UNIVERSIDAD RICARDO PALMA

FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA ELECTRÓNICA

Diseño de una Red Definida por Software (SDN) mediante el protocolo OpenFlow para la Universidad Ricardo Palma, Lima, 2023.

TESIS

Para optar el título profesional de Ingeniero Electrónico

AUTORES

Rojas Vargas, Giovanny Joel

ORCID: 0009-0009-6290-6565

Cruz Guerrero, Freddy Alonso

ORCID: 0009-0008-2849-7430

ASESOR

Cuadrado Lerma, Luis Alberto

ORCID: 0000-0001-9689-3461

Lima, Perú

2023

METADATOS COMPLEMENTARIOS

Datos del autor(es)

Rojas Vargas, Giovanny Joel
DNI: 70350866
Cruz Guerrero, Freddy Alonso
DNI: 76256722

Datos del asesor

Cuadrado Lerma, Luis Alberto
DNI: 10448199

Datos del jurado

JURADO 1
Burneo Gonzales, Katia Janet
DNI: 09391942
ORCID: 0000-0002-7046-8106

JURADO 2
Rodriguez Alcazar, Jose Luis Antonio
DNI: 08242196
ORCID: 0000-0003-2238-3017

JURADO 3
Palomino Navarro, Isaac Manuel
DNI: 06260402
ORCID: 0009-0008-9854-9259

Datos de la investigación

Campo del conocimiento OCDE: 2.02.01
Código del Programa 712026

DECLARACIÓN JURADA DE ORIGINALIDAD

Nosotros, Cruz Guerrero Freddy Alonso, con código de estudiante N°201611876, con DNI N°76256722, con domicilio en Calle las palmeras 247 Buenos Aires San Gabriel Mz. L Lt. 1, distrito de Villa María del Triunfo, provincia y departamento de Lima, y Rojas Vargas Giovanny Joel, con código de estudiante N°201611877, con DNI N°70350866, con domicilio en Av. Zorritos 1399 block 21 dpto 103, distrito de Cercado de Lima, provincia y departamento de Lima, en nuestra condición de bachilleres en Ingeniería Electrónica de la Facultad de Ingeniería, declaramos bajo juramento que:

La presente tesis titulada: "Diseño de una Red Definida por Software (SDN) mediante el protocolo OpenFlow para la Universidad Ricardo Palma, Lima, 2023" es de nuestra única autoría, bajo el asesoramiento del docente Cuadrado Lerma Luis Alberto, y no existe plagio y/o copia de ninguna naturaleza, en especial de otro documento de investigación presentado por cualquier persona natural o jurídica ante cualquier institución académica o de investigación, universidad, etc.; la cual ha sido sometida al antiplagio Turnitin y tiene el 13 % de similitud final.

Dejamos constancia que las citas de otros autores han sido debidamente identificadas en la tesis, el contenido de estas corresponde a las opiniones de ellos, y por las cuales no asumimos responsabilidad, ya sean de fuentes encontradas en medios escritos, digitales o de internet.

Asimismo, ratificamos plenamente que el contenido íntegro de la tesis es de nuestro conocimiento y autoría. Por tal motivo, asumimos toda la responsabilidad de cualquier error u omisión en la tesis y somos conscientes de las connotaciones éticas y legales involucradas.

En caso de falsa declaración, nos sometemos a lo dispuesto en las normas de la Universidad Ricardo Palma y a los dispositivos legales nacionales vigentes.

Surco, 5 de Noviembre de 2023

(Nombre completo)	(Nombre completo)
DNI N°76256722	DNI N°70350866

INFORME DE ORIGINALIDAD-TURNITIN

DISEÑO DE UNA RED DEFINIDA POR SOFTWARE (SDN) MEDIANTE EL PROTOCOLO OPENFLOW PARA LA UNIVERSIDAD RICARDO PALMA, LIMA 2023.

INFORME DE ORIGINALIDAD

13%	%	%	13%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	Submitted to Universitat Politècnica de València Trabajo del estudiante	1%
2	Submitted to UNAPEC Trabajo del estudiante	1%
3	Submitted to UNIV DE LAS AMERICAS Trabajo del estudiante	1%
4	Submitted to Submitted on 1690341597174 Trabajo del estudiante	1%
5	Submitted to Pontificia Universidad Catolica del Peru Trabajo del estudiante	1%
6	Submitted to Universidad Tecnica De Ambato-Direccion de Investigacion y Desarrollo , DIDE Trabajo del estudiante	1%
7	Submitted to Universidad Tecnologica del Peru Trabajo del estudiante	1%

Mg Ing Eduardo Ale Estrada

DEDICATORIA

Dedico esta tesis a mi querida mamá Wilma, a mi querido papá Juan, mis hermanos Giancarlo, Jhon y Gina, amigos y colegas quienes me brindan el apoyo día a día para para salir adelante, muy agradecido con todos.

Giovanny Joel Rojas Vargas

Esta tesis va dedicada a mis padres, mi hermana y mi tía, quienes siempre me han apoyado e impulsado a ser mejor persona y lograr mis objetivos día a día, enseñándome que con dedicación y ganas todo es posible.

Freddy Alonso Cruz Guerrero

AGRADECIMIENTO

Muy agradecidos con la universidad, nuestra alma máter, y con los asesores de tesis por habernos aportado la guía y los conocimientos necesarios para poder lograr la realización de esta tesis, dándonos así un empujón más en nuestra formación profesional para poder seguir realizando y alcanzando nuestras metas.

Giovanny Rojas y Freddy Cruz.

ÍNDICE GENERAL

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

En este proyecto de tesis, se llevó a cabo la creación y simulación de una red definida por software utilizando el protocolo OpenFlow en la Universidad Ricardo Palma en el año 2023. Para lograrlo, se configuró una máquina virtual en la que se instaló el software de simulación MININET. Esta herramienta permitió la simulación de la infraestructura de red principal de la Universidad Ricardo Palma.

La máquina virtual se creó utilizando el software de virtualización VirtualBox, con una configuración que incluía una memoria base de 6000MB, un sistema operativo Ubuntu de 64 bits y un tamaño de disco de 50GB. Luego, se procedió a instalar el software MININET mediante comandos de Linux desde la consola de la máquina virtual. Además, se instaló el controlador OpenDaylight utilizando código Java.

Para el diseño de la arquitectura de la Universidad Ricardo Palma, se empleó Mininedit, un complemento de MININET que facilitó la tarea de diseño. Una vez completada la configuración de la arquitectura en el simulador, se ejecutó la simulación y se pudieron visualizar los resultados a través de la interfaz de la dirección IP de la máquina virtual que albergaba el controlador OpenDaylight. Esta interfaz proporcionó opciones de visualización y automatización para la red simulada.

Palabras claves: Red definida por software (SDN), Protocolo Openflow, Mininet, virtualización, Controlador OpenDaylight, VirtualBox, Java.

ABSTRACT

In this thesis project, we carried out the creation and simulation of a software-defined network using the OpenFlow protocol at the University Ricardo Palma in 2023. To achieve this, we configured a virtual machine in which we installed the MININET simulation software. This tool enabled the simulation of the University Ricardo Palma's primary network infrastructure.

The virtual machine was created using the VirtualBox virtualization software, with a configuration that included a base memory of 6000MB, a 64-bit Ubuntu operating system, and a disk size of 50GB. Subsequently, we proceeded to install the MININET software using Linux commands from the virtual machine's console. Additionally, we installed the OpenDaylight controller using Java code.

For the design of the University Ricardo Palma's architecture, we utilized Miniedit, an add-on for MININET that facilitated the design task. Once the architecture configuration was completed in the simulator, we executed the simulation, and the results could be visualized through the interface of the virtual machine's IP address that hosted the OpenDaylight controller. This interface provided visualization and automation options for the simulated network.

Keywords: Software-Defined Network (SDN), OpenFlow Protocol, Mininet, Virtualization, OpenDaylight Controller, VirtualBox, Java.

INTRODUCCIÓN

En la Universidad Ricardo Palma, en la actualidad, hay alrededor de 12.000 estudiantes, aproximadamente 1.100 profesores y 600 empleados, además de una amplia comunidad de egresados que desempeñan diversas funciones en el sector público y privado en el Perú y en el extranjero.

Es esta tesis empezamos definiendo el planteamiento del problema junto con los objetivos generales y específicos. En este caso, se trataba de diseñar una red definida por software (SDN) mediante el protocolo OpenFlow para la Universidad Ricardo Palma en Lima, en el año 2023. A continuación, delimitamos y establecimos enfoques generales para el desarrollo de la tesis.

