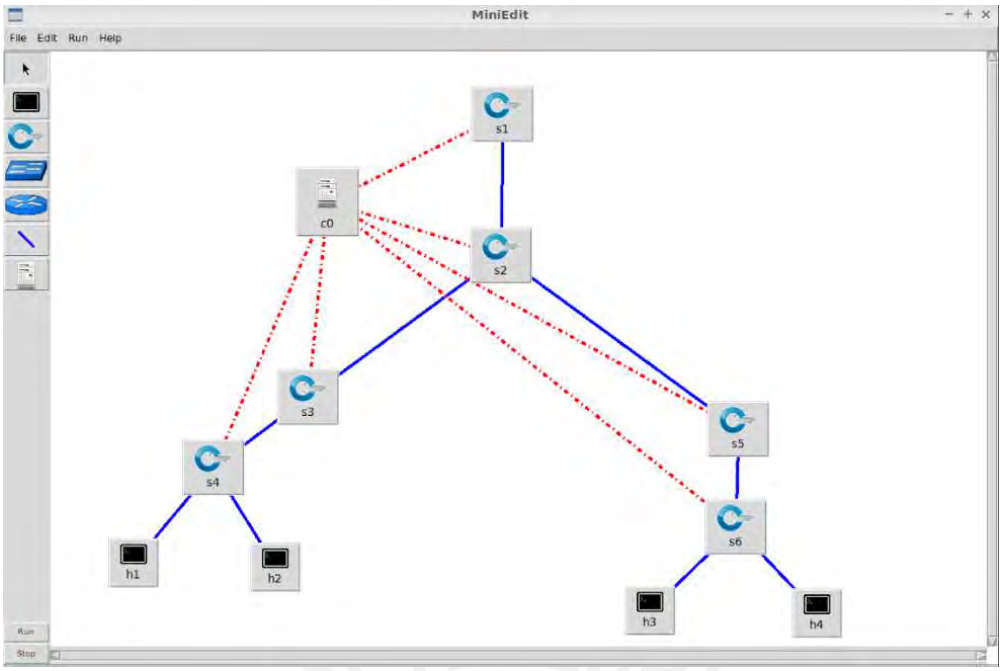

Figura 22: Topología de la solución en MININET
Fuente: Elaboración propia (2020)

En cada sede se colocarán 2 *switchs* SDN: uno que tendrá la funcionalidad de *router* que se conectará a la sede central y el segundo que tendrá la función de *switch* que tendrá conectados a sus puertos los usuarios finales (se simulan como computadoras en la imagen).

3.7 Desglose de Costos

En la tabla 4 y 5, se puede apreciar el desglose del OPEX y CAPEX asociado a los requerimientos necesarios mencionados en las secciones de selección de componentes. Se incluye la memoria de 32 GB, ya que se desconoce el almacenamiento actual del servidor en la red PUCP.

Tabla 5. Tabla de Capex

TABLA DE CAPEX			
Item	**Cantidad**	**Coste Unitario($)**	**Coste Total($)**
Switch Pica8 -P3297	6	3960	23760
Tarjeta DDR4 32GB	1	200	200
Controlador Floodlight	2	0	0
Ingeniero Senior	2	2500	5000
			28960

Fuente: Elaboración propia (2022)

Tabla 6. Tabla de Opex

TABLA DE OPEX			
Item	Cantidad	Coste Unitario($)	Coste Total($)
Servicio de Soporte	6	3960	23760
Ingeniero Senior	1	38520	38520
			62280

Fuente: Elaboración propia (2022)

En la tabla 5, se considera un ingeniero Senior para el despliegue completo de la topología con conocimientos sobre virtualización, *Networking* y *Redes Wireless.* En la tabla 6, se considera el servicio de soporte, ya que en los equipos actuales es relevante obtener un servicio de una empresa especializada en los equipos utilizados para un menor tiempo de inactividad en caso de algún evento de falla además de un ingeniero Senior para verificación de conectividad y trabajos de *Troubleshooting* en casos no urgentes de fallos.

4. PRUEBAS Y ANALISIS

En este capítulo, se presentarán las pruebas que se realizarán para obtener las métricas de calidad de servicio (QoS) de los parámetros de los distintos servicios que la red PUCP posee para finalmente compararlos en las topologías de *Legacy* PUCP y la propuesta solución SD-WAN PUCP. Ambas topologías realizadas en simuladores y utilizando únicamente 2 sedes externas: la escuela de música de la universidad y el centro de Idiomas Católica de Pueblo Libre.

Calidad de servicio es la habilidad de toda red para diferenciar y ofrecer prioridad a ciertos tipos de tráficos dependiendo de las necesidades que posea la red. Los principales parámetros que se medirán a continuación son retardo o latencia en cada servicio implementado y el ancho de banda entre las conexiones con los equipos de red en las sedes.

4.1 Modelo Legacy PUCP

Este escenario representa como se encuentra desplegada actualmente la red PUCP, se realizó en un entorno de simulación llamado GNS, la topología que se armó se muestra a continuación en la figura 23.

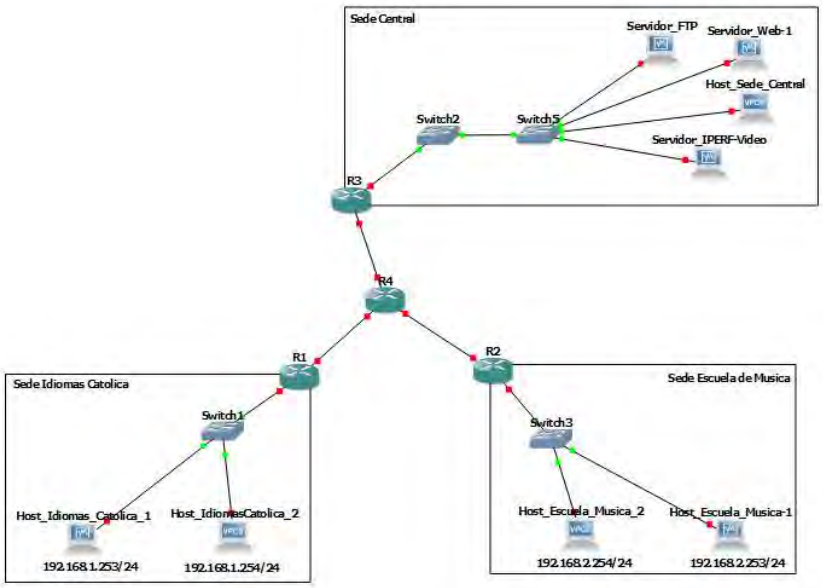

Figura 23: Topología LEGACY PUCP en GNS3
Fuente: Elaboración propia (2020)

La estructura de las sedes externas en la figura 23 consta de los siguientes equipos:

- Existen dos *hosts*, uno que es básico para funciones de ping o *tracert*, ya que solo se puede colocar direcciones IP, mascara y *default gateway*; y el otro que es una máquina virtual instalada en Oracle VM VitrtualBox con sistema operativo Ubuntu 20.04. Se muestra en la figura 24 el terminal del host básico.

Figura 24: Terminal Host básico – GNS3
Fuente: GNS3 (2020)

- Hay un *switch* de acceso que gestiona el tráfico de ambos hosts conectados a este.
- Finalmente, existe un *router* de borde que se encarga de brindar conectividad con la sede central.

La estructura de la sede central en la figura 23 consta de los siguientes equipos:

- Existen cuatro hosts, uno básico de la misma manera que en las sedes externas y los restantes son máquinas virtuales Ubuntu 20.04 que funcionaran como servidores de los diferentes servicios que se realizaran pruebas.
- Hay un switch2 que cumple la funcionalidad como capa *core*.
- Hay un switch5 que cumple la funcionalidad como capa de acceso que gestionara todo el tráfico de los *hosts* conectados a este.

En relación con las conexiones WAN, se usan los tres *routers* de borde de cada sede y el *router* central que representa la nube de internet, este tiene la función de interconectar las sedes por medio de túneles IPSEC VPN.

A continuación, se muestra la tabla 6 y 7 con los datos de cada enlace entre los equipos:

Tabla 7. Direcciones IP de host Legacy PUCP

Sede	Hostname	IP Address	Mascara	MAC Address	Default gateway
Sede Idiomas Catolica	Host_Idiomas_Catolica_1	192.168.1.253	24	08:00:27:2B:34:2A	192.168.1.1
	Host_Idiomas_Catolica_2	192.168.1.254	24	00:50:79:66:68:00	192.168.1.1
Sede Escuela de Musica	Host_Escuela_Musica_1	192.168.2.253	24	08:00:27:EC:CB:35	192.168.2.1
	Host_Escuela_Musica_2	192.168.2.254	24	00:50:79:66:68:01	192.168.2.1
Sede Central	Servidor_FTP	192.168.3.253	24	08:00:27:E8:0F:04	192.168.3.1
	Servidor_Web	192.168.3.252	24	08:00:27:49:31:AA	192.168.3.1
	Servidor_IPERF-VIDEO	192.168.3.251	24	08:00:27:90:77:FE	192.168.3.1
	Host_Sede_Central	192.168.3.254	24	00:50:79:66:68:02	192.168.3.1

Fuente: Elaboración propia (2020)

Tabla 8. Direcciones IP interfaces Routers

Sede	Hostname	Interfaz	Dirección IP	Mascara
Sede idiomas	R1	ethernet 1/1	10.10.10.1	30
		ethernet 1/0	192.168.1.1	24
Sede Escuela de	R2	ethernet 1/1	10.10.10.5	30
		ethernet 1/0	192.168.2.1	24
Sede central	R3	ethernet 1/1	10.10.10.9	30
		ethernet 1/0	192.168.3.1	24
WAN	R4	ethernet 1/1	10.10.10.6	30
		ethernet 1/0	10.10.10.2	30
		ethernet 1/2	10.10.10.10	30

Fuente: Elaboración propia (2020)

En relación con la seguridad del tráfico entre las conexiones con las sedes externas se usaron túneles VPN IPSEC que permite encriptar la información que viaja por el túnel y para la desencriptación se usan las llaves de autenticación. Las características de este túnel IPSEC se muestran a continuación:

- Confidencialidad: este parámetro permite garantizar la encriptación del tráfico que viaja por el túnel, el algoritmo que se utiliza es el *Advanced Encryption Standart* (AES) con un tamaño de llave de 256 bits.
- Integridad: este parámetro permite resguardar el tráfico para que no se pueda modificar. Para esta tarea se usa algoritmos de hashing, en este caso particular se usó el *Secure Hash Algorithm* (SHA 1).
- Autenticación: este parámetro realiza una validación de ambos nodos por medio de un intercambio de las contraseñas configuradas previamente. En este caso particular se usó como llave en ambos túneles cisco123.

4.2 Modelo SD-WAN PUCP

Este escenario representa la propuesta solución de esta tesis, se realizó en un entorno de simulación llamado MININET sobre máquinas virtuales Ubuntu 20.04. Una maquina donde se instaló el controlador Floodlight y la otra maquina donde se instaló el entorno de MININET. La topología propuesta se muestra a continuación en la figura 25.

