PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

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

TRABAJO DE INVESTIGACIÓN PARA LA OBTENCIÓN DEL GRADO DE BACHILLER EN CIENCIAS CON MENCIÓN EN INGENIERIA ELECTRONICA

AUTOR:

Alexis Pierre Estela Lozano

ASESOR:

M.Sc.Ing. Gumercindo Bartra Gardini

Lima, noviembre, 2020

RESUMEN

El presente trabajo académico está dividido en 2 capítulos:

El primer capítulo se presenta cómo se encuentra desplegada actualmente la red de la Pontificia Universidad Católica del Perú como es el caso de la ***Local Area Network*** (LAN) y la ***Wide Area Network*** (WAN). Centrándose en la segunda para la explicación de los tipos de conexiones con los edificios externos. Se toma como base literaturas sobre arquitecturas actuales, realización de mediciones de los diferentes servicios que esta institución ofrece a los usuarios y la calidad con la que se les brinda, con el objetivo de presentar las problemáticas de las redes actuales usando como ejemplo la red PUCP.

El segundo capítulo, se enfoca en brindar una introducción al nuevo paradigma ***Software Defined Network*** (SDN) como solución a las problemáticas encontradas en las redes actuales. Se explica cómo se encuentra distribuido una arquitectura SDN, el tipo de comunicación de los planos a través de los *Application Protocol Interface* (API), el funcionamiento de esta tecnología en diferentes escenarios como Data Centers, redes WAN y redes de campus; y finalmente se presenta una solución usando redes SDN para la red PUCP junto con sus parámetros medibles.

INDICE GENERAL

INDICE DE FIGURAS

INTRODUCCIÓN

En la actualidad las redes de gran tamaño con conexiones entre sedes externas poseen una gran cantidad de dispositivos conectados a estos equipos de red y este número sigue en crecimiento. Además, en general, este tipo de redes posee todo tipo de tráfico, como es el caso de red PUCP que brinda una gran cantidad de servicios para los usuarios como servidores de correo electrónico, servidores de video, etc. Esta gran cantidad de tráfico que se genera puede ocasionar congestión de los enlaces, retardos elevados o en muchos de los casos caída de los enlaces y en consecuencia perdida de conectividad con el servicio. Estos eventos generan que las redes grandes no sean escalables y dificultan el soporte y mantenimiento de la red al equipo de TI de la universidad.

Las redes SDN crean un gran cambio en las redes actuales, ya que permite una eficiente gestión de la red, redundancia de enlaces y finalmente un control centralizado de dicha red, esto se logra, ya que todo el control de tráfico (plano de control) que en las redes tradicionales se gestionaba en el dispositivo de red como es el caso de los *routers,* en una red SDN , el control lo realiza un nuevo elemento llamado controlador, cuya principal característica es que tendrá visibilidad completa de la red , además de tener el control de designar la funcionalidad a cada dispositivo de red dependiendo de la necesidad que posea la institución en cada momento. Esto genera una disminución de los costos de los dispositivos a usar, disminución de la dificultad en el mantenimiento y *troubleshooting* de la red para el equipo de TI de la universidad.

El objetivo de este trabajo académico es presentar un diseño de una red WAN para la red PUCP usando la tecnología SDN, a este tipo de red se le conoce como SD-WAN. Esta red será capaz de permitir un aumento de dispositivos masivo por ende podrá gestionar una gran cantidad de tráfico y brindar un menor tiempo de latencia entre la conexión de las sedes externas hacia los servicios que proporciona la sede central mediante enlaces de internet.

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.

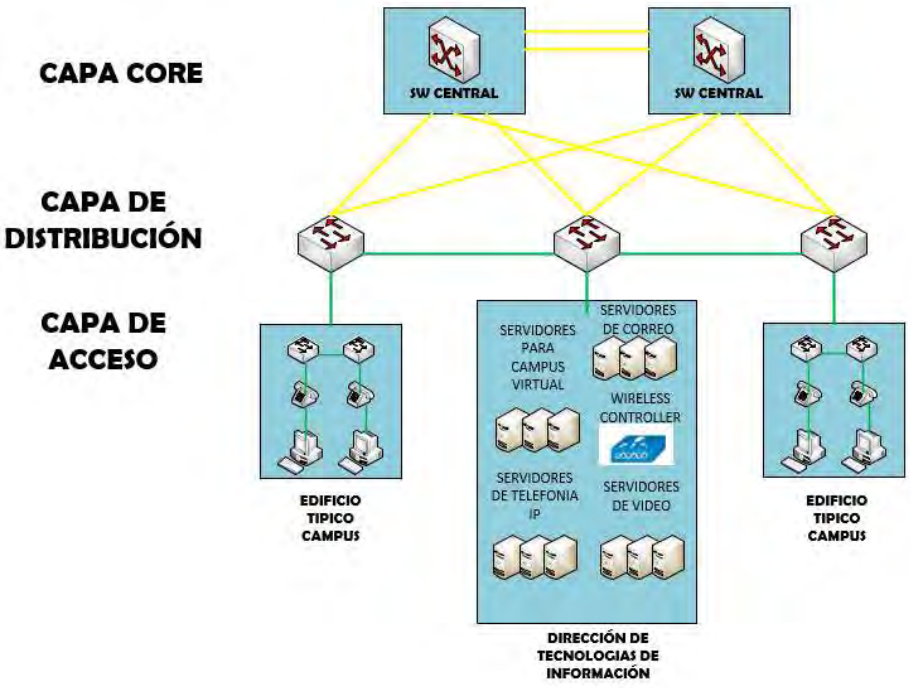

Figura 1: Estructura de la red PUCP
Fuente: Elaboración propia (2020)

1.1.1.1 Capa Core

La red está conformada por dos *Switchs* 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 *switchs* que se interconectan a los *switchs* de la capa de acceso, que se encuentran en todos los pisos de los edificios, y los *switchs* de la capa core. Los *switchs* 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 *switchs* 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.