En el segundo capítulo, se abordó la parte teórica que se emplearía en nuestro diseño. Se revisaron los antecedentes de la investigación, se definió qué es una red definida por software, el protocolo OpenFlow, el controlador OpenFlow, Mininet y Miniedit, además de la virtualización. También se detalló el tipo y nivel de investigación, así como el diseño de investigación, que incluyó una descripción de la estructura de la Universidad Ricardo Palma.

El tercer capítulo se centró en la ingeniería, comenzando con la selección del controlador OpenFlow. Se creó un entorno virtual en VirtualBox, donde se configuró una máquina virtual con el sistema operativo Linux, aprovechando su carácter de código abierto para facilitar la instalación del controlador. También se implementó Mininet, un emulador que permitió simular la red de la Universidad Ricardo Palma.

En el cuarto capítulo, se llevó a cabo la simulación de la red definida por software (SDN) en la Universidad Ricardo Palma utilizando el controlador OpenDaylight. Esto implicó la ejecución del comando sudo ./karaf, seguido por la ejecución del emulador Miniedit y la creación de una topología que replicara la estructura de red física de la Universidad Ricardo Palma. Se verificó la conectividad de red entre los dispositivos finales. Además, se accedió a la red de forma remota a través de la consola web del controlador OpenDaylight, lo que permitió la automatización de la red desde la topología mediante API. Esto habilitó la automatización, supervisión y visualización de las conexiones físicas en la red.

CAPÍTULO I: PLANTEAMIENTO DEL PROBLEMA

1.1. Descripción y formulación del problema general y específicos

Actualmente la red de la Universidad Ricardo Palma tiene en su esquema conexiones que operan mediante equipos físicos como lo son routers, firewalls, AP, switches, computadoras, laptops, celulares, impresoras, telefonía VoIP, etc. Los cuales se encuentran estratégicamente ubicados por el área de la universidad con la finalidad de poder satisfacer las necesidades de navegación por internet, con fines educativos, de los estudiantes y docentes. Esto genera un incremento en el tráfico de datos y de contenido multimedia al aumentar la cantidad de dispositivos que se conectan a la red, haciendo que el ancho de banda de la arquitectura actual no sea suficiente. Asimismo, al querer escalar la red actual se tendría que hacer de manera física, lo cual conlleva a costos más elevados tanto en equipos físicos como en personal capacitado que se encargue de su configuración y/o instalación. Y al querer realizar cualquier cambio o adicionamiento en la red, lo que se generará será una respuesta lenta por parte de la misma, ya que al agregar nuevos equipos se necesitaría hacer nuevas configuraciones a todos los equipos de la red que se encuentren en interacción con el cambio deseado, lo que demandará tiempo y recursos.

Es por eso que una Red Definida por Software (SDN) sería un avance y una mejor solución hacia una tecnología más actual y eficiente, permitiendo gestionar y controlar todos los equipos que se encuentran en el esquema de red a través de software.

1.1.1. Problema general

¿Cómo diseñar una de red definida por software (SDN) mediante el protocolo Open Flow para la Universidad Ricardo Palma, Lima 2023?

1.1.2. Problemas Específicos:

a) ¿Cómo diseñar la arquitectura de una red definida por software (SDN) en la Universidad Ricardo Palma, Lima 2023?
b) ¿Cómo simular una red definida por software (SDN) mediante el protocolo Open Flow para la Universidad Ricardo Palma, Lima 2023?

1.2. Objetivo general y específico

1.2.1. Objetivo general

Diseñar una red definida por software (SDN) mediante el protocolo Open Flow para la Universidad Ricardo Palma, Lima 2023.

1.2.2. Objetivos específicos

a) Diseñar la arquitectura de una red definida por software (SDN) usando el protocolo Open Flow, Lima 2023.
b) Simular la arquitectura de una red definida por software (SDN) usando el protocolo Open Flow, Lima 2023.

1.3. Delimitación de la investigación: temporal espacial y temática

a) Delimitación Teórica

   La presente investigación se diseñará mediante el uso de máquinas virtuales y libros de redes definidas por software.

b) Delimitación Tecnológica

   La presente investigación aplicará tecnología SDN, emuladores, programación en LINUX y entornos virtuales.

c) Delimitación Espacial

   La presente investigación se desarrollará en la sede de la Universidad Ricardo Palma, ubicada en el distrito de Santiago de Surco.

d) Delimitación Temporal

   La presente investigación se desarrollará desde mayo a noviembre del año 2023.

1.4. Justificación e importancia

Desde el punto de vista pedagógico: En la Universidad Ricardo Palma - Lima, se tiende semestralmente al aumento de estudiantes, por lo que se sobreentiende y se sabe que es un objetivo reducir la complejidad de gestión de las redes cuando hay incremento de conexiones que pueden saturar el tráfico de datos.

Desde el punto de vista tecnológico: El uso de una red definida por software (SDN) proporciona una nueva perspectiva en la automatización de redes, ya que se basa únicamente en software para controlar los diversos recursos de la red.

Desde el punto de vista económico: Este enfoque resulta ser una solución viable para superar los costos asociados con la implementación de equipos tradicionales, al tiempo que ofrece la flexibilidad de implementar equipos de diferentes fabricantes sin problemas de compatibilidad de estándares y protocolos, por lo que es de suma importancia realizar un diseño como el que se propone en esta tesis y poder obtener un control de red centralizado y ya no de forma distribuida, como se tendría en una red TCP/IP, y así entender esta tecnología como un paso hacia la modernización actual.

CAPÍTULO II: MARCO TEÓRICO

2.1. Antecedentes del estudio de investigación

2.1.1. Antecedentes Internacionales

Guanoluisa E.D. (2019). Diseño de la arquitectura de una red SDN mediante el protocolo Openflow con simulación en el software mininet para la infraestructura de una PYMES. Tesis. Ecuador. Universidad de las Américas. Nos manifiesta en sus conclusiones lo siguiente: Se estudió que las redes SDN flexibilizan toda administración y el despliegue de la red, permite gestionar de manera centralizada bajo un controlador todos los flujos de datos optimizando el tiempo de respuesta y reduciendo los costos de operación. Un controlador SDN puede administrar toda la red definida por software mediante diferentes funcionalidades que nos ofrece esta tecnología, como seguridades, routing, segmentación de red, este enfoque permite conservar la integridad de las API de código abierto permitida por el controlador SDN de la misma forma refuerza la orquestación del centro de datos la automatización y la gestión de toda la red en un mismo panel de control.

Zhuma, E.R. (2019). Análisis del desempeño de redes definidas por software (SDN) frente a redes con arquitectura TCP/IP. Tesis. Ecuador. Universidad Técnica Estatal de Quevedo. Nos dice en sus conclusiones que: La identificación de protocolos, controladores y emuladores ayudó a establecer con qué herramientas de software se puede contar y cuáles de estas se acoplan a los requerimientos de esta investigación. La selección del controlador, en este caso OpenDayLight y el protocolo OpenFlow es de gran importancia debido a que permite tener una correcta interacción entre los dispositivos y la administración de las redes. Así mismo esta investigación se enfatizó en la selección de dos emuladores que son Mininet y GNS3 que brindan las prestaciones correctas para permitir crear diferentes tipos de topologías y realizar evaluaciones

Rentería, B. L. y Paucar, A. A. (2022). Análisis e implementación de un prototipo de red SDN en el campus norte de la UNACH. Tesis. Ecuador. Universidad Nacional de Chimborazo. En sus conclusiones nos dice que: Se diseñó una red SDN que primero fue simulada en el emulador Mininet, para llevar a cabo las pruebas y análisis del rendimiento de la red, y posteriormente se implementó, con este análisis se concluye que se debe realizar un análisis extenso de los controladores SDN para el diseño de la red.

2.1.2. Antecedentes Nacionales

Valles W. (2022). Diseño de una red definida por software (SDN) para brindar una gestión centralizada de las configuraciones y una adecuada gestión de crecimiento de los nodos a la red de un operador de servicios. Tesis. Perú. Universidad Peruana de Ciencias Aplicadas (UPC). En sus conclusiones manifiesta que: Se pudo evidenciar que a través de la red SDN la propagación de flujos de red es inmediata, reduciendo considerablemente el tiempo de los aprovisionamientos y baja de servicios.