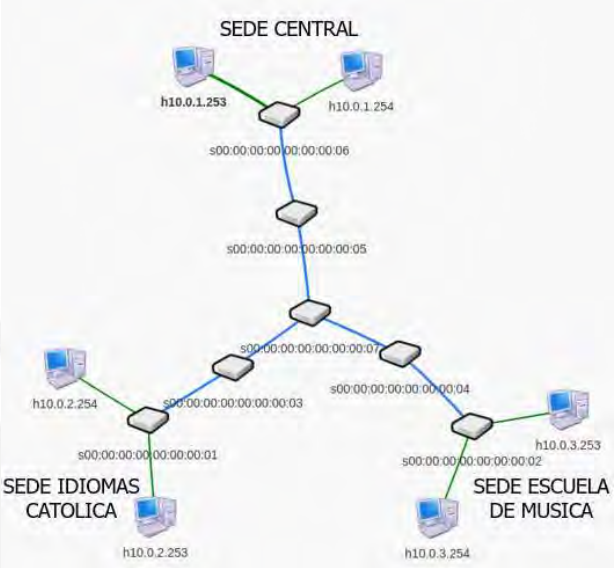

Figura 25: Topología SD-WAN PUCP en MININET
Fuente: Elaboración propia (2020)

La estructura de las sedes en la figura 25 consta de los siguientes equipos:

- Existen 2 host que son máquinas virtuales a los que se les puede controlar por medio de un terminal como se puede observar en la figura 26.

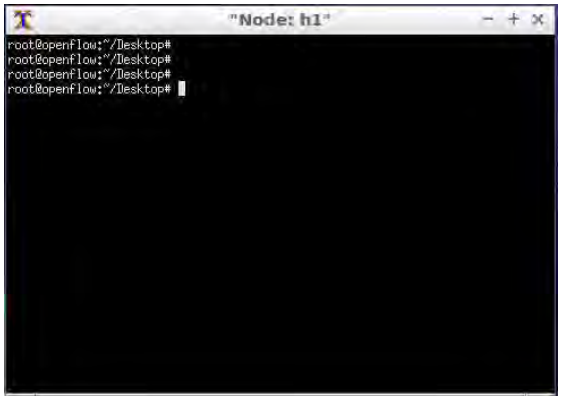

Figura 26: Terminal de host en MININET
Fuente: Elaboración propia (2020)

- Existe un *switch* que cumple la función de capa de acceso el cual gestiona el tráfico de los *hosts* conectados a este
- El segundo *switch* cumple la función de *router* de borde, la funcionalidad la designa el controlador.
- Existe un *switch* central que cumple la función de permitir la interconectividad entre las sedes.
- Existe un controlador que posee una visibilidad completa de la red que se encuentra ubicado en otra máquina virtual.

A continuación, se muestra la tabla 9 y 10 con la información de las direcciones IP de cada host y los enlaces entre cada *switch*.

Tabla 9. Direcciones IP interfaces Routers

Sede	Hostname	IP Address	Mascara	MAC Address	Default gateway
Sede Idiomas Catolica	Host_Idiomas_Catolica_1	10.0.2.253	24	FE:0D:DD:AA:F2:84	ethernet 0
	Host_Idiomas_Catolica_2	10.0.2.254	24	22:79:2E:CD:71:5A	ethernet 0
Sede Escuela de Musica	Host_Escuela_Musica_1	10.0.3.253	24	56:13:C5:FF:14:1B	ethernet 0
	Host_Escuela_Musica_2	10.0.3.254	24	6E:02:22:F7:8D:32	ethernet 0
Sede Central	Host_Central_1	10.0.1.253	24	A2:A7:C8:23:F8:86	ethernet 0
	Host_Central_2	10.0.1.254	24	8A:B5:29:1F:BA:A0	ethernet 0
Controlador Floodlight	Controlador Floodlight	192.168.0.17	24	NO	NO

Fuente: Elaboración propia (2020)

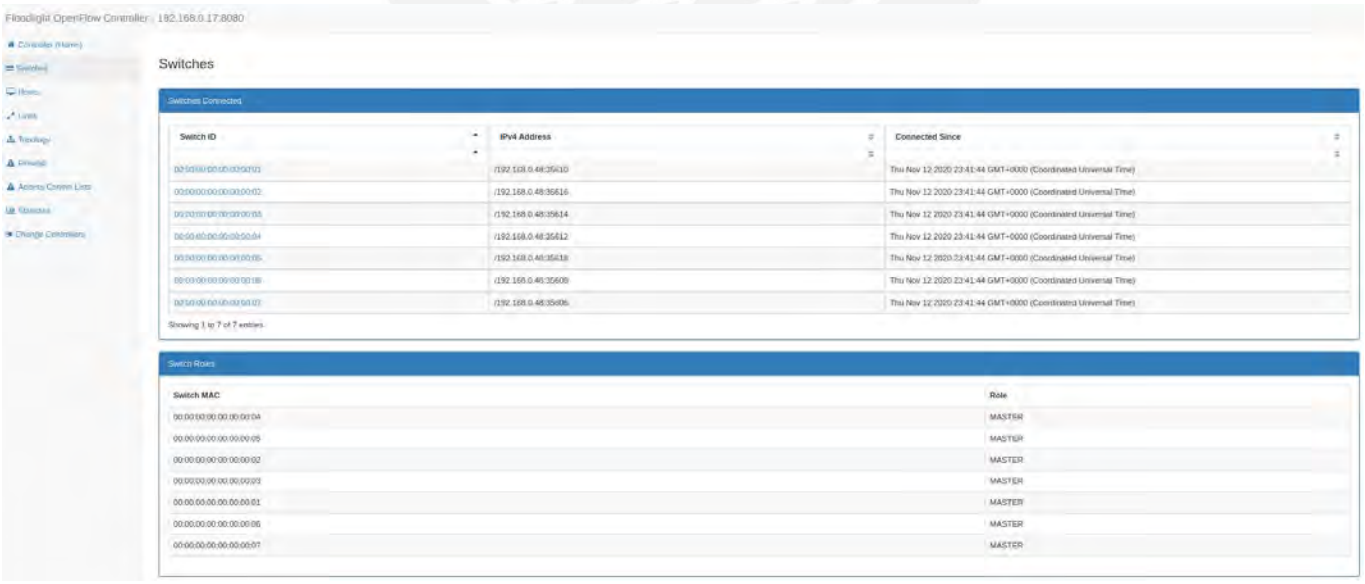

Figura 27: Dashboard de Floodlight
Fuente: MININET (2020)

Tabla 10. Enlaces entre switchs

Direction	Puerto Origen	Switch Origen	Puerto Destino	Switch Destino
bidireccional	3	00:00:00:00:00:00:00:01	1	00:00:00:00:00:00:00:03
bidireccional	1	00:00:00:00:00:00:00:02	2	00:00:00:00:00:00:00:04
bidireccional	2	00:00:00:00:00:00:00:03	1	00:00:00:00:00:00:00:07
bidireccional	1	00:00:00:00:00:00:00:04	3	00:00:00:00:00:00:00:07
bidireccional	1	00:00:00:00:00:00:00:05	1	00:00:00:00:00:00:00:06
bidireccional	2	00:00:00:00:00:00:00:06	2	00:00:00:00:00:00:00:07

Fuente: Elaboración propia (2020)

4.3 Pruebas de servicios

Se realizaron pruebas de servicio con diferentes tipos de tráficos para realizar comparaciones de métricas de telemetría en ambos escenarios. A continuación, se muestran cuáles son los servicios que se realizaran las pruebas.

- Validación VPN
- Servidor IPERF
- Servidor WEB
- Servidor FTP

4.3.1 Pruebas de servicios modelo LEGACY PUCP

En este escenario se utilizará los *hosts* que se encuentra en la sede central como servidores previamente configurados para que funcionen como tal. A continuación, se realizarán las pruebas desde un host de cada sede para obtener métricas de tiempo de descarga y ancho de banda en una arquitectura tradicional.

4.3.1.1 Validación VPN

En esta prueba se usará la herramienta Wireshark y para capturar los paquetes ICMP que se enviaran de hosts en diferentes sedes para verificar la encriptación de tráfico. Se tomará como ejemplo el túnel VPN creado entre la Sede Idiomas Católica y la sede Central, ya que en relación con la sede de la escuela de música se sigue el mismo procedimiento.

En este caso se realizó un ping hacia la dirección IP del Host_Sede_Cental desde el Host_Idiomas_Catolica_2 se realizaron capturas en 3 puntos importantes en la ruta:

i) Host_Idiomas_Catolica_2 – *Switch* 1: en la figura 28, se puede observar que hay tráfico de *request* por parte del origen y response por parte del destino por lo tanto existe conectividad, como se puede observar en la parte inferior de la figura, la dirección IP de origen y la dirección IP destino y la data que se envía que no se encuentra encriptado aún.

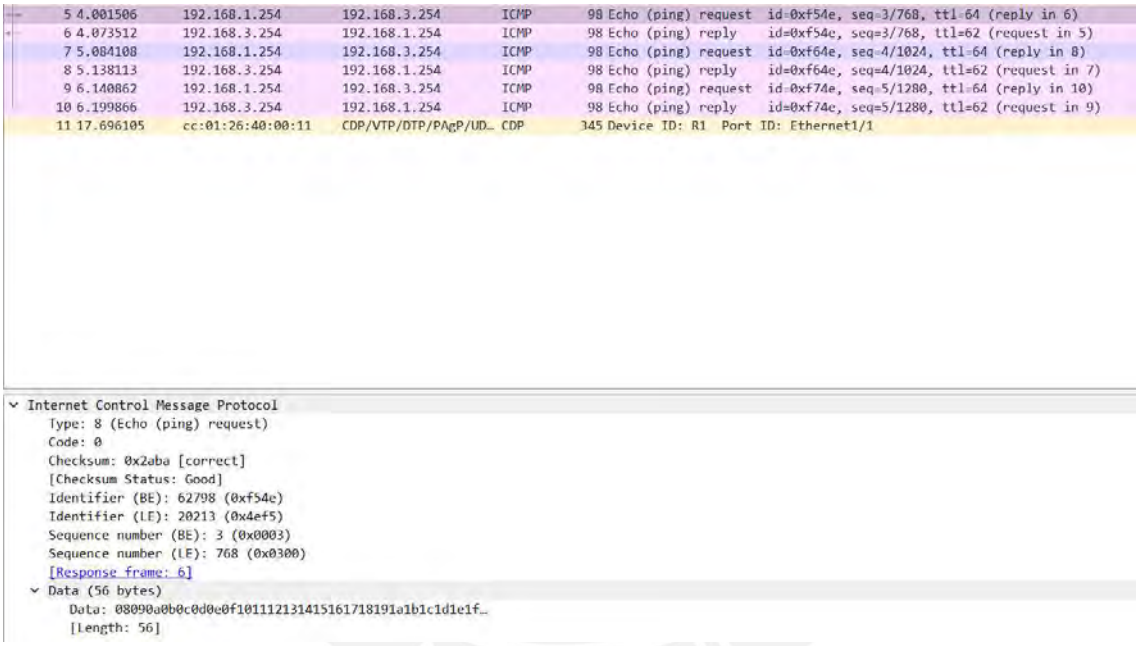

Figura 28: Captura de paquete PING
Fuente: Wireshark (2020)

ii) Router R1 – Router R4: en la figura 29 se puede observar los primeros paquetes que se capturan (de color azul) son mensajes de establecimiento del túnel VPN IPSEC, en esta parte se realiza el intercambio de las llaves. Y en la figura 30, se observa que los paquetes tienen como origen el Router R1 y como destino el Router R3 sin usar los enlaces de R4, es decir se envía todo el tráfico a través del túnel por lo que no se puede observar la data, pero se muestra el parámetro SPI

que se usa para la desencriptación de la data en el receptor (R3) como en la figura 28.