1.1.2.1. Conexiones al Core

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.

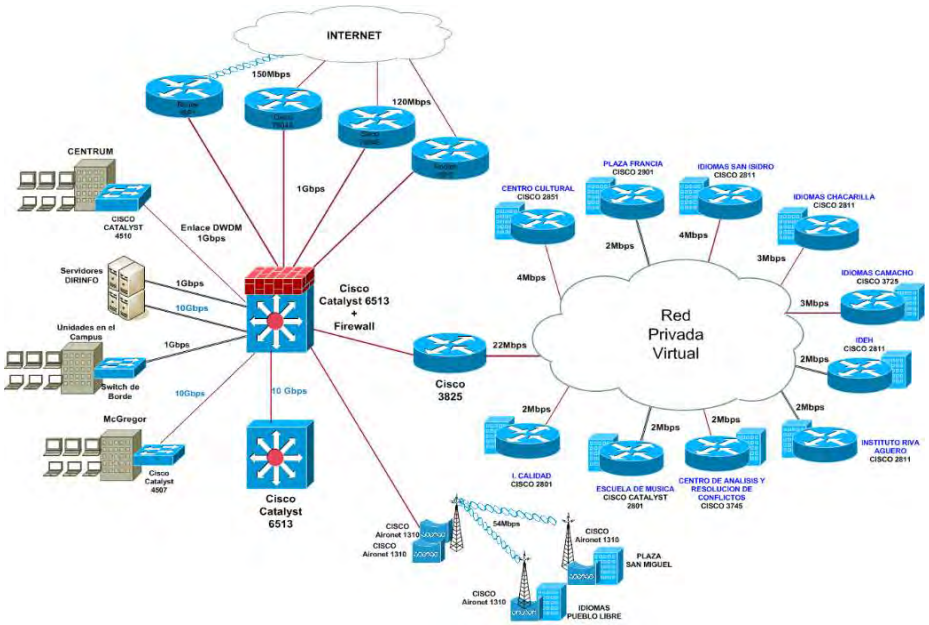

Figura 3: Velocidades de las conexiones WAN de la red PUCP [7]

1.2. Problemática de la red PUCP

La red actual de la PUCP básicamente se conforma por equipos de conmutación como *routers* y *switchs* 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 (*switchs, 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 *switchs*. 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 5, 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.

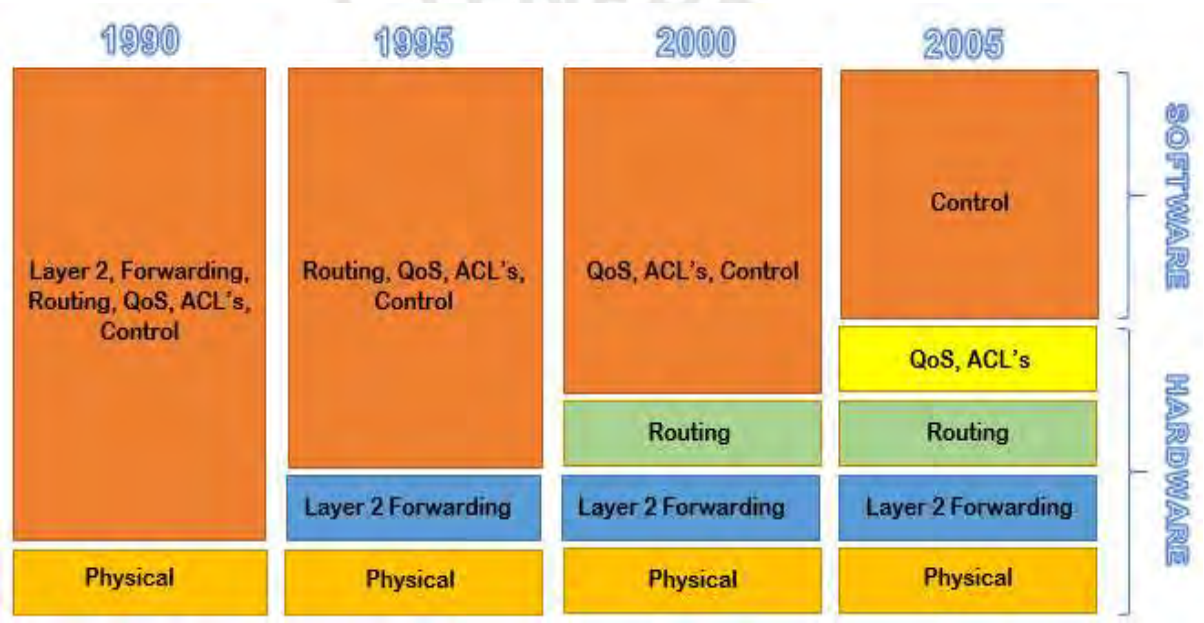

Figura 4: Evolución de las funcionalidades de los equipos de red [9]

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.