Rodríguez, E. (2020) Diseño y simulación de una red definida por software (SDN) para la implementación de un laboratorio avanzado de datos para la EP de Telecomunicaciones de la Facultad de Ingeniería Electrónica y Eléctrica de la Universidad Nacional Mayor de San Marcos. Tesis. Perú. Universidad Nacional Mayor de San Marcos (UNMSM). Como consecuencia del análisis de las especificaciones técnicas que definen la arquitectura de una Red Definida por Software se ha propuesto y simulado el funcionamiento y comportamiento con tráfico IPv4 e IPv6 de una red SDN para la Facultad de Ingeniería Electrónica y Eléctrica de la Universidad Nacional Mayor de San Marcos que facilitará la enseñanza de los alumnos y les permitirá afrontar los retos del ámbito profesional conociendo la tecnología a la cual se están dirigiendo las redes de datos de las compañías operadoras más importantes del país.

Cuba, G. y Becerra, J. (2015). Diseño e implementación de un controlador SDN/OpenFlow para una red de campus académica. Tesis. Perú. Pontificia Universidad Católica del Perú. Nos dice en sus conclusiones que: El diseño del controlador soporta una migración total de la Red PUCP a SDN, evitando así el uso de un Router centralizado y consecuentemente, las desventajas de la red actual.

2.2. Bases teóricas vinculadas a la variable o variables de estudio

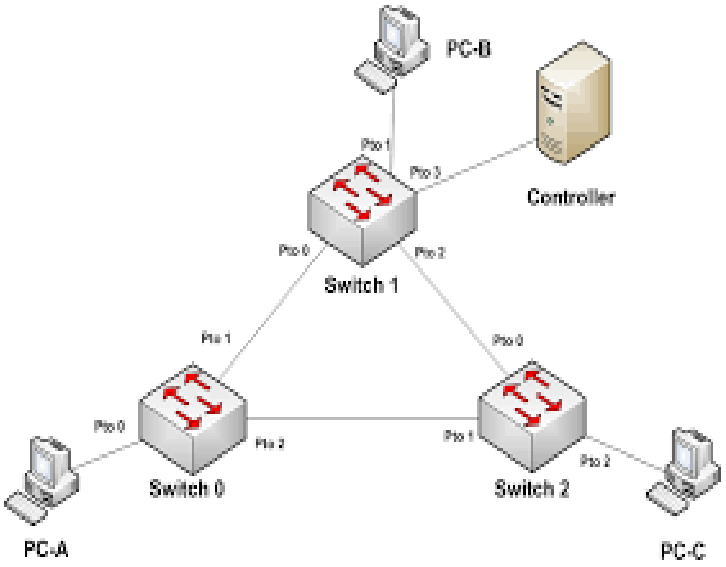
2.2.1. Red definida por software (SDN)

Según Rodriguez, C. (2014) define que las Redes Definidas por Software (SDN), también conocidas como SDN (Software Defined Network), son una tecnología que separa el plano de datos del plano de control y utiliza controladores para gestionar la transferencia de información en los interruptores de redes cableadas. En la actualidad, existen numerosos controladores SDN disponibles, tanto de código abierto como comercial, lo que plantea la cuestión de cuál elegir en el contexto de las Redes Definidas por Software.

En este artículo se examinan detalladamente las características más destacadas de los controladores SDN y se presentan algunos aspectos importantes a considerar para su selección y evaluación. En la figura 1 se puede ver un ejemplo de una topología de una red definida por software (SDN).

Figura 1

Topología de una red definida por software (SDN).



Nota. https://jci.uniautonoma.edu.co/2018/2018-7.pdf

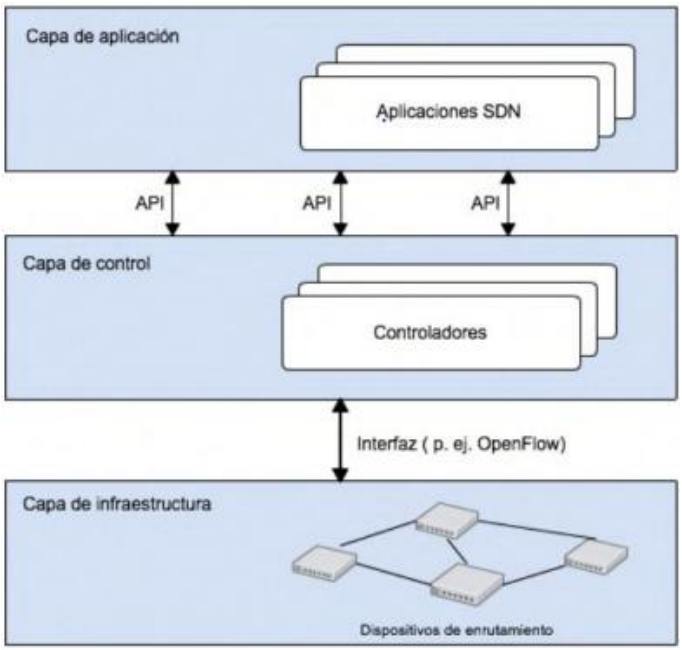
2.2.2. Arquitectura de una red definida por software (SDN)

Se buscó mucha información sobre la arquitectura de una red definida por software (SDN), la cual hay una definición muy certera que es la del autor Ramon Jesus Millan Tejedor (2014) donde manifiesta que arquitectura de una red definida por software consta de 3 capas, la capa infraestructura donde vemos los dispositivos de enrutamiento, la capa de control donde tenemos los controladores y la capa de aplicación que conecta mediante API con la capa de control y así poder usar las aplicaciones de una SDN.

En la figura 2 se muestra las tres capas de una arquitectura SDN.

Figura 2

Capas de una arquitectura de red definida por software (SDN)



*Nota.*https://www.ramonmillan.com/tutoriales/sdnredesinteligentes.php#arquitectura

- ***Capa de infraestructura de una Red definida por Software (SDN)***

Según José Óscar Romero Martínez, en su trabajo Seguridad en Redes definidas por software (SDN) (2021) nos dice que la primera capa en la arquitectura SDN es la de infraestructura y se emplea para el enrutamiento de un conjunto de paquetes basados en los dispositivos de red que forman parte de la infraestructura de red. Su función principal consiste en llevar a cabo las acciones de reenvío de paquetes de flujo de acuerdo con las directrices proporcionadas por el controlador y en informar sobre el estado de la red cuando las aplicaciones de red lo soliciten. En esta capa, existen varios protocolos disponibles, principalmente en el protocolo OpenFlow, ya que es ampliamente reconocido y es el más utilizado por grandes empresas, como Google. Los dispositivos de red más comunes que se encuentran en esta capa son los switches y los routers. Por lo general, un switch opera en la capa de enlace (L2).

Dado que la funcionalidad de una red definida por software (SDN) basado en OpenFlow es definida por un controlador de red basado en software, nos referimos a todos los dispositivos de red como switches.

Estos elementos de infraestructura son esenciales para el funcionamiento de la SDN y se pueden dividir en los siguientes componentes:

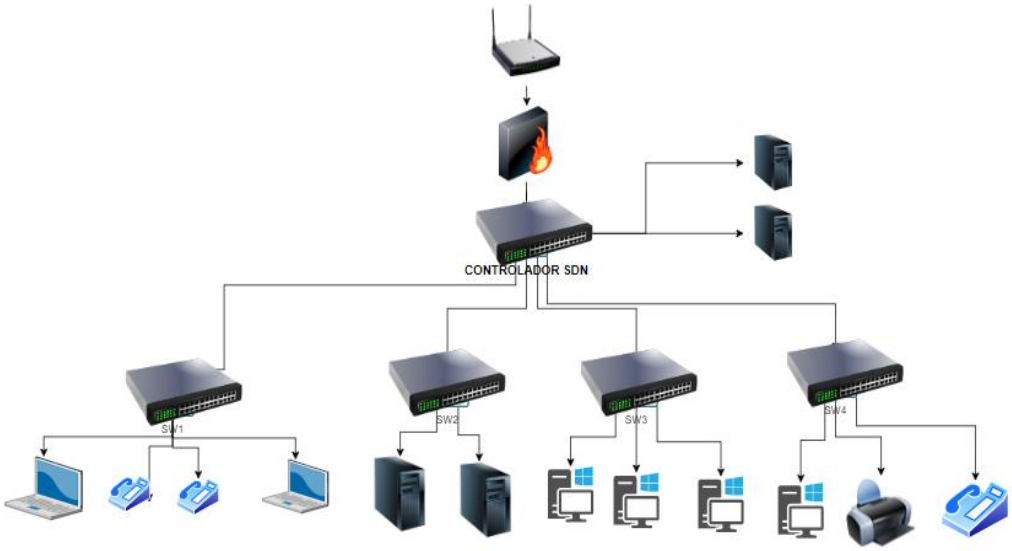
a) Dispositivos de Red Físicos

Estos son los componentes físicos de la red, como conmutadores (switches) y enrutadores, que forman parte de la infraestructura de la red. En una red SDN, estos dispositivos son configurados y gestionados de manera centralizada por el controlador SDN, lo que permite una mayor flexibilidad en el enrutamiento y la gestión del tráfico.