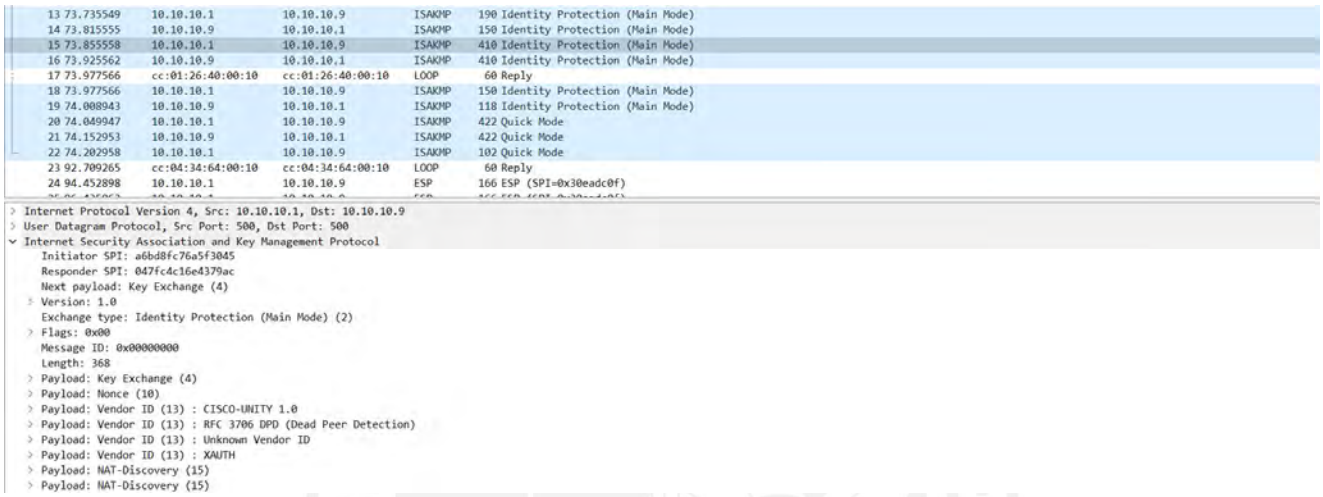

Figura 29: Estabilización de túnel VPN
Fuente: Wireshark (2020)

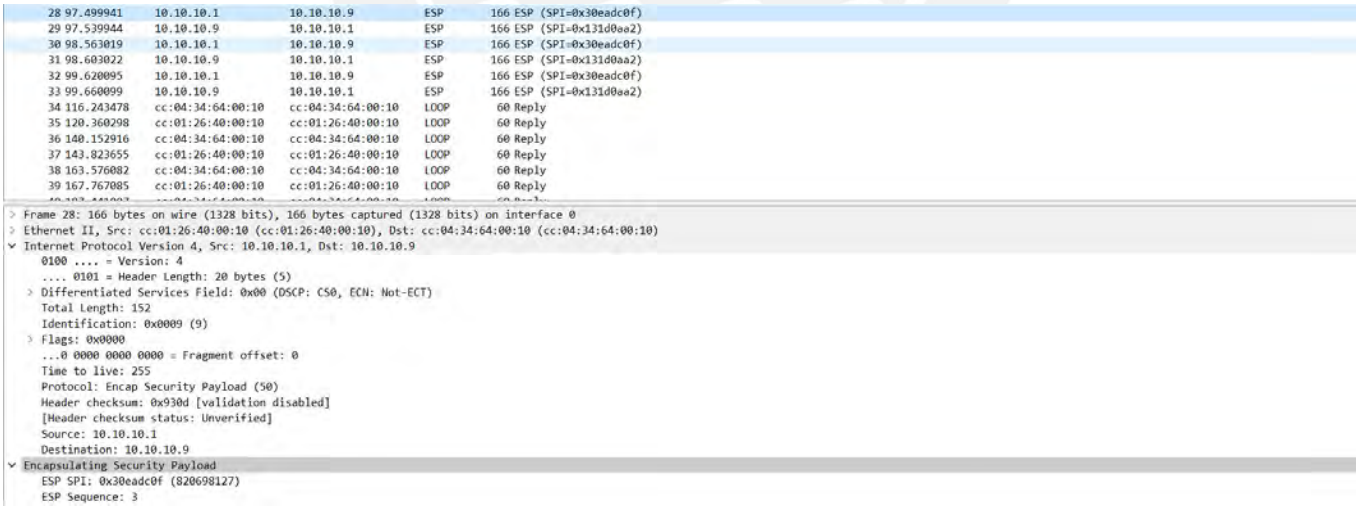

Figura 30: Trafico encriptado por el túnel VPN
Fuente: Wireshark (2020)

iii) Switch5 – Host_Sede_Central: en la figura 31, se observa que los paquetes llegan al destino desencriptados por el *router* R3, ya que se puede visualizar la data.

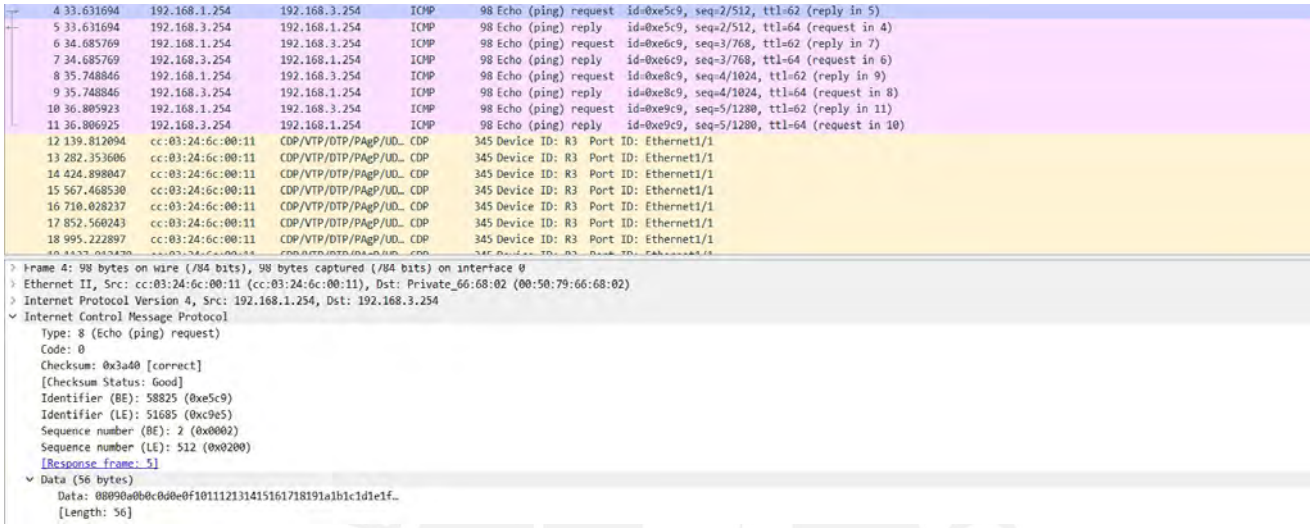

Figura 31: Captura de paquete de llegada PING
Fuente: Wireshark (2020)

4.3.1.2 Servicio IPERF

IPERF es un servicio de tipo cliente – servidor que permite conocer el ancho de banda entre las conexiones de dos equipos que poseen conectividad entre sí. A continuación, se realizará las mediciones de los anchos de banda en cada host colocado en modo cliente apuntando de cada modelo que se presentaron inicialmente.

i) Host_Idiomas_Catolica_1

En la figura 32 se observa el modo servidor que se coloca al host en la sede central y en la figura 33 se observa la comunicación que se crea entre estas dos computadoras por medio del puerto 5001 desde la dirección IP 192.168.1.253.

```
gbartra@openflow:~$ iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 192.168.3.251 port 5001 connected with 192.168.2.253 port 40184
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0-10.8 sec  9.88 MBytes  7.69 Mbits/sec
[  4] local 192.168.3.251 port 5001 connected with 192.168.1.253 port 52890
[  4]  0.0-10.7 sec  9.75 MBytes  7.64 Mbits/sec
^Cgbartra@openflow:~$
```

Figura 32: Modo servidor IPERF - GNS3
Fuente: GNS3 (2020)

```
gbartra@openflow:~$ iperf -c 192.168.3.251
------------------------------------------------------------
Client connecting to 192.168.3.251, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.253 port 52890 connected with 192.168.3.251 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.2 sec  9.75 MBytes  8.02 Mbits/sec
gbartra@openflow:~$
gbartra@openflow:~$
```

Figura 33: Modo cliente IPERF – HOST Idiomas Católica
Fuente: GNS3 (2020)

ii) Host_Escuela_Musica_1

En la figura 34, se muestra el host en la sede de escuela de música en modo cliente apuntando a la IP del servidor en la sede central y en la figura 32 se puede observar cómo se crea la comunicación por el puerto 5001 desde la dirección 192.168.2.253.

```
gbartra@openflow:~$ iperf -c 192.168.3.251
------------------------------------------------------------
Client connecting to 192.168.3.251, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 192.168.2.253 port 40184 connected with 192.168.3.251 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.1 sec  9.88 MBytes  8.18 Mbits/sec
gbartra@openflow:~$
```

Figura 34: Modo cliente IPERF – HOST Escuela de música
Fuente: GNS3 (2020)

4.3.1.3 Servicio FTP

En esta prueba se configuro el host en la sede central como un servidor FTP con la finalidad que los hosts en las sedes externas puedan acceder a este y descargar los documentos que sean necesarios. Se uso un documento de bloc de notas de un peso de 983 bytes llamado prueba para realizar el proceso de descarga.

En este escenario se usa el host llamado Servidor_FTP de la topología con una IP de 192.168.3.253.

i) Host_Idiomas_Catolica_1

En la figura 35, se establece la comunicación con el servidor FTP desde el host de la sede de idiomas católica, se realizó el proceso de descarga, por medio del uso del comando GET, de forma exitosa como se puede observar en la figura 36 y se usó la herramienta Wireshark para capturar los paquetes del envió del documento para obtener el tiempo que tardo como se puede observar en la figura 37.

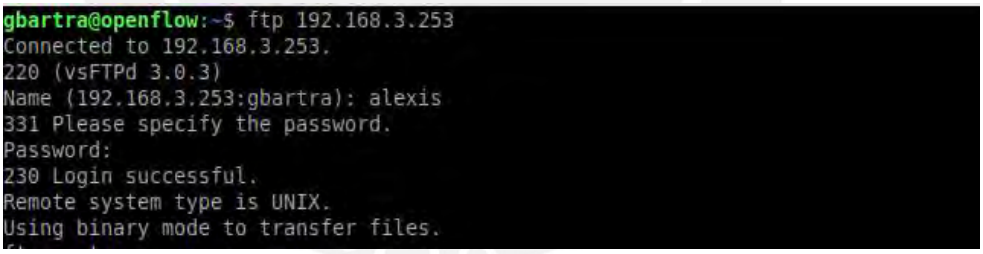

Figura 35: Conexión exitosa – Server FTP – Idiomas Católica
Fuente: GNS3 (2020)

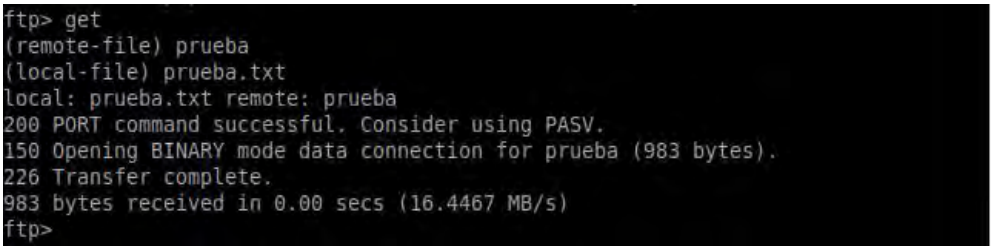

Figura 36: Transferencia de archivo – Server FTP – Idiomas Católica
Fuente: GNS3 (2020)

En relación con el cálculo del tiempo estimado de descarga se tomará el tiempo final que se muestra en la figura 37 en el paquete número 65 y se restará con el tiempo del paquete número 49 en la figura 38. El tiempo de descarga obtenido es 0.25915 segundos.