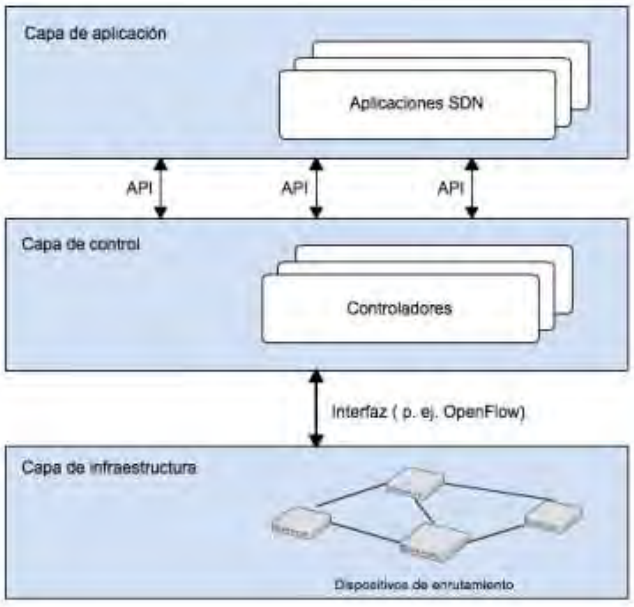

Figura 5: Separación de planos en SDN [8]

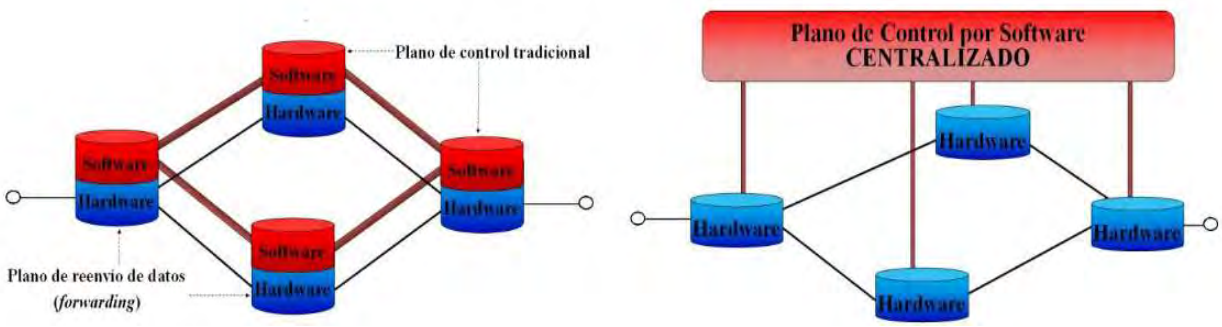

Figura 6: Arquitectura SDN [14]

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 Interface*) 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.

1.4.2 Objetivos específicos

1. Analizar los servicios actuales que brinda la red PUCP
2. Realizar una estrategia de provisionamiento de nuevos servicios.
3. Establecer redundancias entre los enlaces críticos para asegurar la supervivencia de la red en caso de fallos y permitir rutas alternativas
4. Brindar un control masivo de todos los dispositivos conectados a la red
5. Realizar la simulación de la red en una plataforma virtualizada.
6. 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.

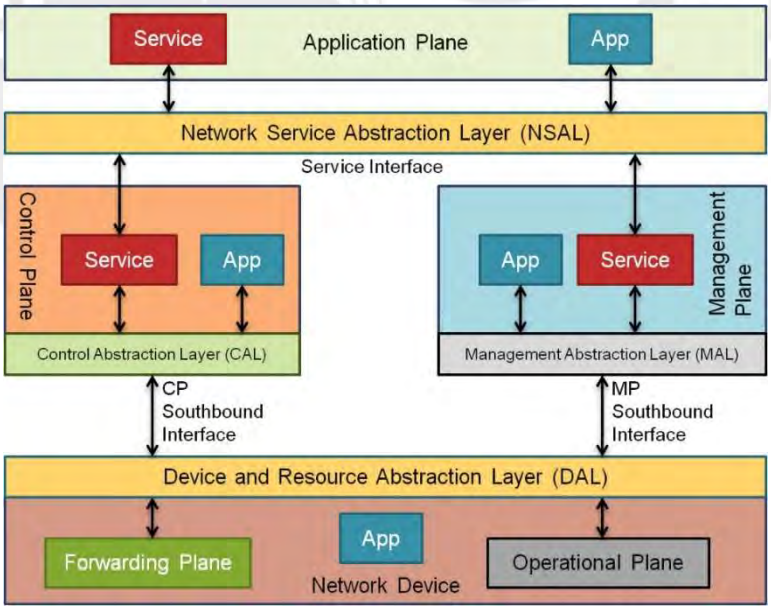

Figura 7: Arquitectura SDN [17]

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].