En la figura 3 podemos ver un ejemplo de dispositivos de red físicos que pueden conformar una Red definida por Software (SDN).

Figura 3

Dispositivos de red físicos que conforman una Red definida por Software (SDN).



Nota. Elaboración propia.

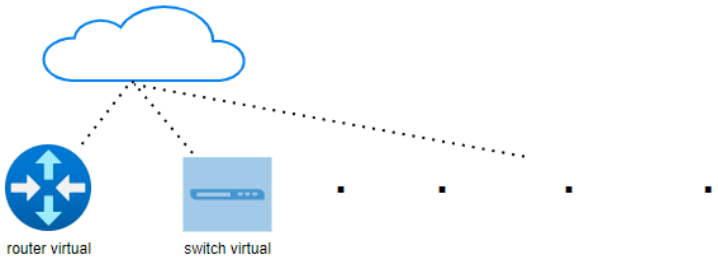
b) Dispositivos de Red Virtualizados

Además de los dispositivos físicos, una red SDN puede incluir dispositivos de red virtualizados, como conmutadores virtuales (vSwitches) y enrutadores virtuales. Estos elementos pueden ser desplegados y gestionados de manera flexible en entornos de virtualización y nube.

En la figura 4 podemos ver un ejemplo de dispositivos de red virtualizados que pueden conformar una Red definida por Software (SDN).

Figura 4

Dispositivos virtualizados que conforman una Red definida por Software (SDN).



Nota. Elaboración propia.

c) Enlaces de Comunicación

Los enlaces de comunicación son las conexiones físicas o lógicas que conectan los dispositivos de red en la infraestructura. En una red SDN, estos enlaces pueden ser configurados y gestionados de manera dinámica para adaptarse a las necesidades del tráfico y las políticas de la red.

d) Controlador SDN

Aunque el controlador SDN se encuentra en la capa de control, es un componente esencial de la infraestructura, ya que desempeña un papel central en la gestión y configuración de los dispositivos de red físicos y virtuales.

e) Elementos de Red Tradicionales

En muchas implementaciones de SDN, coexisten con elementos de red tradicionales, como dispositivos heredados que no son compatibles con SDN. El controlador SDN puede interactuar con estos elementos mediante adaptadores y protocolos de comunicación específicos.

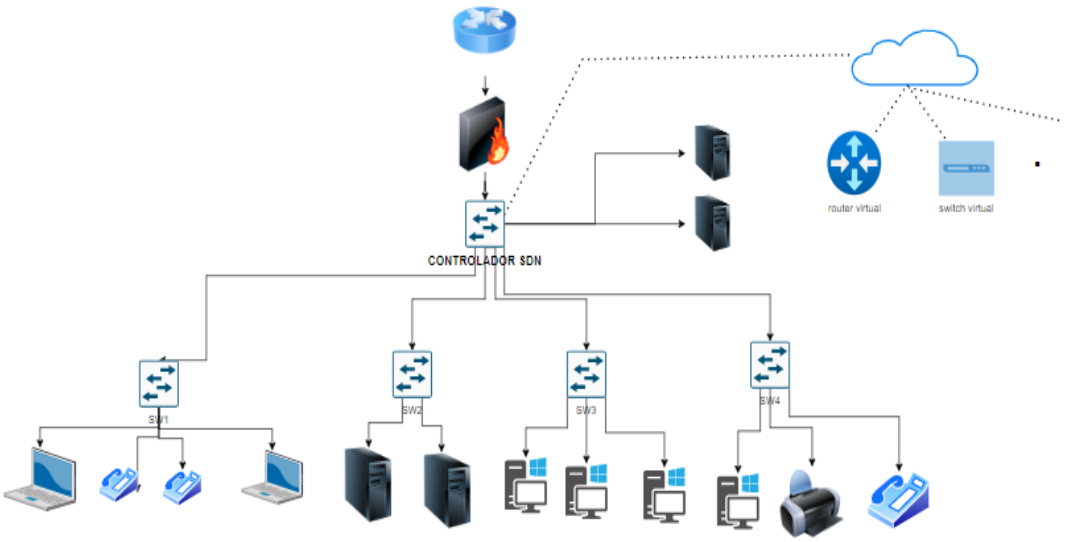
f) Red de Transporte

La infraestructura de la red SDN también depende de la red de transporte subyacente, que proporciona la conectividad física entre los dispositivos de la red SDN. Esto puede incluir enlaces de fibra óptica, cables de cobre, enrutadores y otros componentes de red. Es decir, la capa de infraestructura de una red SDN está compuesta por los dispositivos de red físicos y virtuales, los enlaces de comunicación, el controlador SDN y, en algunos casos, elementos de red tradicionales. La gestión centralizada y la programabilidad de esta infraestructura son características clave de las redes SDN, lo que permite una mayor flexibilidad y eficiencia en la gestión del tráfico y la implementación de políticas de red.

En la figura 5 se muestra un ejemplo de una red de transporte para una Red definida por software (SDN).

Figura 5

Red de transporte para una Red definida por Software.



Nota. Elaboración propia.

- ***Capa de control de una SDN***

Johan Eduardo Cáceres Guevara Arquitectura y funcionamiento de redes definidas por software (SDN) (2021) nos define que el componente central más crucial en la arquitectura SDN es el controlador de red, ya que proporciona una visión centralizada de la red y administra tanto la capa de aplicación como la capa de datos o infraestructura. El control de una configuración SDN se basa en uno o más controladores SDN que emplean APIs abiertas para ejercer su control sobre los switches de la red. Además de definir las reglas de reenvío para los switches, los controladores también supervisan el entorno, lo que les permite tomar decisiones de reenvío integradas con la gestión del tráfico en tiempo real.

Los controladores interactúan con el resto de la infraestructura de SDN utilizando las interfaces de comunicación, comúnmente denominadas interfaces sur (Southbound) y norte (Northbound) y los parámetros de la capa de control, las cuales son:

a) Controlador SDN

El controlador SDN es el componente central de la capa de control. Es el cerebro de la red SDN y se encarga de tomar decisiones de enrutamiento y políticas de tráfico. Puede

comunicarse con los dispositivos de red a través de protocolos de control como OpenFlow o APIs propietarias.

b) Aplicaciones SDN

Las aplicaciones SDN son programas o módulos que se ejecutan en la capa de control y utilizan la información proporcionada por el controlador para implementar políticas de red específicas. Estas aplicaciones pueden ser desarrolladas por terceros para adaptarse a necesidades específicas de la red.

c) Interfaz del controlador

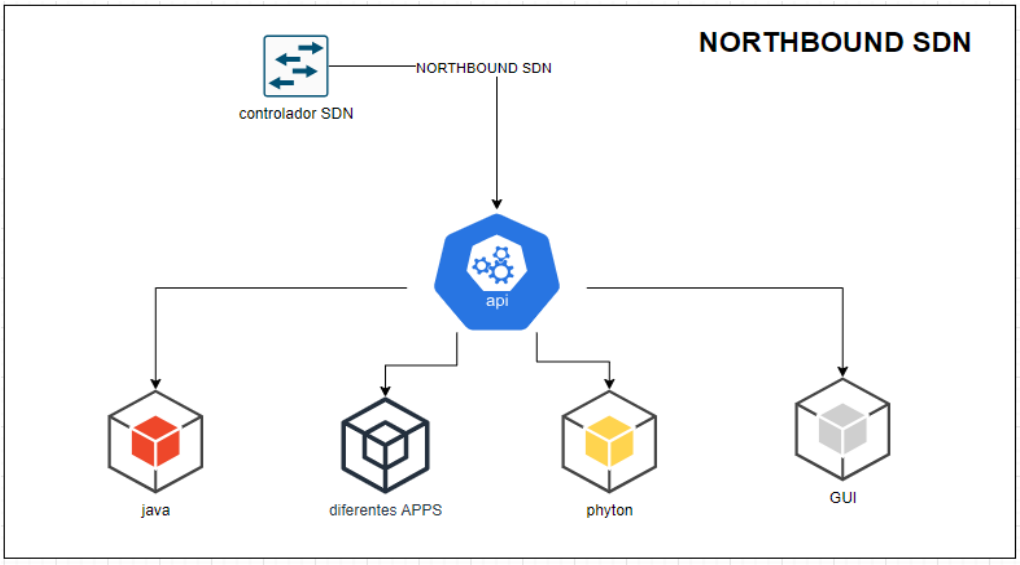
c.1. Interfaz Norte del Controlador (Northbound API)

Esta interfaz permite que las aplicaciones se comuniquen con el controlador SDN. Proporciona un conjunto de comandos y funciones que las aplicaciones pueden utilizar para configurar y controlar el comportamiento de la red.