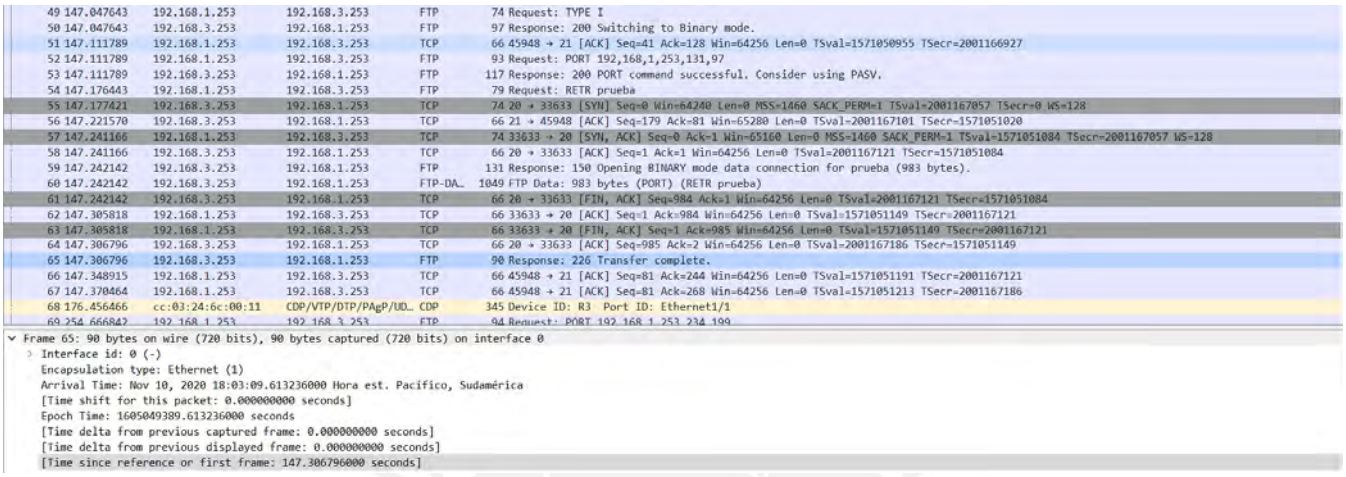

Figura 37: Paquete Final de transferencia– Server FTP – Idiomas Católica
Fuente: Wireshark (2020)

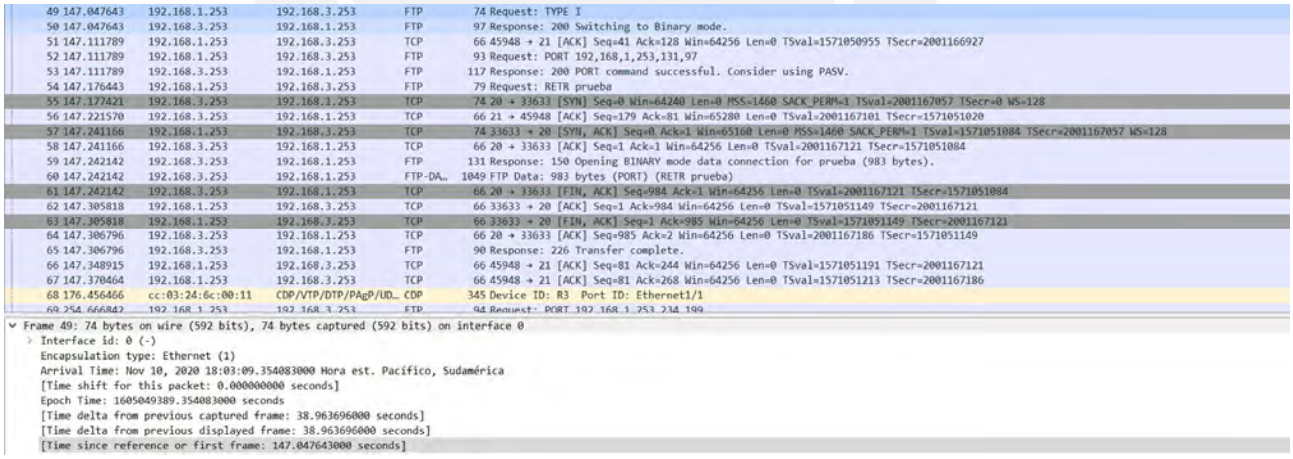

Figura 38: Paquete Inicial de transferencia– Server FTP – Idiomas Católica
Fuente: Wireshark (2020)

ii) Host_Escuela_Musica_1

En la figura 39, se establece la conexión exitosa con el servidor FTP desde la sede de la escuela de música. De la misma forma se usó la herramienta Wireshark para capturar los paquetes para determinar el tiempo de demora en la descarga en la figura 40 y 41.

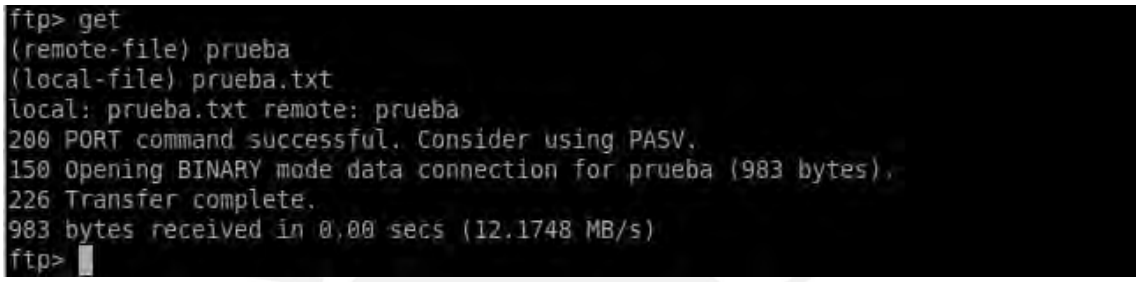

Figura 39: Transferencia de archivo – Server FTP – Escuela de música
Fuente: Wireshark (2020)

De la misma manera, para el cálculo del tiempo de descarga se usa el paquete número 184 de la figura 40 y el paquete 168 de la figura 41. Por lo tanto, el tiempo calculado es 0.25958 segundos.

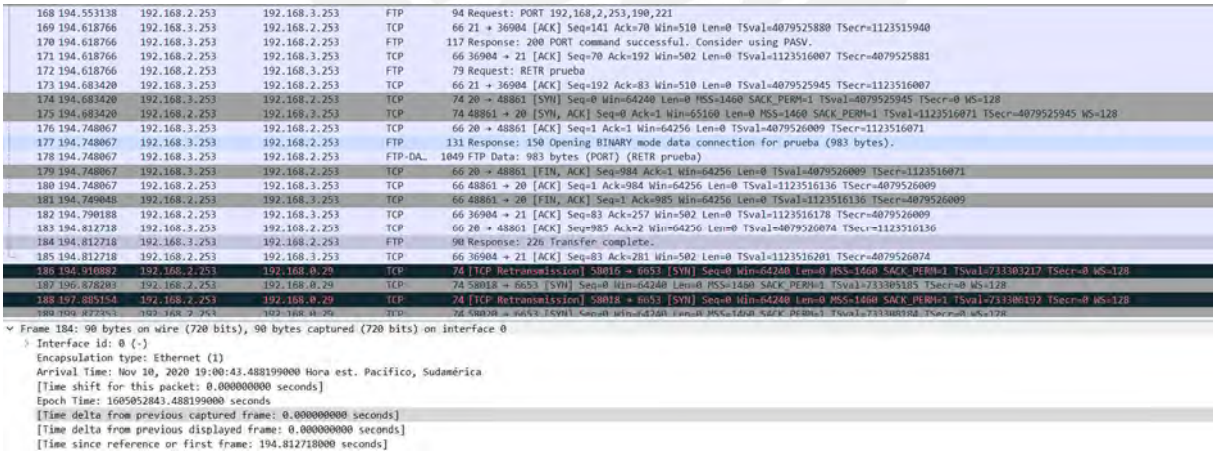

Figura 40: Paquete Final de transferencia – Server FTP – Escuela de música
Fuente: Wireshark (2020)

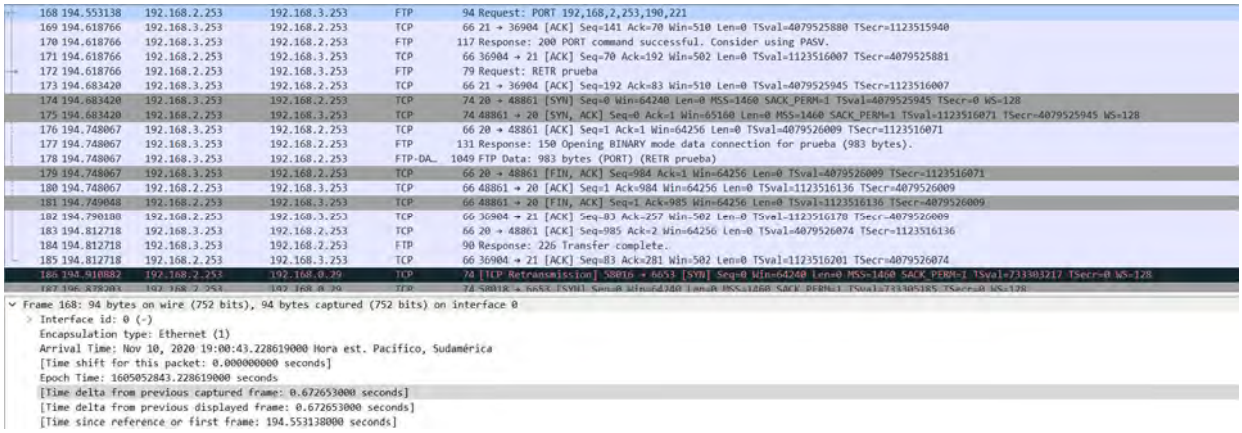

Figura 41: Paquete Inicial de transferencia – Server FTP – Escuela de música
Fuente: Wireshark (2020)

4.3.1.4 Servicio WEB

Esta prueba se realizó configurando previamente el host WEB SERVER con IP 192.168.3.252 en la sede central, este servidor cumple la función de guardar páginas WEB para que las sedes externas puedan descargar y usar dicho contenido en este caso de prueba se creó una página en la máquina virtual WEB SERVER.

i) Host_Idiomas_Catolica_1

En la figura 42, se puede observar la página web descargada exitosamente desde la sede central, ya que la IP que aparece en la sección de URL es la del WEB SERVER.

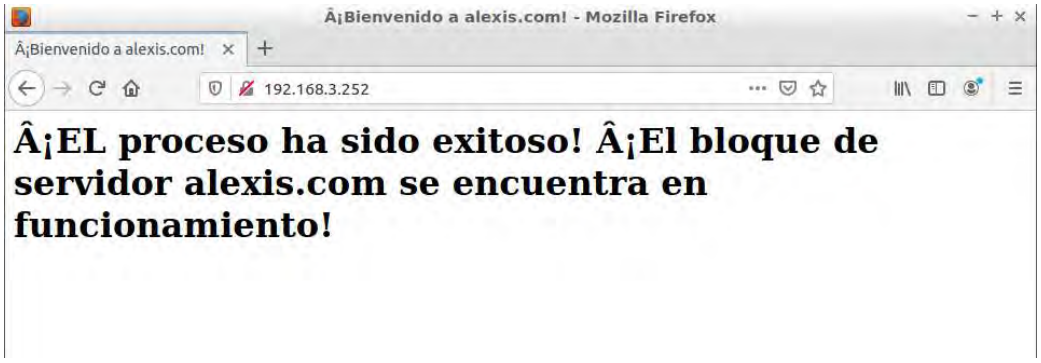

Figura 42: Pagina Web creada – Server WEB – Idiomas Católica
Fuente: GNS3 (2020)

Con el objetivo de obtener la métrica de tiempo de descarga de la página WEB desde la sede central se usó la herramienta Wireshark como se puede observar en la figura 43. El tiempo obtenido es de 5.3 segundos del paquete número 139, ya que este es el último de dicha comunicación.