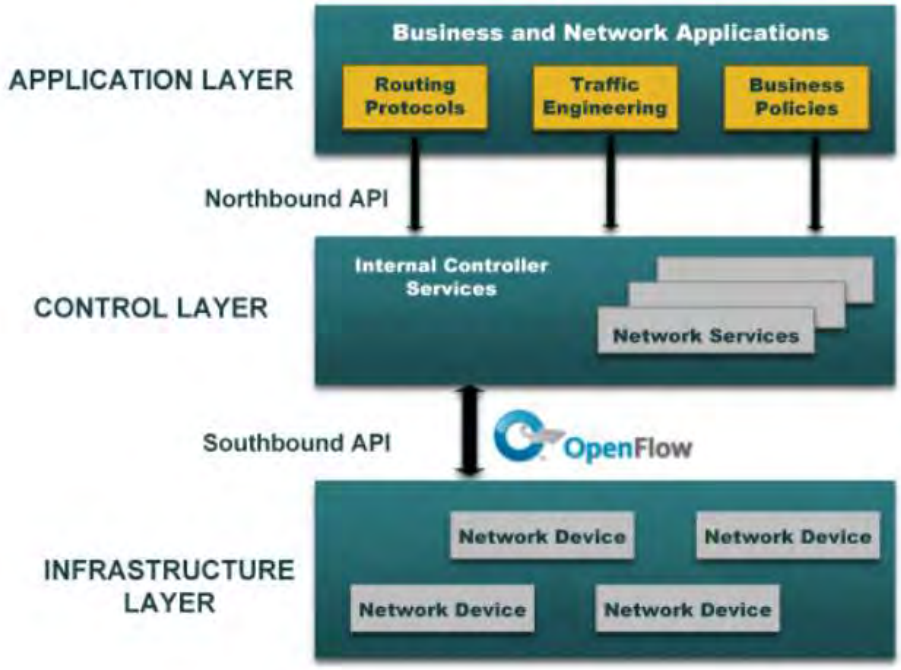

Figura 8: Arquitectura SDN con las interfaces de comunicación [21]

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.

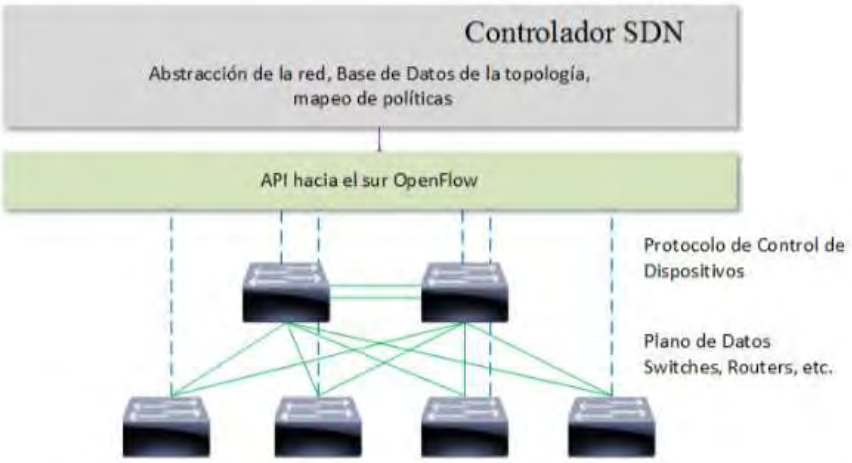

Figura 9: Arquitectura SDN con las interfaces de comunicación [10]

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.

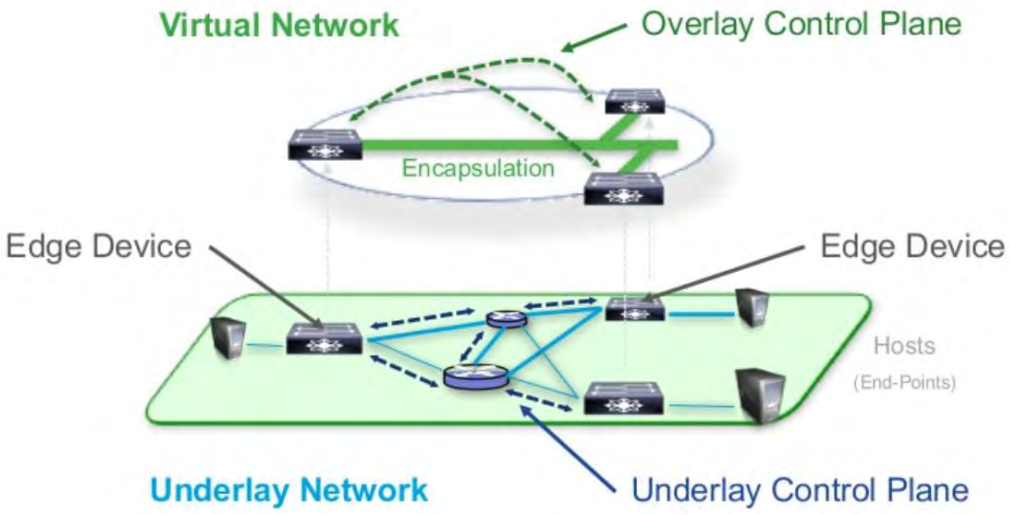

Figura 10: Modelo SDN Overlay [22]

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.

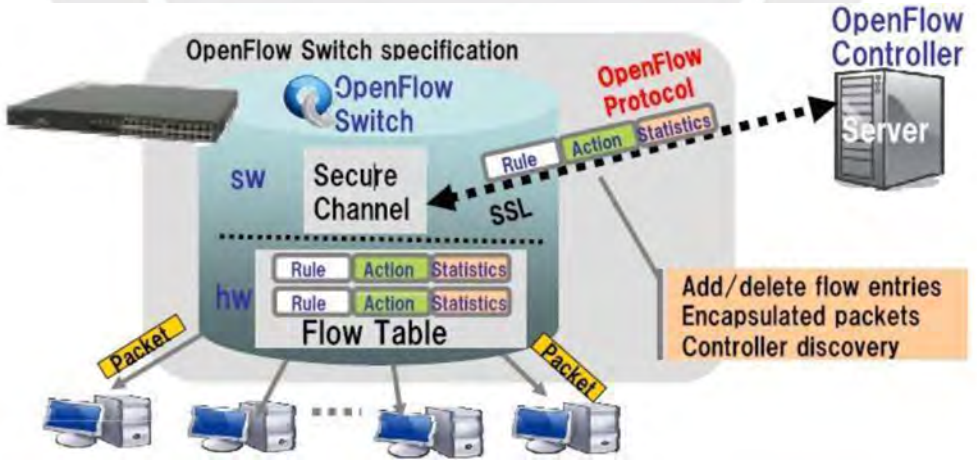

Figura 11: Arquitectura SDN/OpenFlow [1]

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 del *switch*.