En la figura 6 vemos un ejemplo de comunicación de la interfaz norte del controlador.

Figura 6

Comunicación de la interfaz norte del controlador.



Nota. Elaboración propia.

La interfaz Northbound sirve como un medio de acceso al controlador SDN en sí, permitiendo que un administrador de red lo utilice para llevar a cabo la configuración o recuperación de datos. Esta funcionalidad se puede realizar mediante una interfaz gráfica de usuario (GUI) o, alternativamente, aprovechando una API que facilita el acceso de otras aplicaciones al controlador SDN. Este enfoque posibilita la creación de scripts y la

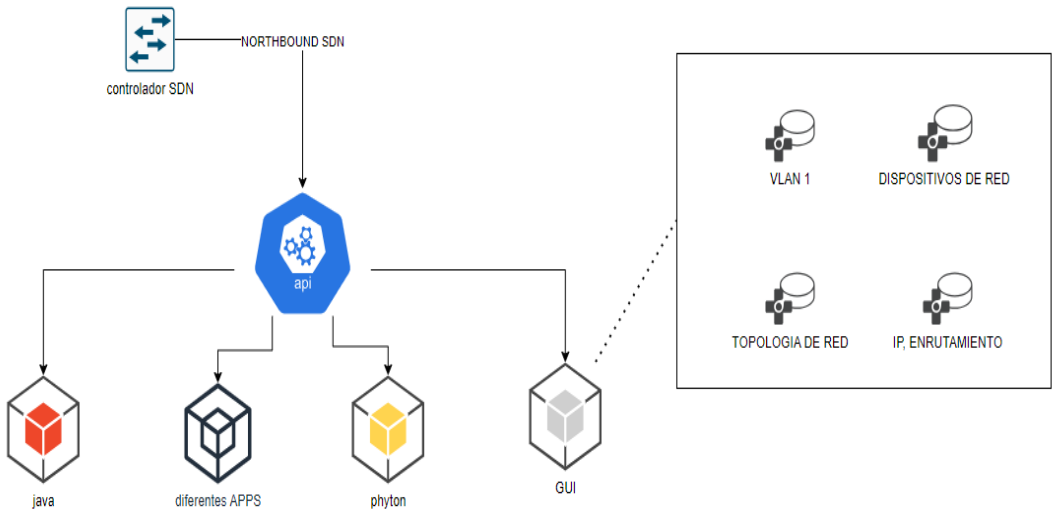
automatización de tareas de administración de la red. A continuación, se presentan ejemplos concretos de lo que se puede lograr con esta capacidad:

- Obtener un listado completo de la información de todos los dispositivos de red en la infraestructura.
- Visualizar el estado actual de todas las interfaces físicas en la red.
- Agregar una nueva VLAN de manera uniforme en todos los conmutadores de la red.
- Representar gráficamente la topología de la red en su totalidad.
- Configurar de manera automática las direcciones IP, la configuración de enrutamiento y las listas de acceso cuando se crea una nueva máquina virtual.

En la figura 7 podemos apreciar la funcionalidad de la interfaz gráfica de usuario (GUI).

Figura 7

Funcionalidad de la interfaz gráfica de usuario (GUI).



Nota. Elaboración propia.

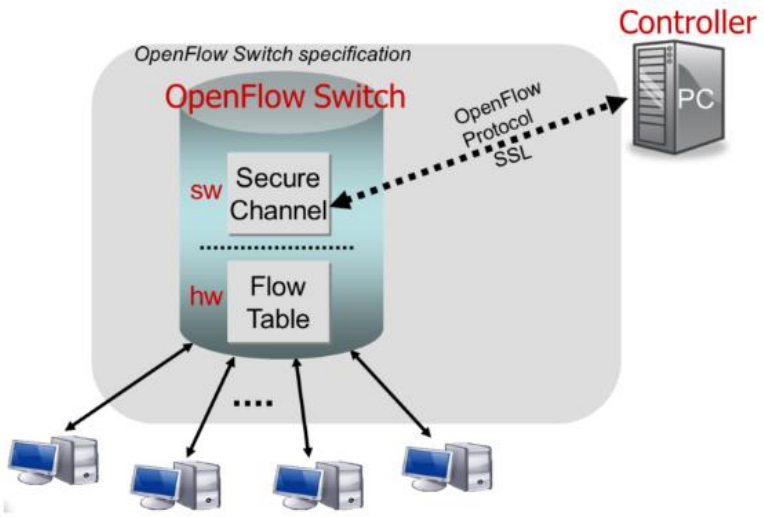
c.2. Interfaz Sur del Controlador (Southbound API)

Esta interfaz permite que el controlador SDN se comunique con los dispositivos de red, como switches y routers. Los protocolos como OpenFlow son ejemplos de interfaces que permiten al controlador programar y gestionar estos dispositivos.

En la figura 8 se muestra el funcionamiento de la interfaz sur del controlador.

Figura 8

Interfaz Sur del Controlador.



*Nota. https://*www.fiber-optic-cable-sale.com/will-sdn-change-future-network.html

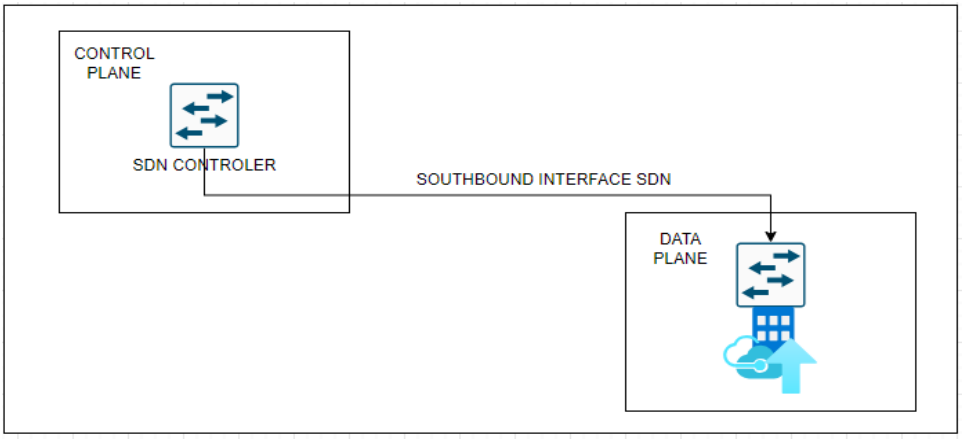
d) Base de Datos de Estado de la Red

El controlador SDN mantiene una base de datos que almacena información sobre el estado actual de la red. Esto incluye información sobre topología, flujos de tráfico, configuración de dispositivos y otras métricas relevantes.

En la figura 9 podemos observar el almacenamiento del estado de la red en la base de datos.

Figura 9

Flujo de la base de datos de la red.



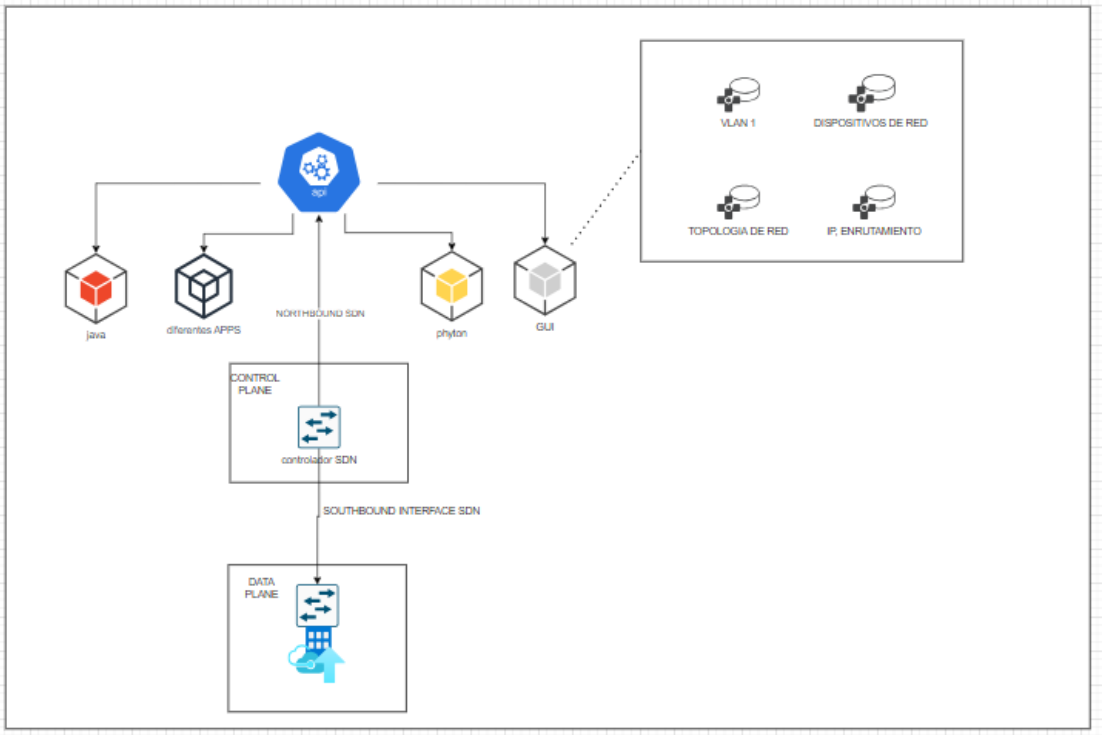
Nota. Elaboración propia.