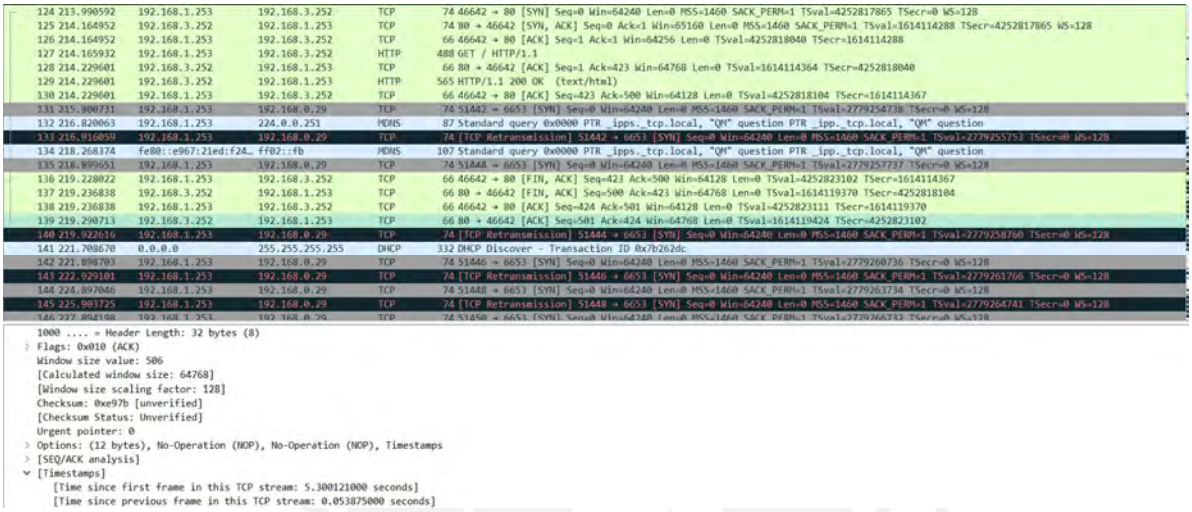

Figura 43: Comunicación – Server WEB – Idiomas Católica
Fuente: Wireshark (2020)

ii) Host_Escuela_Musica_1

La conexión con la página web creada es igual que en la figura 42. De forma similar se usó la herramienta Wireshark para obtener métricas de tiempo de descarga como se puede observar en la figura 44. Para el cálculo del tiempo se necesitaría restar el tiempo de del paquete número 87 en la figura 45 y el tiempo del paquete número 69. Por lo tanto, el tiempo de descarga es 5.061 segundos.

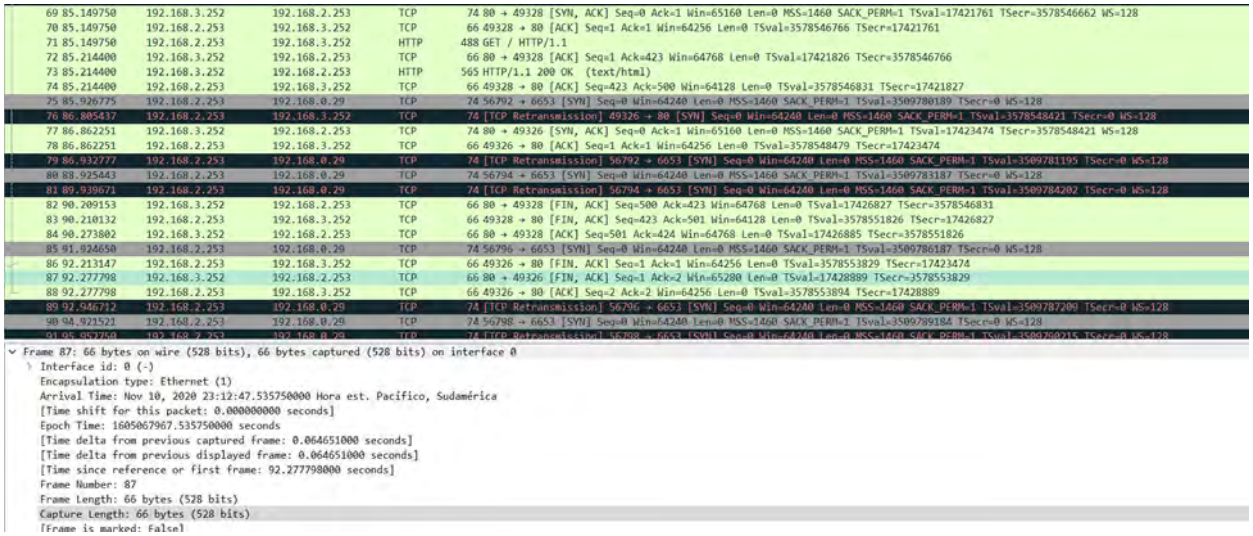

Figura 44: Comunicación – Server WEB – Escuela de música
Fuente: Wireshark (2020)

4.3.2 Pruebas de servicios modelo SD-WAN PUCP

En este escenario, de la misma manera que en el otro modelo, se colocaron los *hosts* en las sedes externas como clientes el host_Central_1 como servidor.

4.3.2.1 Servidor IPERF

De la misma manera que en el modelo LEGACY, se medirá el ancho de banda de las conexiones entre las sedes externas con la sede central, la dirección IP del servidor es la del Host_Central_2.

i) Host_idiomas_Catolica_2

En la figura 45, se puede observar que se coloca al host Central como servidor, mientras que en la figura 46 se coloca al host de idiomas católica como cliente apuntando hacia la IP del servidor.

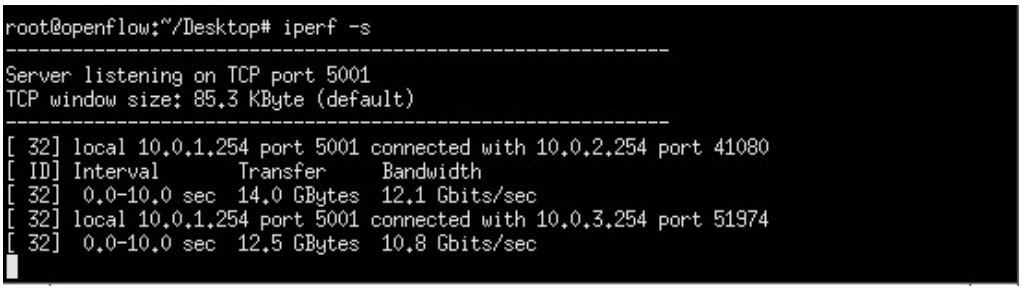

Figura 45: Modo servidor IPERF - MININET
Fuente: MININET (2020)

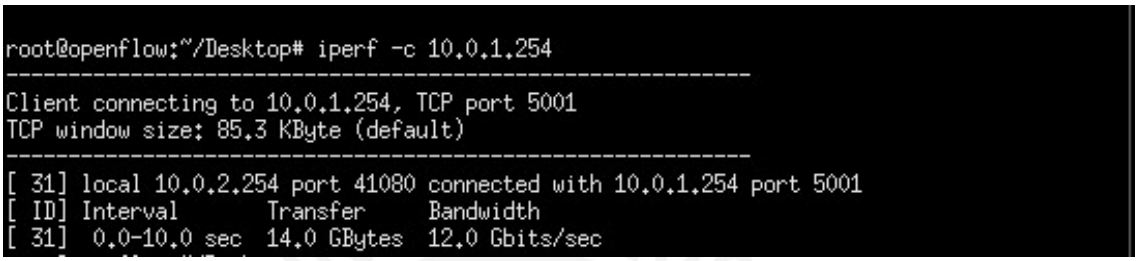

Figura 46: Modo cliente - IPERF - HOST Idiomas Católica
Fuente: MININET (2020)

ii) Hos_Escuela_Musica_2

En la figura 47, se puede observar colocar el host en modo cliente apuntando hacia la IP del servidor.

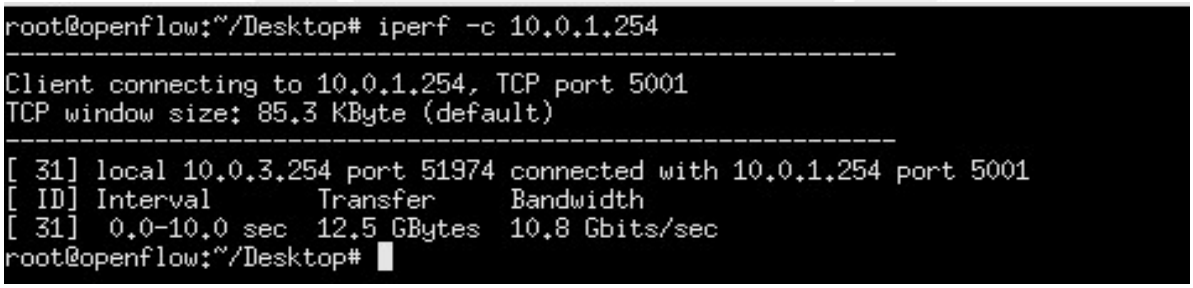

Figura 47: Modo cliente - IPERF - HOST Escuela música
Fuente: MININET (2020)

4.3.3 Servicio FTP

Esta prueba cumple la misma funcionalidad como se explicó en el modelo anterior, de la misma manera se usará un documento de bloc de notas de un peso de 983 bytes llamado prueba para realizar el proceso de descarga.

i) Host_idiomas_Catolica_2

En la figura 48, se puede observar colocar al host 1 (Sede central) como servidor con el comando “INETD” y el host 3 (Sede idiomas católica) conectarse al servidor FTP para solicitar con GET el documento de prueba.

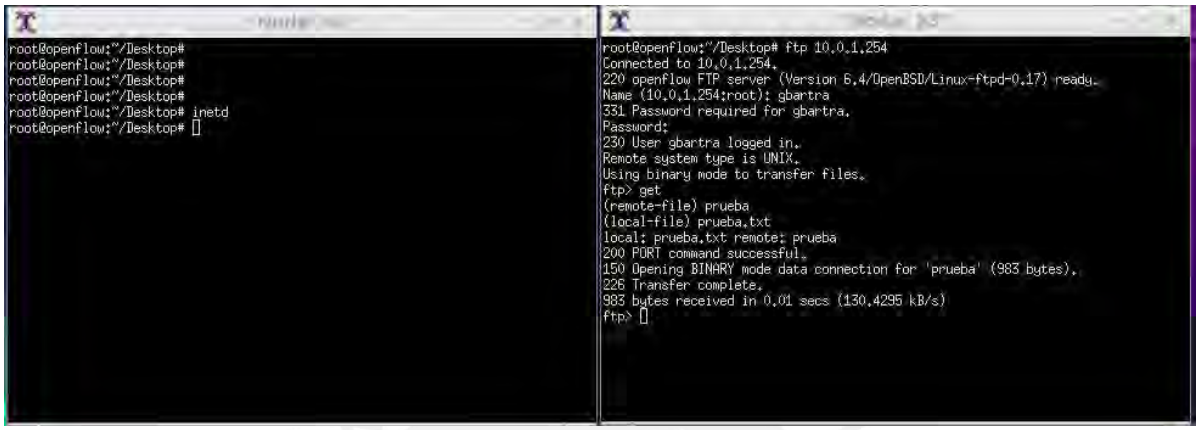

Figura 48: Conectividad servidor FTP – HOST Idiomas católica
Fuente: MININET (2020)

Para el cálculo de la métrica del tiempo de descarga desde el servidor se usará la herramienta Wireshark. A continuación, se muestran las capturas del último paquete y el primer paquete de la comunicación en la figura 48 y 49. Por lo tanto, el tiempo de descarga del archivo de prueba en esta sede es 0.069 segundos.