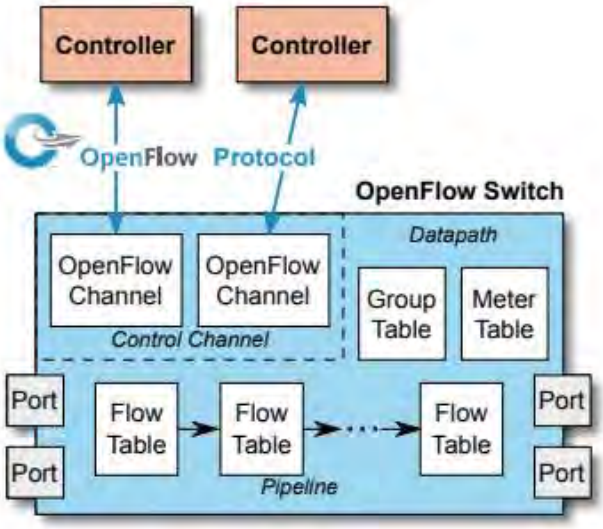

Figura 12: Componentes del Switch OpenFlow [25]

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.

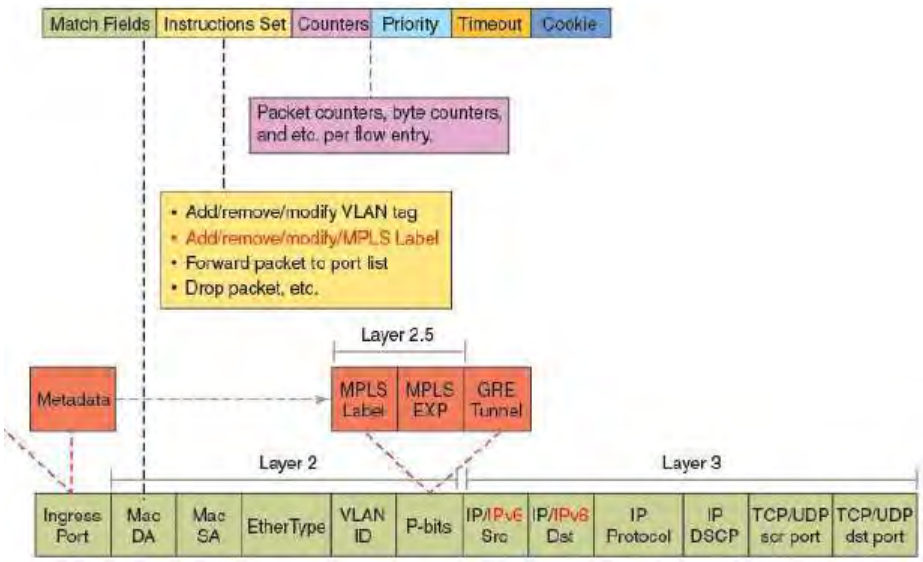

Figura 13: Campos de una Flow entry [26]

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 relacionados 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.

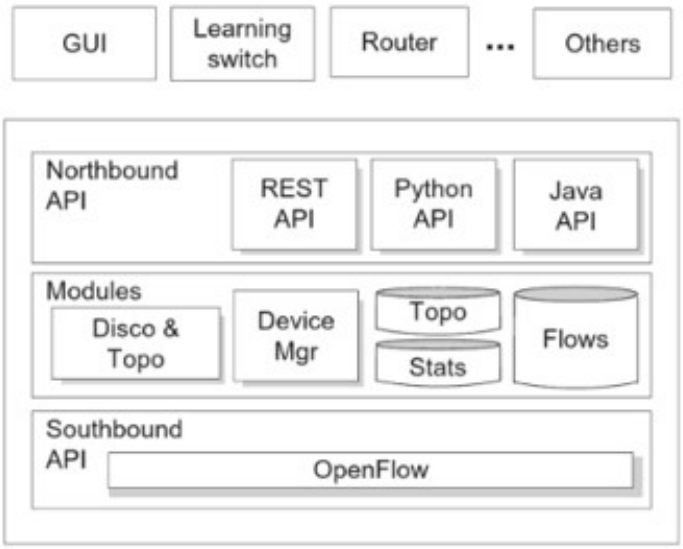

Figura 14: Diagrama de bloques controlador [31]

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.

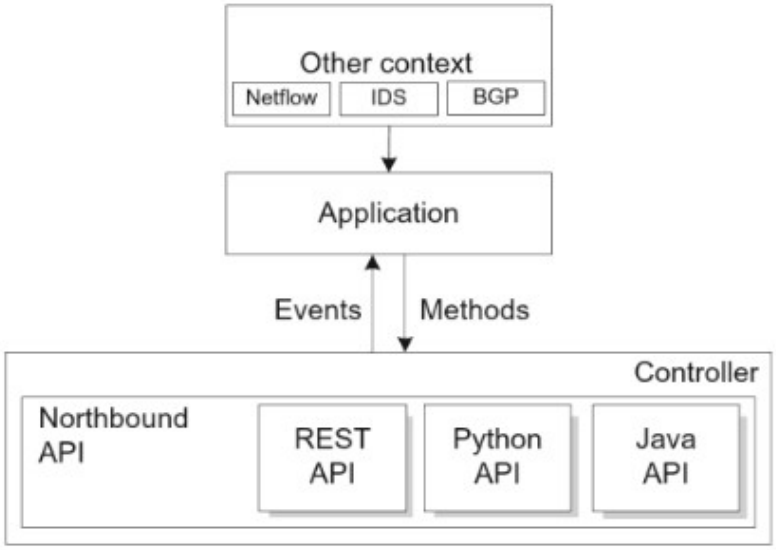

Figura 15: Mensajes entre control plane y application plane [31]

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.