Es decir, la capa de control de una red SDN está compuesta por un controlador central, aplicaciones que aprovechan ese controlador para implementar políticas específicas, interfaces de programación para la comunicación con aplicaciones y dispositivos de red, y una base de datos que almacena información sobre el estado de la red. Esta arquitectura permite una gestión y control de red más flexible y dinámica en comparación con las redes tradicionales.

En la figura 10 podemos ver la arquitectura de la capa de control.

Figura 10

Arquitectura de la capa de control.



Nota. Elaboración propia.

- ***Capa de aplicación de una SDN***

Según David Andrés Serrano Carrera en su investigación de Redes Definidas por Software (SDN): OpenFlow (2015), la capa de aplicación SDN se comunica con la red a través de una interfaz de programación de aplicaciones (API) que se conecta con la capa de control, y están diseñadas para cumplir con los requisitos de los usuarios. Los ejemplos de aplicaciones SDN:

a) Enrutamiento adaptable

Tradicionalmente, el enrutamiento ha sido complejo y lento en su convergencia. Con

SDN, han surgido dos conceptos populares: el balance de carga, que utiliza varios algoritmos para resolver el alto costo de los balanceadores de carga dedicados, y el diseño de intercambio de información entre capas para mejorar la integración entre entidades y proporcionar calidad de servicio (QoS).

b) Itinerancia sin interrupciones

La transferencia o "handover" al usar dispositivos móviles requiere un servicio continuo. Con SDN, las redes entre diferentes operadores con diversas tecnologías pueden compartir un plano de control común. Se han propuesto diferentes enfoques de transferencia, como Hoolock para redes Wi-Fi y WiMAX.

c) Mantenimiento de la red

Las herramientas de configuración típicas, como traceroute o tcpdump, no son adecuadas para mantener automáticamente una red extensa debido a su susceptibilidad al error humano. SDN proporciona una visión global de la red y un control centralizado de la configuración.

d) Seguridad de la red

Las redes tradicionales emplean cortafuegos o servidores proxy para proteger las redes físicas, lo que puede ser una tarea laboriosa para los administradores. SDN permite el análisis de patrones de tráfico para detectar posibles problemas de seguridad, como ataques de denegación de servicio (DDoS), y puede dirigir paquetes sospechosos a sistemas de prevención de intrusiones (IPS), modificar reglas de reenvío para bloquear tráfico no deseado o garantizar la privacidad de los usuarios.

e) Virtualización de la red

SDN facilita la existencia de múltiples arquitecturas de red heterogéneas en una infraestructura compartida. Esto se logra separando la red física en múltiples instancias virtuales y asignándolas a diferentes usuarios, controladores o aplicaciones mediante túneles, etiquetas VLAN y MPLS.

f) Computación en la nube (Cloud Computing)

Los centros de datos en las redes para la computación en la nube requieren características como escalabilidad, independencia de la ubicación para proporcionar recursos dinámicos y diferenciación de QoS. La conmutación virtual, implementada en SDN a través de soluciones como Open vSwitch, se utiliza para la comunicación entre máquinas virtuales en el mismo host.

2.2.3. Protocolo OpenFlow

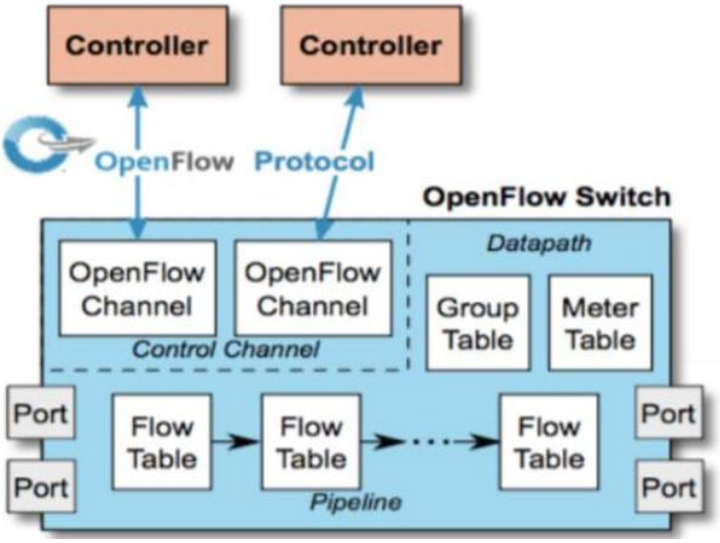
Hay un texto muy importante sobre las redes definidas por software de Johan Eduardo Cáceres Guevara (2021) que nos dice que OpenFlow fue propuesto inicialmente por Nick McKeown en 2008 y posteriormente estandarizado por la Open Networking Foundation (ONF) en 2011. Su desarrollo tiene como objetivo establecer una comunicación estandarizada entre el switch OpenFlow y el controlador basado en software en la arquitectura SDN. Esto permitió la programación de las tablas de flujo por parte de las aplicaciones de software.

- Los switches que son compatibles con OpenFlow conforman el plano de datos.
- En el plano de control, se encuentran uno o más controladores OpenFlow.
- La conexión entre el plano de control y los interruptores se establece mediante un canal de control seguro, conocido como la interfaz OpenFlow.

En la figura 11 se puede observar cómo funciona el protocolo OpenFlow junto al switch OpenFlow.

Figura 11

Protocolo OpenFlow y switch OpenFlow.



Nota. https://upcommons.upc.edu/bitstream/handle/2099.1/21633/Memoria.pdf

El protocolo OpenFlow es un estándar abierto utilizado en las redes definidas por software (SDN) para la comunicación entre el controlador SDN y los dispositivos de red, como los switches OpenFlow.

a) Versión de OpenFlow

Especifica la versión del protocolo OpenFlow que se está utilizando, como OpenFlow

1.0, 1.3, 1.5, etc. La versión determina las capacidades y características del protocolo que están disponibles.

b) Conexión al Controlador

Los parámetros de conexión al controlador SDN incluyen la dirección IP del controlador y el número de puerto en el que el controlador está escuchando para las conexiones entrantes de los dispositivos de red.

c) Mensajes OpenFlow

El protocolo OpenFlow utiliza varios tipos de mensajes para la comunicación entre el controlador y los switches OpenFlow. Algunos de los mensajes más comunes incluyen:

- Mensajes de solicitud

Enviados por el controlador para solicitar información o realizar acciones en los switches, como solicitar estadísticas de tráfico o modificar reglas de flujo.

- Mensajes de respuesta

Enviados por los switches para responder a las solicitudes del controlador.

- Mensajes de flujo

Utilizados para agregar, modificar o eliminar reglas de flujo en los switches. Estos mensajes incluyen información sobre cómo los paquetes deben ser procesados por el switch.

- Mensajes de estado

Proporcionan información sobre el estado del switch, como estadísticas de puertos y tablas de flujo.

d) Reglas de Flujo

Las reglas de flujo especifican cómo los paquetes deben ser procesados por un switch. Esto incluye información como el campo de coincidencia (por ejemplo, dirección MAC, dirección IP, puerto de entrada), la acción a realizar (reenviar a un puerto específico, descartar, modificar, etc.), y la prioridad de la regla.

e) Timeout de Reglas

Establece cuánto tiempo debe mantenerse activa una regla de flujo antes de ser eliminada automáticamente si no se actualiza.

f) Grupos de Flujos

Los grupos de flujos permiten la agrupación de múltiples flujos en una entidad única. Esto puede utilizarse para simplificar la gestión de políticas de tráfico más complejas.

g) Prioridades de Flujos

Se pueden asignar prioridades a las reglas de flujo para determinar el orden en que se aplican cuando hay múltiples reglas que coinciden con un paquete.

h) Acciones Personalizadas

Algunas implementaciones de OpenFlow permiten la definición de acciones personalizadas que pueden ser utilizadas en las reglas de flujo para realizar acciones específicas no estándar.