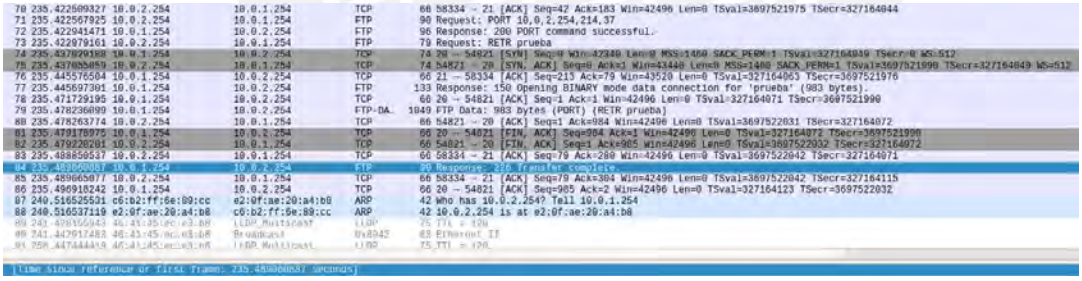

Figura 49: Paquete Final - FTP - HOST Idiomas Católica
Fuente: MININET (2020)

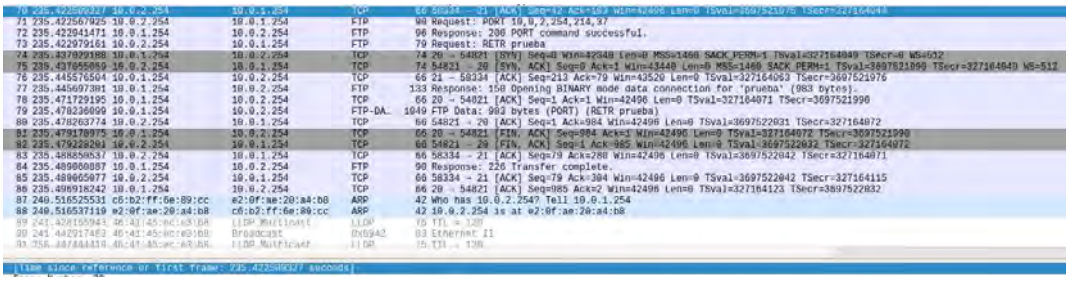

Figura 50: Paquete Inicial - FTP - HOST Idiomas católica
Fuente: MININET (2020)

ii) Host_Escuela_Musica_2

De forma semejante se realizará la misma prueba en la sede de la escuela de música. En la figura 50, se puede observar la comunicación entre el servidor – cliente y la solicitud GET para descargar el documento de prueba.

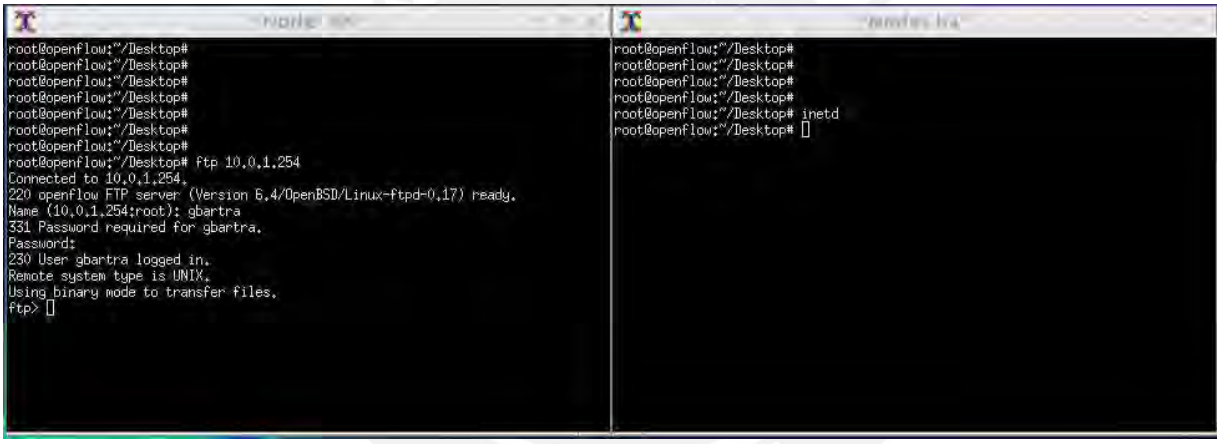

Figura 51: Conectividad servidor FTP – HOST Escuela música
Fuente: MININET (2020)

Para el cálculo del tiempo de descargar se usará las figuras 51 y 52 que se muestran a continuación. Por lo tanto, el tiempo de descarga del archivo de prueba en esta sede es de 0.061 segundos.

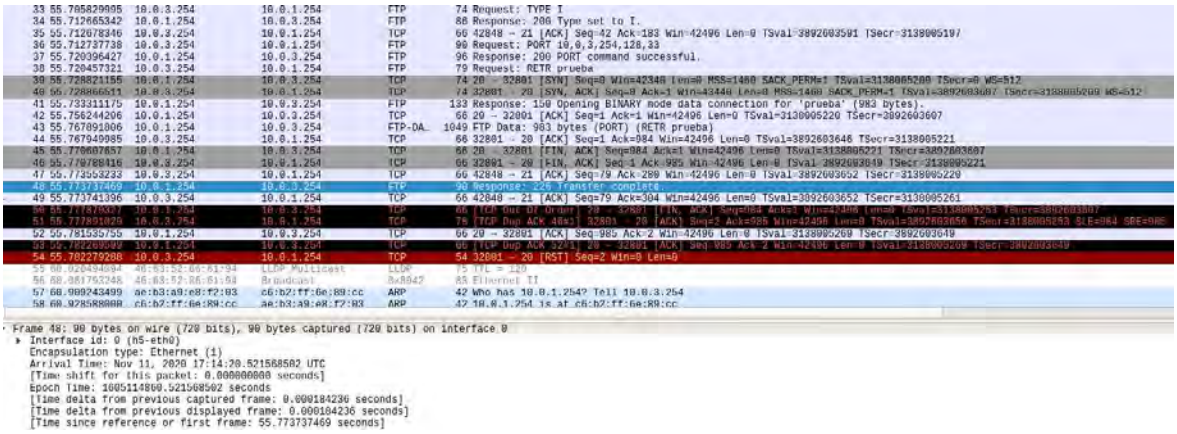

Figura 52: Paquete Final - FTP - HOST Escuela música
Fuente: MININET (2020)

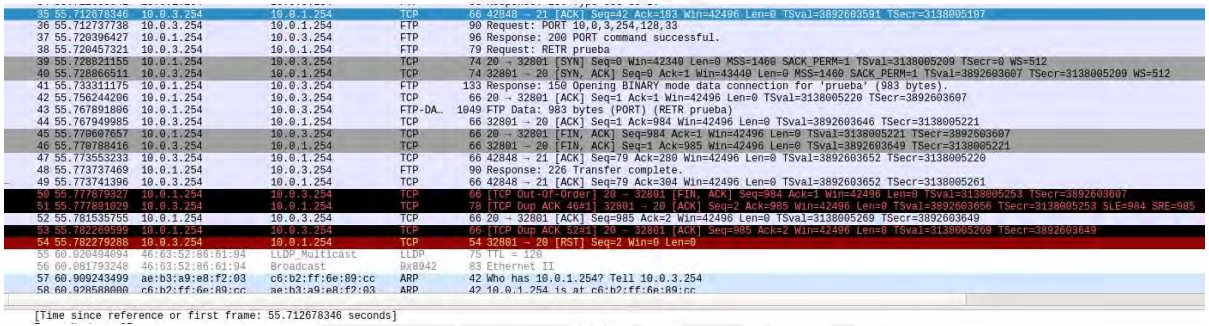

Figura 53: Paquete Inicial - FTP - HOST Escuela música
Fuente: MININET (2020)

4.3.4 Servicio WEB

Esta prueba se realizó configurando el servidor WEB en la sede central con dirección IP 10.0.1.254 y las sedes externas serán los clientes. A continuación, se muestras las pruebas en ambas sedes.

i) Sede_Idiomas_Catolica_2

En la figura 53, se puede observar la comunicación del host 3 (sede idiomas católica) con el servidor WEB en la sede central y la solicitud GET para la descarga de la página WEB.

Figura 54: Comunicación Servidor WEB – Host Idiomas católica
Fuente: MININET (2020)

Para obtener le tiempo de descarga de la página WEB se usan las figuras 54 y 55 de las capturas de paquetes en Wireshark. Por lo tanto, el tiempo de descarga fue de 0.0409 segundos

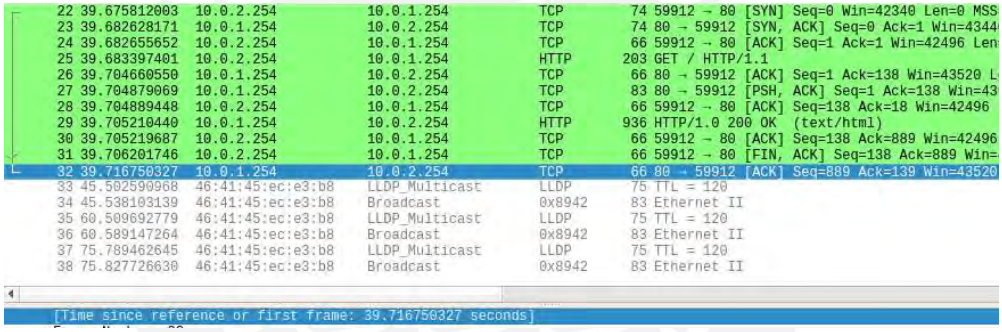

Figura 55: Paquete Final – Web Server – Host Idiomas Catolica
Fuente: MININET (2020)

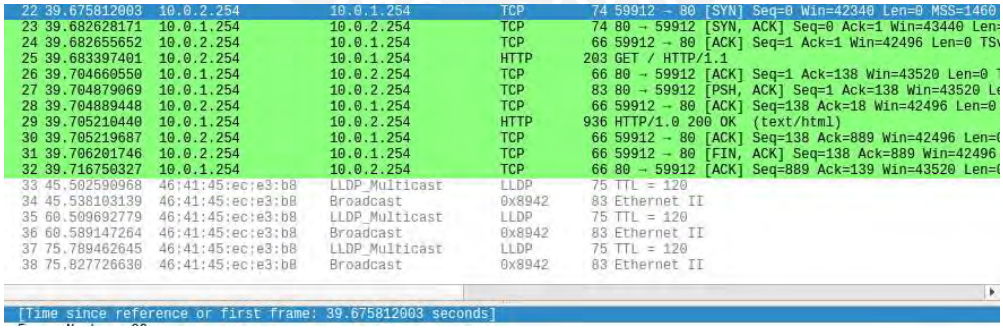

Figura 56: Paquete Inicial – Web Server – Host Idiomas Catolica
Fuente: MININET (2020)

ii) Sede_Escuela_Musica_2

Se adjunta las imágenes correspondientes, ya que el proceso es similar al caso de la sede anterior. El tiempo obtenido de descarga es 0.0558 segundos.