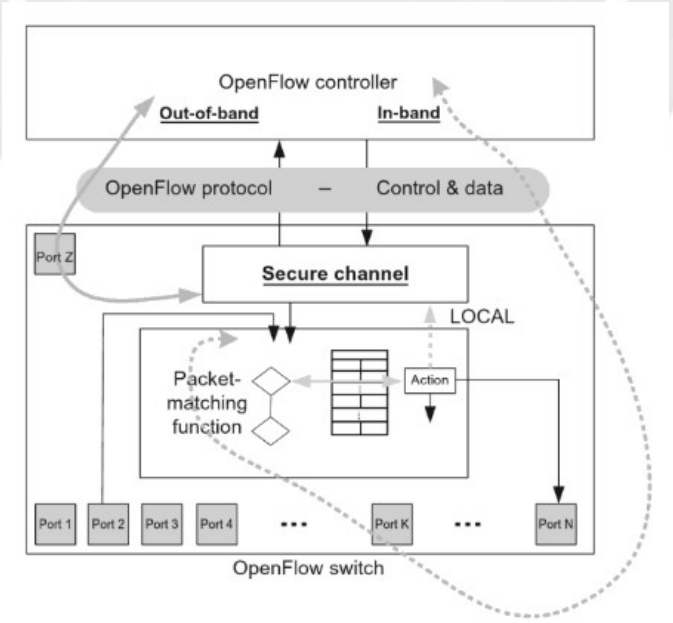

Figura 16: Canal seguro controlador- switch [31]

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].

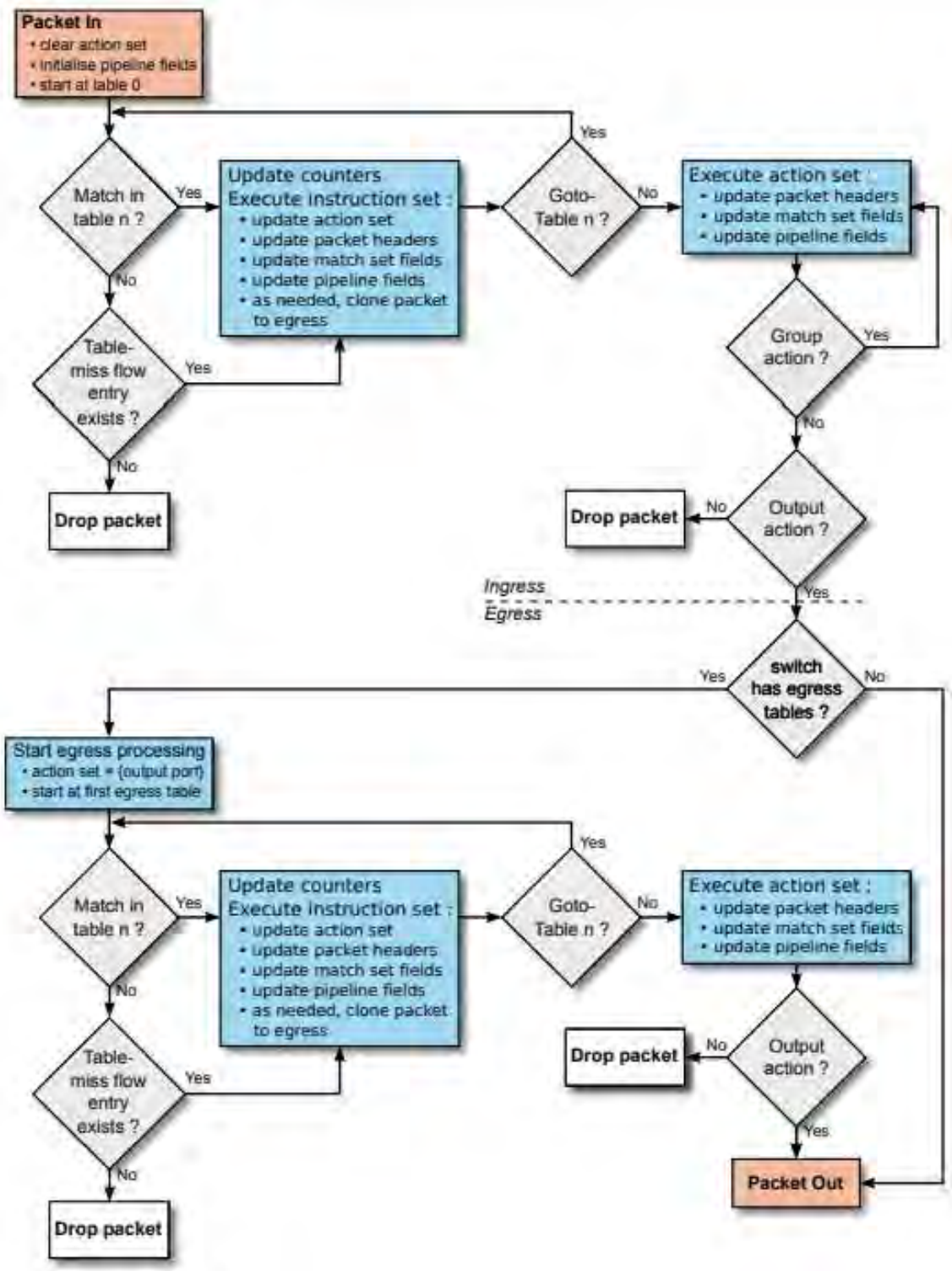

Figura 17: Diagrama de bloques del funcionamiento de la arquitectura SDN [25]

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).