Los detalles precisos pueden variar según la versión del protocolo y la implementación específica del controlador y los switches OpenFlow utilizados en la red.

- ***Switch OpenFlow***

Hay un documento muy importante sobre Programación de redes SDN mediante el controlador POX de Pablo Yagües Fernández (2015) que nos dice que un Switch OpenFlow (OFS) está compuesto por una o más tablas de flujo y una tabla de grupos, así como un canal OpenFlow (TCP/SSL) en las capas 4 y 5 que se utiliza para la comunicación entre el controlador y el plano de control del enrutador. Para llevar a cabo todas estas funciones, el Switch OpenFlow requiere una serie de componentes, los cuales se detallan a continuación.

a) Dirección IP del Controlador

La dirección IP del controlador SDN al que el switch OpenFlow se conecta para recibir instrucciones y enviar información de estado.

b) Puerto del Controlador

El número de puerto en el que el controlador SDN está escuchando para las conexiones de los switches OpenFlow.

c) Tabla de Flujo

La capacidad de definir y configurar tablas de flujo en el switch. Esto puede incluir el número de tablas, prioridades, acciones permitidas, y criterios de coincidencia para las reglas de flujo.

d) Flujos Estáticos

La capacidad de agregar flujos estáticos al switch, que pueden ser configurados directamente en el switch y no necesitan ser instalados por el controlador.

e) Timeout de Reglas

Los valores de tiempo para el tiempo de vida de las reglas de flujo en el switch, especificando cuánto tiempo deben mantenerse antes de ser eliminadas.

f) Detección de Fallas

Configuraciones relacionadas con la detección y manejo de fallas en el switch, como la frecuencia de los mensajes de keep-alive al controlador.

g) Control de Ancho de Banda

Algunos switches OpenFlow permiten configurar políticas de control de ancho de banda para regular el tráfico.

h) Calidad de Servicio (QoS)

Configuración de políticas de calidad de servicio para priorizar ciertos tipos de tráfico sobre otros.

i) Configuración de Puertos

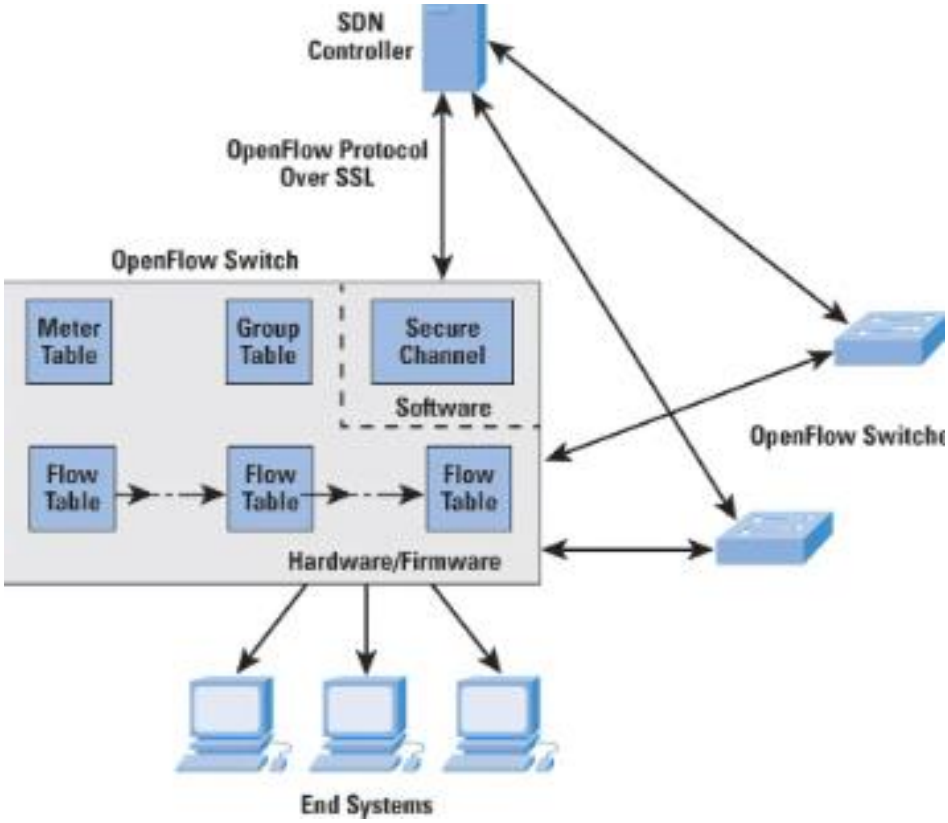
Parámetros relacionados con la configuración de puertos individuales, como velocidad, duplicación de puertos, y control de acceso.

Estos son los parámetros que pueden configurarse en un switch OpenFlow. La disponibilidad y la funcionalidad específica pueden variar según el modelo y el fabricante del switch, así como según la versión de OpenFlow que admita. La configuración se realiza generalmente a través de la interfaz de administración del switch o mediante comandos de línea de comandos en el CLI, dependiendo de la implementación.

En la figura 12 podemos ver un ejemplo del funcionamiento del switch OpenFlow y algunos de sus parámetros.

Figura 12

Switch Openflow y sus parámetros.



Nota. https:// www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-59/community.cisco.com/t5/networking/ct-p/4461-network-infrastructure

2.2.4. Controladores OpenFlow

Según el profesor Daniel Diaz Ataucuri en su libro Arquitectura de Redes Virtuales (SDN-NFV) (2022) nos presenta los controladores OpenFlow, los cuales están definidos a continuación.

- ***RYU***

Daniel Diaz Ataucuri en su libro Arquitectura de Redes Virtuales (SDN-NFV) (2022) nos comenta que el controlador Ryu es un controlador SDN de código abierto que implementa el protocolo OpenFlow y se utiliza para gestionar y controlar redes definidas por software. OpenFlow es un protocolo que permite la programación y control centralizado de dispositivos de red, como conmutadores y enrutadores, en una SDN.

Estas son algunas de las características y conceptos clave del controlador Ryu:

a) Controlador OpenFlow

Ryu es un controlador SDN que implementa el protocolo OpenFlow para comunicarse con dispositivos de red compatibles con OpenFlow. Esto significa que Ryu puede programar y gestionar el comportamiento de los conmutadores OpenFlow en la red.

b) Código Abierto

Ryu es un proyecto de código abierto bajo la licencia Apache 2.0, lo que significa que su código fuente es accesible y modificable por la comunidad de desarrolladores. Esto promueve la colaboración y la adaptación de Ryu a diferentes necesidades y entornos de red.

c) Programabilidad

Una de las principales ventajas de Ryu es su capacidad para programar el comportamiento de la red de manera flexible. Los desarrolladores pueden crear aplicaciones personalizadas que se ejecutan en el controlador Ryu y utilizan las API proporcionadas para definir políticas de red, implementar servicios y controlar el tráfico de acuerdo con sus requisitos específicos.

d) Soporte Multiplataforma

Ryu es compatible con múltiples sistemas operativos y plataformas, lo que facilita su implementación en diferentes entornos de red, incluyendo redes empresariales, de centros de datos y de proveedores de servicios.

e) API Abierta (Northbound API)

Ryu ofrece una API de programación abierta, lo que permite a los desarrolladores crear aplicaciones personalizadas que interactúen con el controlador para configurar y controlar la red.

f) Amplia Comunidad de Desarrolladores

Ryu cuenta con una comunidad activa de desarrolladores y usuarios que contribuyen al proyecto, lo que significa que hay una amplia base de conocimientos y soporte disponible.

g) Extensibilidad

Ryu es altamente extensible y permite la creación de módulos y componentes personalizados que se pueden agregar al controlador para ampliar su funcionalidad.

h) Rendimiento y Escalabilidad

Ryu está diseñado para ser eficiente en términos de rendimiento y escalabilidad, lo que

lo hace adecuado para implementaciones en redes de gran envergadura.

Ryu es un controlador SDN de código abierto que implementa el protocolo OpenFlow y se utiliza para gestionar y controlar redes definidas por software. Ofrece flexibilidad, programabilidad y extensibilidad, lo que lo convierte en una opción popular para aquellos que desean construir y personalizar soluciones de red SDN.