Figura 57: Comunicación Servidor WEB – Host Escuela música
Fuente: MININET (2020)

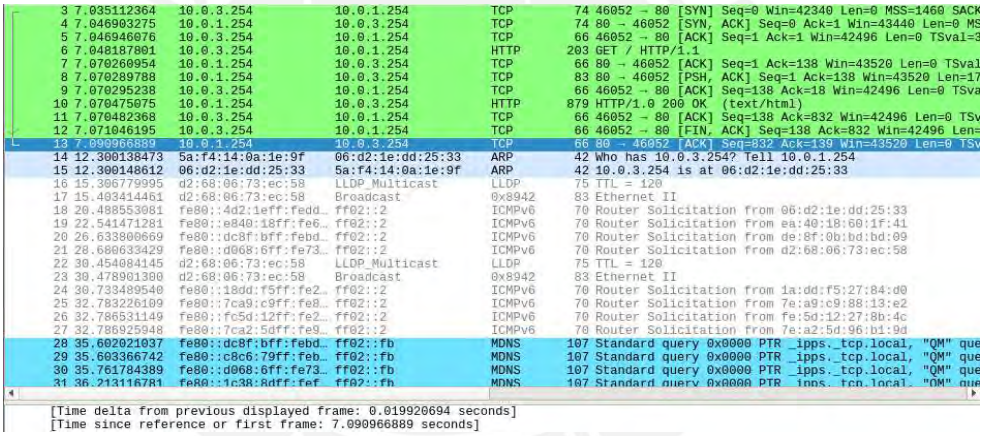

Figura 58: Paquete Final – Web Server – Host Escuela música
Fuente: MININET (2020)

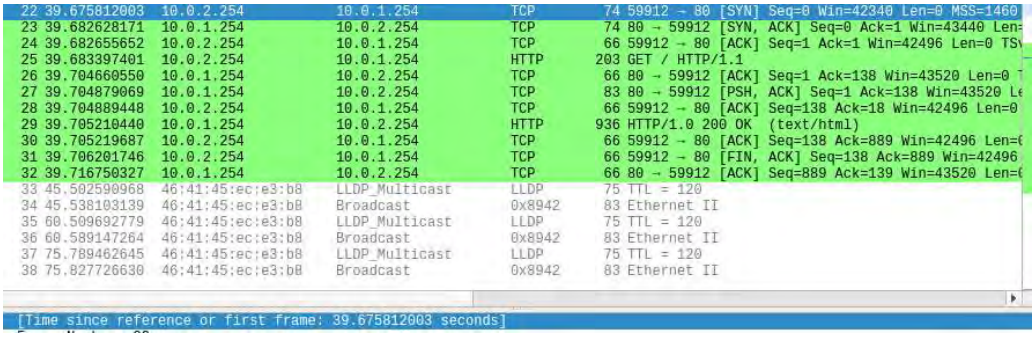

Figura 59: Paquete Inicial – Web Server – Host Escuela música
Fuente: MININET (2020)

4.4 Resultados de pruebas

En base a las medidas de telemetría realizadas en las pruebas en la sección anterior se muestra a continuación una tabla que muestra la comparación entre ambos modelos.

Tabla 11. Resultados de pruebas

	Servidor IPERF	Servidor Web (segundos)	SERVIDOR FTP (segundos)
IC - GNS3	8.02 Mbit/s	5.3	0.25915
EM - GNS3	8.18 Mbit/s	5.061	0.25958
IC - MININET	12.0 Gbit/s	0.0409	0.069
EM - MININET	10.9 Gbit/s	0.0558	0.061

Fuente: Elaboración propia (2020)

Como se observa en la tabla 9 los hosts en el modelo de MININET se encuentran resaltados, ya que los valores son mucho menores en relación con los del modelo LEGACY.

4.5 Despliegue de nuevo servicio

Se desplegará el servicio de video *streaming,* este servicio permitirá tráfico de video y voz en tiempo real entre las sedes con una latencia baja por el uso de la tecnología SD WAN. Para este despliegue se usará en la plataforma VLC media player que es un reproductor de código abierto, se utilizó el host en la sede central como *streamer* y los receptores las sedes externas. A continuación, se muestra la prueba de la transmisión *streaming* hacia la sede de idiomas católica en ambos modelos para mostrar tiempos de latencia. Se uso un video de prueba descargado de internet llamado ICON_VERSION8_1 de un peso de 43.9 MB.

4.5.1 Streaming en modelo LEGACY

En la figura 59 y 60, se puede observar el paquete final de la trasmisión y el paquete inicial de la misma enviados hacia el host de idiomas católica. Con el fin de obtener el tiempo que duro la transmisión se usara el tiempo obtenido de los paquetes en dichas figuras. Por lo tanto, el tiempo de transmisión obtenido es de 20.5078 segundos.

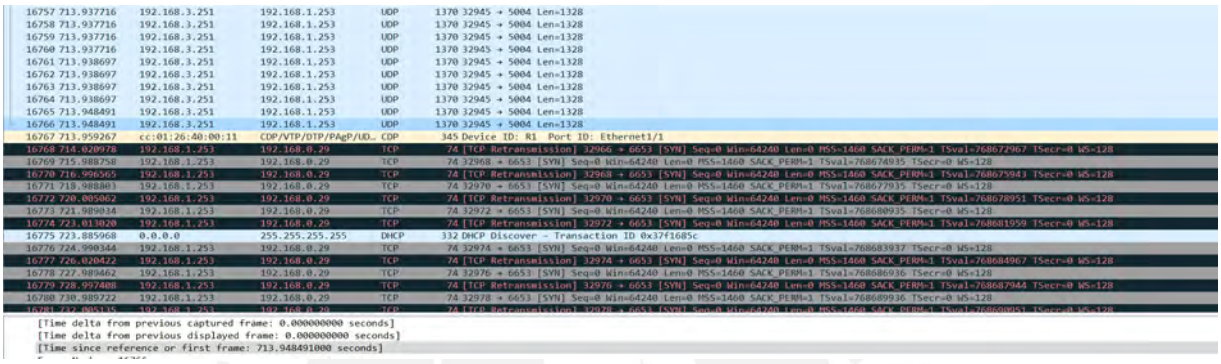

Figura 60: Paquete Final - Streaming - Host Idiomas Católica - GNS3
Fuente: Wireshark (2020)

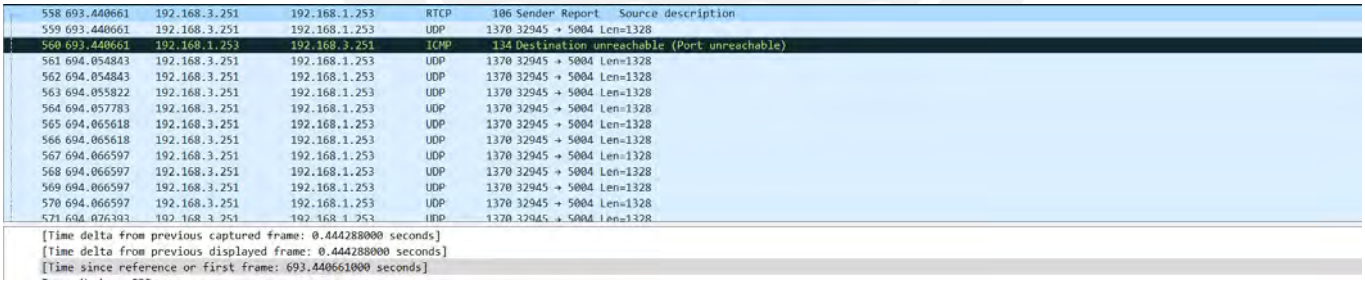

Figura 61: Paquete Inicial - Streaming - Host Idiomas Católica - GNS3
Fuente: MININET (2020)

4.5.2 Streaming en modelo SD-WAN

En la figura 61 y 62, se puede observar los paquetes final e inicial para obtener el mismo tiempo de transmisión. Por lo tanto, el tiempo de transmisión obtenido es de 19.22 segundos.

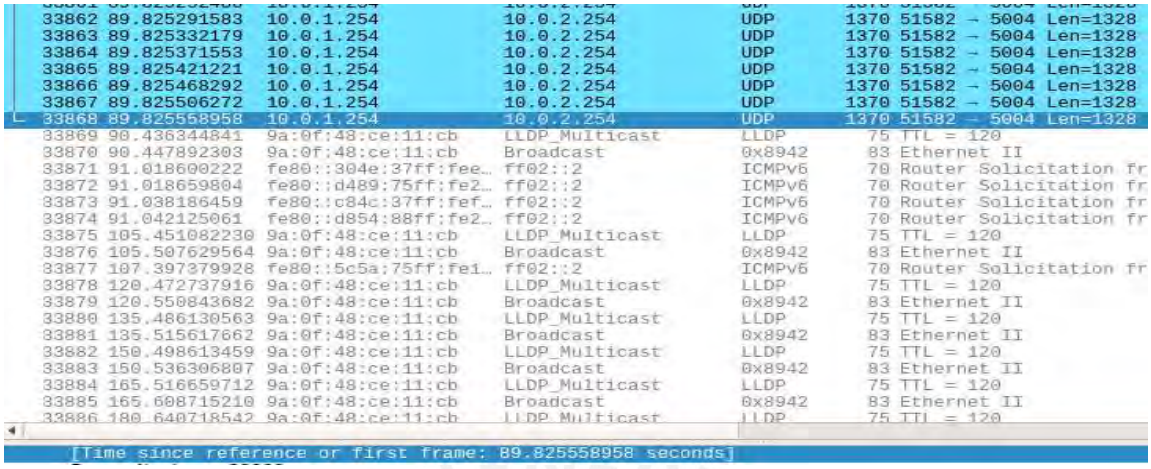

Figura 62: Paquete Inicial - Streaming - Host Idiomas Católica - MININET
Fuente: MININET (2020)

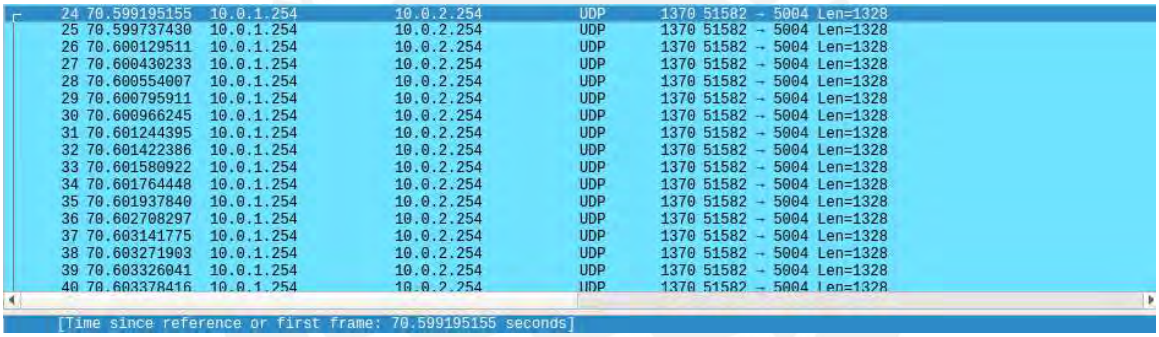

Figura 63: Paquete Inicial - Streaming - Host Idiomas Católica - MININET
Fuente: MININET (2020)

CONCLUSIONES

- Se logró realizar el objetivo principal que es el diseño de la red PUCP usando la tecnología SDN, la cual se denominó SD - WAN PUCP sobre un entorno simulado llamado MININET.
- Se realizó los análisis de los servicios que posee la red PUCP midiendo las métricas de telemetría por medio de la herramienta Wireshark.
- El video Streaming es un nuevo servicio que permite la transmisión de datos de video en vivo, se aprovisionó en la red propuesta con un tiempo de latencia con un error muy bajo.
- Las redundancias serán realizadas por el *service provider* que se usa en la topología propuesta.
- El control masivo de todos los dispositivos de la red es por medio del uso del controlador Floodlight que posee comunicación con cada uno de estos.
- Se realizarón las comparaciones de las métricas de telemetría (QoS) en ambos modelos presentados en este documento en base a los diferentes servicios que se usan y se obtuvo tiempos de ejecuciones mucho menores en la topología propuesta como se puede observar en la tabla 11.