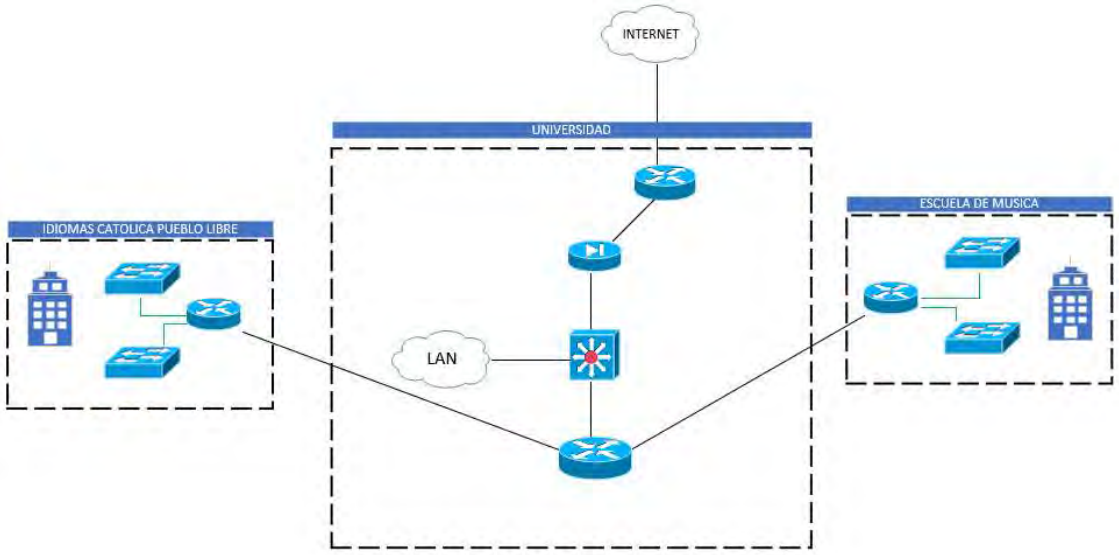

Figura 18: Conexiones físicas red WAN PUCP
Fuente: Elaboración propia (2020)

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.

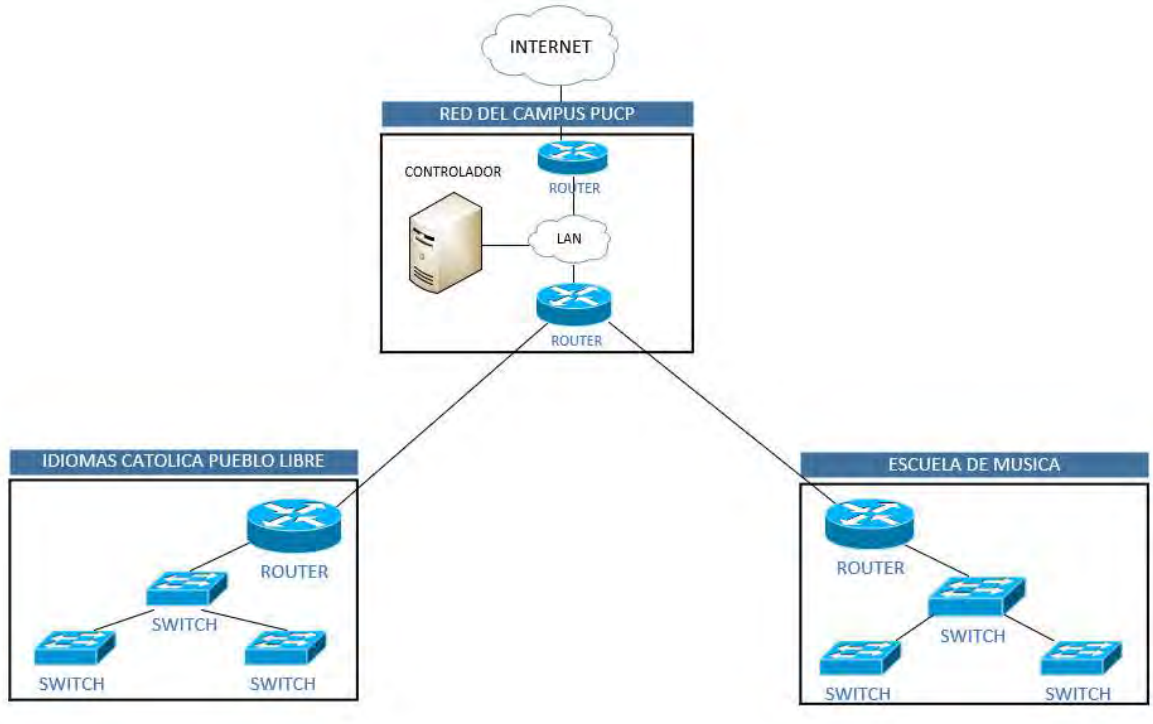

Figura 19: Topología física SD-WAN PUCP
Fuente: Elaboración propia (2020)

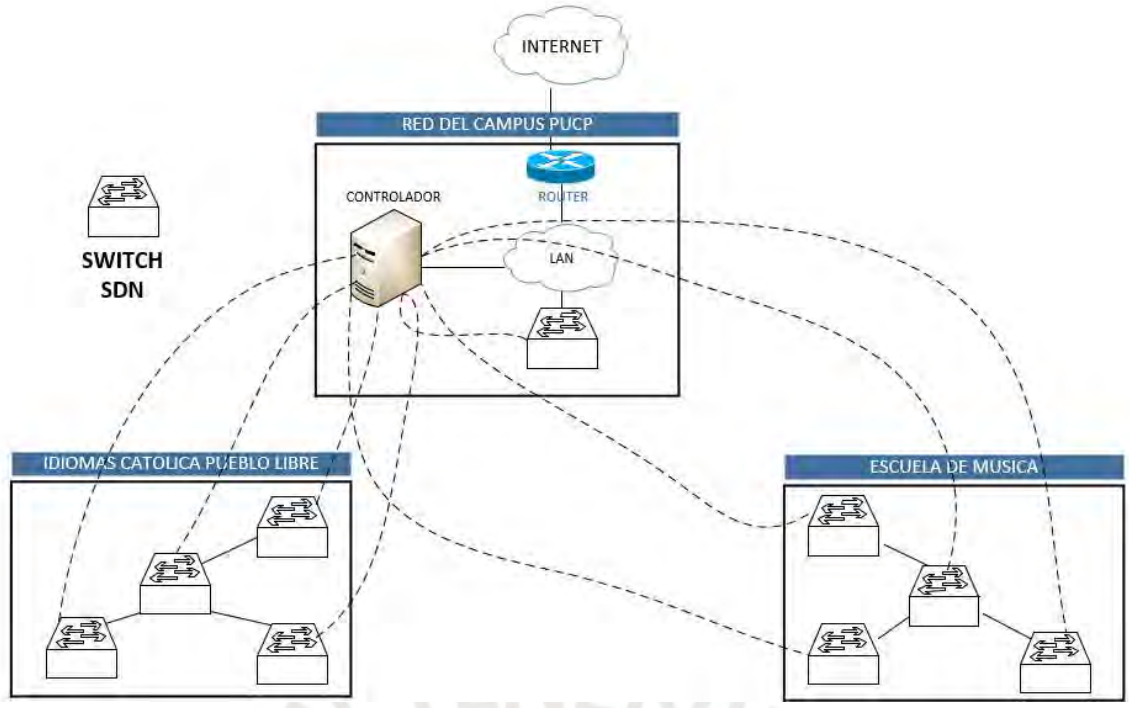

Figura 20: Topología lógica SD-WAN PUCP
Fuente: Elaboración propia (2020)

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.

3. CONCLUSIONES

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.

3.4 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.

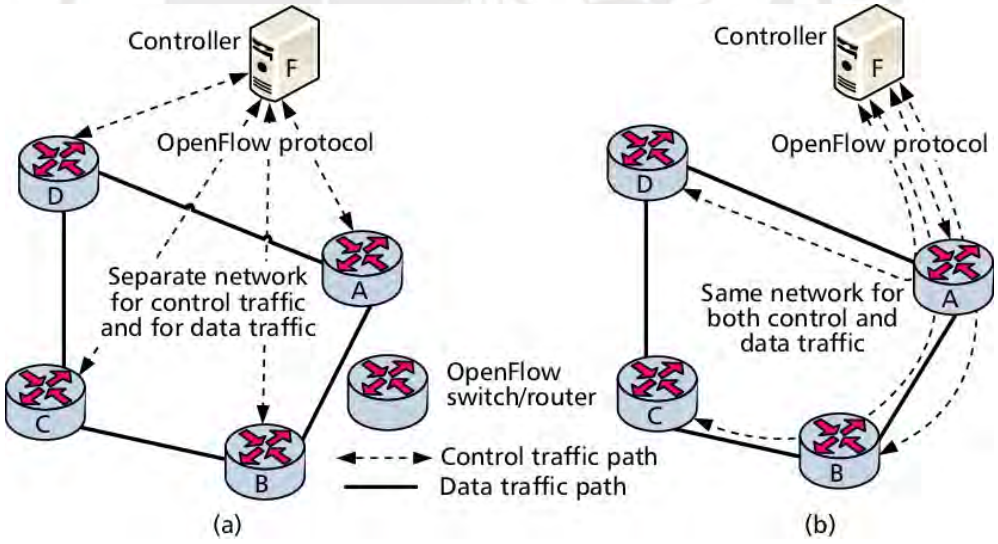

Figura 21: Conexiones del canal seguro *Switch*- controlador a) *Out of band* y b) *In band* [32]

4. RECOMENDACIONES Y TRABAJOS FUTUROS

- Realizar pruebas en un entorno de simulación de redes SDN como MININET para comprobar tiempos de latencia y ancho de banda entre usuarios en redes distintas con el fin de tener parámetros medibles en relación con una red tradicional.
- Realizar pruebas en simulador de redes *Legacy* como GNS3, Eve-ng o Packet Tracert sin necesidad de diferenciación de vlan's solo para conocer parámetros de una red tradicional.
- Realizar pruebas de estrés en ambos tipos de redes para obtener una comparación de cuál es el factor que los diferencian en escalabilidad de dispositivos.
- En el escenario SDN, implementar un segundo controlador como *backup* en caso de caída o perdida de conexión con el controlador principal.
- Implementar el diseño de la red SD-WAN PUCP, pero orientado a dispositivos inalámbricos como *Access Point*, ya que es a estos equipos los que se conectan los usuarios en las distintas sedes.
- Realizar la implementación de una la red SD-WAN PUCP de una forma más real aumentando los diferentes protocolos que se usan para tener conectividad con el *Service Provider* como BGP, MPLS-VPN, etc.

5. BIBLIOGRAFÍA

[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).