TRABAJO FUTURO

- Con el objetivo de obtener valores más exactos, ambas redes pueden ser simuladas con separación de VLAN's entre diferentes *Host* en una misma sede, protocolos de enrutamiento como OSPF y medir estadísticas de ñtelemetría en el tráfico dirigido a diferentes VLAN's en las otras sedes.
- Por motivos de la situación actual del país se realizaron las simulaciones en entornos virtualizados, se puede obtener valores de una red SD-WAN por medio del uso de los equipos PICA 8 del grupo GIRA en la universidad con el fin de contrastar con interfaces reales.
- Se puede extrapolar las mismas configuraciones, pero con el uso de otros controladores sobre el mismo entorno de simulación MININET con el fin de obtener una mejor robustez de la solución.
- En las simulaciones hechas en este trabajo solo se consideró un controlador en la red, pero por lo general existen más de uno con funciones diferentes para cada servicio y función de *backup* en caso fallas del controlador principal.
- Para el uso más específico de equipos CISCO de la solución SD-WAN se puede usar el entorno de simulación Eve-ng o PNetlab, ya que permite el uso de imágenes de equipos de los diferentes *vendors* que existen como HUAWEI, CISCO, JUNIPER, FORTINET, etc.

REFERENCIAS BIBLIOGRÁFICAS

[1] G. J. Cuba and J. M. Becerra, “Diseño e implementación de un controlador SDN/openflow para una red de campus académica,” PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ, 2015.

[2] Y. Ramdoss, “Cisco Catalyst 6500 Series Switches Troubleshooting TechNotes,” 2020. https://www.cisco.com/c/en/us/products/switches/catalyst-6500-series-switches/index.html (accessed May 29, 2020).

[3] “Datos administrativos - PUCP | Pontificia Universidad Católica del Perú,” 2020. https://www.pucp.edu.pe/la-universidad/nuestra-universidad/pucp-cifras/datos-administrativos/?seccion=7estructura-informatica&indicador=red-local-cableada-en-el-campus-de-pando (accessed May 14, 2020).

[4] “Understanding Catalyst 2900 and Catalyst 4000 Naming Conventions - Cisco,” 2009. https://www.cisco.com/c/en/us/support/docs/switches/catalyst-4000-series-switches/10605-97.html (accessed May 29, 2020).

[5] “Preguntas frecuentes sobre el controlador de LAN inalámbrica (WLC) - Cisco,” 2009. https://www.cisco.com/c/en/us/support/docs/wireless/4400-series-wireless-lan-controllers/69561-wlc-faq.html (accessed May 14, 2020).

[6] CISCO, “Cómo funcionan las redes privadas virtuales,” 2008. https://www.cisco.com/c/es_mx/support/docs/security-vpn/ipsec-negotiation-ike-protocols/14106-how-vpn-works.html (accessed Jun. 02, 2020).

[7] “Datos administrativos - PUCP | Pontificia Universidad Católica del Perú.” https://www.pucp.edu.pe/la-universidad/nuestra-universidad/pucp-cifras/datos-administrativos/?seccion=7estructura-informatica&indicador=red-metropolitana-y-acceso-a-internet (accessed May 15, 2020).

[8] I. Bernal and D. Mejía, “Las Redes Definidas por Software y los Desarrollos Sobre Esta Temática en la Escuela Politécnica Nacional,” *Rev. Politécnica*, vol. 37, 2016, doi: 10.33333/RP.V37I1.610.

[9] J. M. Malgosa, “Programación de redes SDN mediante el controlador POX,” UNIVERSIDAD POLITÉCNICA DE CARTAGENA.

[10] “(PDF) Lineamientos para el Despliegue de Redes SDN/OpenFlow,” *Rev. Venez. Comput.*, vol. 4, pp. 21–33, 2017, Accessed: May 28, 2020. [Online]. Available: https://www.researchgate.net/publication/333902840_Lineamientos_para_el_Despliegue_de_Redes_SDNOpenFlow.

[11] H. A. Eissa, K. A. Bozed, and H. Younis, “Software Defined Networking,” 2019.

[12] M. S. Ruiz and C. De Montegancedo, “Principios y Aplicaciones de las Redes Activas,” 1999, Accessed: May 28, 2020. [Online]. Available: https://e-archivo.uc3m.es/handle/10016/2343.

[13] C. Luis, “DISEÑO Y SIMULACIÓN DE UN PROTOTIPO DE RED DEFINIDA POR SOFTWARE (SDN) USANDO EL PROTOCOLO OPENFLOW,” Universidad de Guayaquil, 2017.

[14] A. Serrano, “Redes Definidas por Software (SDN): OpenFlow,” UNIVERSITAT POLITECNICA DE VALENCIA, 2015.

[15] C. Carballo González, “‘IX Simposio Internacional de Telecomunicaciones’ SD-WAN, UNA OPORTUNIDAD PARA LA TRANSFORMACIÓN DIGITAL SD-WAN, AN OPPORTUNITY FOR DIGITAL TRANSFORMATION.”

[16] “RFC 7426 - Software-Defined Networking (SDN): Layers and Architecture Terminology.” https://tools.ietf.org/html/rfc7426 (accessed Jun. 21, 2020).

[17] “Overview of RFC7426: SDN Layers and Architecture Terminology - IEEE Software Defined Networks.” https://sdn.ieee.org/newsletter/september-2017/overview-of-rfc7426-sdn-layers-and-architecture-terminology (accessed Jun. 28, 2020).

[18] Red Hat, “¿Qué es una API?,” *Red Hat*, 2014. https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces (accessed May 17, 2020).

[19] P. Vijay Tijare and D. Vasudevan, “IJESRT INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY THE NORTHBOUND APIs OF SOFTWARE DEFINED NETWORKS,” © *Int. J. Eng. Sci. Res. Technol.*, vol. 501, doi: 10.5281/zenodo.160891.

[20] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015, doi: 10.1109/JPROC.2014.2371999.

[21] L. Ochoa-Aday, Cervelló-Pastor-Cristina, and A. Fernandez, “ Current Trends of Topology Discovery in OpenFlow-based Software Defined Networks,” 2015, Accessed: Jun. 22, 2020. [Online]. Available: https://www.researchgate.net/publication/311577105_Current_Trends_of_Topology_Discovery_in_OpenFlow-based_Software_Defined_Networks.

[22] V. Moreno, “Evolution of Network Overlays in Data Center Clouds,” in *Cisco Live*, 2014, Accessed: Jun. 23, 2020. [Online]. Available: https://www.ciscolive.com/c/dam/r/ciscolive/emea/docs/2015/pdf/BRKDCT-2328.pdf.

[23] “¿Cuál es la definición de redes de superposición (SDN Overlay)?” https://www.sdxcentral.com/networking/sdn/definitions/what-is-overlay-networking/ (accessed Jun. 23, 2020).

[24] A. S. Nowik, “MIGRACIÓN DE REDES DE DATOS TRADICIONALES HACIA REDES DEFINIDAS POR SOFTWARE,” 2016.

[25] ONF, “OpenFlow Switch Specification Version 1.5.1 (Protocol version 0x06) for information on specification licensing through membership agreements,” vol. 1, p. 283, 2015, Accessed: Jun. 25, 2020. [Online]. Available: http://www.opennetworking.org.

[26] “Overview of OpenFlow v1.3.0.” http://docs.ruckuswireless.com/fastiron/08.0.61/fastiron-08061-sdnguide/GUID-031030CA-62EC-4009-A516-5510238EF8F4.html (accessed Jun. 26, 2020).

[27] Z. K. Khattak, M. Awais, and A. Iqbal, “Performance evaluation of OpenDaylight SDN controller,” in *Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS*, 2014, vol. 2015-April, pp. 671–676, doi: 10.1109/PADSW.2014.7097868.

[28] D. Ricardo, M. Palacios, A. Marcelo, and Z. Zambrano, “ESCUELA POLITÉCNICA NACIONAL,” ESCUELA POLITÉCNICA NACIONAL, 2018.

[29] “¿Qué es un controlador OpenDaylight? - SDxCentral.com.” https://www.sdxcentral.com/networking/sdn/definitions/opendaylight-controller/ (accessed Jun. 26, 2020).

[30] N. España, “UNIVERSIDAD CENTRAL DEL ECUADOR FACULTAD DE INGENIERÍA CIENCIAS FÍSICAS Y MATEMÁTICA CARRERA DE INGENIERÍA INFORMÁTICA DISEÑO Y SIMULACIÓN DE UNA RED DEFINIDA POR SOFTWARE (SDN) TRABAJO DE GRADUACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERA INFORMÁTI,” Universidad Central del Ecuador, Quito, 2016.

[31] P. Goransson, C. Black, and T. Culver, *Software Defined Networks: A Comprehensive Approach - Paul Goransson, Chuck Black, Timothy Culver - Google Libros*, 2nd ed. Cambridge: Todd Green, 2014.

[32] “Software Defined Network: el futuro de las arquitecturas de red.”

[33] Evangelos Haleplidis, “Overview of RFC7426: SDN Layers and Architecture Terminology - IEEE Software Defined Networks,” 2017. https://sdn.ieee.org/newsletter/september-2017/overview-of-rfc7426-sdn-layers-and-architecture-terminology (accessed Jun. 23, 2020).

[34] “¿Qué es un controlador de Floodlight? - Definido - SDxCentral.” https://www.sdxcentral.com/networking/sdn/definitions/what-is-floodlight-controller/ (accessed Jul. 26, 2020).

[35] I. C. H. PICA8, “PICA8 P -3297,” *2014*. Accessed: Jun. 18, 2020. [Online]. Available: https://www.pica8.com/wp-content/uploads/pica8-datasheet-48x1gbe-p3297.pdf.

[36] “Summit X440-Data Sheet Summit X440 Series HIGHLIGHTS.” Accessed: Jul. 12, 2020. [Online]. Available: http://safenet-co.net/uploads/Extreme/Catalog/Summit-X440x-DS.pdf.

[37] “Intel ® Ethernet Switch FM6000 Series 10/40 GbE Low Latency Switching Silicon,” 2013. Accessed: Jul. 12, 2020. [Online]. Available: www.intel.com/go/ethernet.

[38] “SD-WAN – RedSolutions.” https://redsolutions.cl/sd-wan/ (accessed Jun. 23, 2020).

[39] “Mininet: a versatile tool for emulation and prototyping of Software Defined Networking.” http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S1909-83672015000100009 (accessed Jul. 26, 2020).

[40] Miriam Barrios del Sol, ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA Y SISTEMAS DE TELECOMUNICACIONES, "Diseño y despliegue de escenarios de red sobre un entorno de pruebas virtualizado SD-WAN basado en tecnología Viptela", Universidad Politecnica de Madrid, España, 2020

[41] CISCO SD-WAN, https://www.cisco.com/c/en/us/solutions/enterprise-networks/sd-wan/index.html (Acedido mayo 26,2022)

[42] ¿Qué es la SD-WAN? Definición y soluciones, https://www.fortinet.com/lat/products/sd-wan (Acedido mayo 26,2022)