- ***ONOS***

Daniel Diaz Ataucuri en su libro Arquitectura de Redes Virtuales (SDN-NFV) (2022) nos comenta que el controlador ONOS (Open Network Operating System) es un controlador SDN de código abierto diseñado para administrar y controlar redes definidas por software. Es parte del proyecto ONF (Open Networking Foundation) y se enfoca en proporcionar un sistema operativo de red altamente escalable y flexible.
A continuación, se presentan algunas de las características y detalles clave de ONOS:

a) Controlador OpenFlow

ONOS es un controlador SDN que es compatible con el protocolo OpenFlow, lo que significa que puede comunicarse con conmutadores y dispositivos de red compatibles con OpenFlow para controlar su comportamiento.

b) Código Abierto

ONOS es un proyecto de código abierto bajo la licencia Apache 2.0. Esto significa que su código fuente es accesible públicamente y puede ser modificado y personalizado por la comunidad de desarrolladores.

c) Escalabilidad

ONOS está diseñado para ser altamente escalable y puede administrar redes de diferentes tamaños, desde redes de campus hasta redes de centros de datos y proveedores de servicios de gran escala.

d) Arquitectura Modular

ONOS se basa en una arquitectura modular que permite a los desarrolladores extender y personalizar sus funcionalidades. Esto facilita la adopción de nuevos servicios y aplicaciones.

e) Alta Disponibilidad

ONOS incluye características de alta disponibilidad para garantizar que la red se mantenga operativa incluso en situaciones de falla.

f) API Abierta

ONOS proporciona una API de programación abierta que permite a los desarrolladores crear aplicaciones SDN personalizadas y servicios de red. Esto facilita la creación de políticas de red y aplicaciones específicas.

g) Aplicaciones y Servicios

ONOS admite una variedad de aplicaciones y servicios, como enrutamiento, reenvío de paquetes, administración de ancho de banda, gestión de recursos, seguridad de red y más. Los desarrolladores pueden crear y agregar aplicaciones personalizadas según sea necesario.

h) Soporte para Diversos Protocolos

ONOS no se limita solo a OpenFlow; también es compatible con otros protocolos de comunicación y está diseñado para trabajar con una variedad de tecnologías de red.

i) Comunidad Activa

ONOS cuenta con una comunidad activa de desarrolladores y usuarios que contribuyen al proyecto y proporcionan soporte técnico.

j) Integración con Controladores Externos

ONOS puede integrarse con otros controladores SDN y sistemas de administración de red para lograr una gestión y control más completos de la infraestructura de red.

En conclusión, ONOS es un controlador SDN de código abierto que se enfoca en proporcionar un sistema operativo de red escalable y flexible. Ofrece una amplia gama de características y se utiliza para administrar redes definidas por software en diversos entornos, desde redes empresariales hasta infraestructuras de proveedores de servicios.

- ***OpenDayLight (ODL)***

Daniel Diaz Ataucuri en su libro Arquitectura de Redes Virtuales (SDN-NFV) (2022) nos comenta que el controlador OpenDaylight(ODL) es un controlador SDN (Software Defined Networking) de código abierto que se utiliza para gestionar y controlar redes definidas por software. Es uno de los proyectos más populares en el ámbito de SDN y está respaldado por la Fundación Linux.

A continuación, se presentan las características del controlador OpenDaylight:

a) Controlador OpenFlow y Multi-Protocolo

OpenDaylight es compatible con múltiples protocolos de comunicación, incluido OpenFlow. Esto significa que puede comunicarse con dispositivos de red que siguen diferentes estándares y protocolos, lo que le brinda flexibilidad para administrar una variedad de dispositivos.

b) Código Abierto

OpenDaylight es un proyecto de código abierto distribuido bajo la licencia Eclipse Public License (EPL). Su código fuente es público y accesible para la comunidad de desarrolladores, lo que fomenta la colaboración y la personalización.

c) Amplia Comunidad y Ecosistema

OpenDaylight cuenta con una gran comunidad de desarrolladores y usuarios que contribuyen al proyecto y desarrollan aplicaciones y soluciones basadas en ODL. Esto crea un ecosistema sólido y ofrece soporte y recursos adicionales.

d) Arquitectura Modular

OpenDaylight se basa en una arquitectura modular que permite a los desarrolladores personalizar y extender sus funcionalidades según sea necesario. Esto facilita la adaptación de ODL a diferentes entornos y casos de uso.

e) API Abierta (Northbound API)

OpenDaylight ofrece una API de programación abierta que permite a los desarrolladores crear aplicaciones personalizadas y servicios de red. Esta API facilita la configuración y el control de la red.

f) Aplicaciones y Servicios

OpenDaylight admite una variedad de aplicaciones y servicios, como enrutamiento, reenvío de paquetes, administración de ancho de banda, gestión de recursos, seguridad de red, administración de políticas y mucho más. Los desarrolladores pueden crear y agregar aplicaciones según sea necesario.

g) Soporte para Virtualización

OpenDaylight se integra bien con entornos de virtualización y nube, lo que facilita la gestión y la configuración de redes virtuales y físicas.

h) Escalabilidad

OpenDaylight está diseñado para ser escalable y puede manejar redes de diferentes

tamaños y complejidades, desde pequeñas redes empresariales hasta redes de proveedores de servicios a gran escala.

i) Seguridad

OpenDaylight incluye funciones de seguridad y autenticación para proteger el control y la configuración de la red.

j) Integración con Otros Proyectos SDN

OpenDaylight puede integrarse con otros proyectos y tecnologías de SDN para ofrecer una solución más completa y flexible de gestión de red.

En resumen, OpenDaylight es un controlador SDN de código abierto que ofrece una amplia gama de características y funcionalidades para la gestión y control de redes definidas por software. Su arquitectura modular y su compatibilidad con múltiples protocolos lo convierten en una opción popular para implementaciones de SDN en diversos entornos y casos de uso.

2.2.5. Virtualización

Luisa Fernanda Ulloa Z. Nos dice en su trabajo: La virtualización y su impacto en las ciencias computacionales; que la virtualización implica la instalación de sistemas operativos de manera virtual utilizando otro sistema llamado "anfitrión" o Host. Esto permite cargar diversos sistemas, incluso diferentes, de forma aislada para aprovechar al máximo el hardware del equipo y los recursos disponibles en el Host, como la conexión de red, puertos USB y unidades de almacenamiento, así como la capacidad de los procesadores. El único límite para este proceso es la capacidad del hardware del Host, y los avances recientes en virtualización han ampliado sus posibilidades, aunque posiblemente requiera personal más capacitado, pero sigue siendo accesible para muchos a través de Internet o Intranet.

a) Máquinas virtuales

Las máquinas virtuales (VM) son programas informáticos que simulan una computadora física o un servidor, posibilitando la ejecución de sistemas operativos y aplicaciones en un entorno virtual. En términos simples, una máquina virtual es una réplica independiente de una computadora física que funciona dentro de una máquina real.

b) Sistema operativo Ubuntu

Ubuntu es un sistema operativo de código abierto que se fundamenta en el núcleo de Linux.

c) LINUX

Linux es un sistema operativo de código abierto y gratuito basado en el núcleo del sistema operativo Unix. Es conocido por su estabilidad, seguridad y flexibilidad, y es ampliamente utilizado en una variedad de dispositivos y sistemas, desde servidores de gran envergadura hasta dispositivos embebidos y computadoras personales.

2.2.6. Mininet

Según Sergio Córdoba López (2019), nos dice que Mininet es un emulador de redes basada en Linux, que da la posibilidad de crear redes virtuales (switches, routers, hosts, etc) bajo el dominio de un controlador basado en SDN y que cuenta con las siguientes topologías:

a) Topología de Estrella

En una topología de estrella, un nodo central (generalmente un switch) está conectado a varios nodos periféricos (hosts). Todos los nodos periféricos se comunican a través del nodo central.

b) Topología de Anillo

En una topología de anillo, cada nodo está conectado a exactamente dos nodos vecinos, formando un anillo. Los datos circulan en una dirección en el anillo.

c) Topología en Árbol

En una topología en árbol, los nodos están organizados en una estructura jerárquica similar a un árbol. Puede haber un nodo raíz que se conecta a varios nodos secundarios, y cada nodo secundario puede tener sus propios nodos secundarios.

d) Topología Lineal

En una topología lineal, los nodos están conectados uno tras otro en una línea recta. Los datos se transmiten secuencialmente de un extremo al otro.

e) Topología de Malla Completa

En una topología de malla completa, cada nodo está conectado directamente a todos los demás nodos en la red. Esto proporciona una alta redundancia y múltiples rutas de comunicación.

f) Topología Personalizada

Mininet permite definir topologías personalizadas mediante programación. Esto significa que puedes crear topologías específicas para tus necesidades de experimentación.

g) Topología en Celda

En una topología en celda, los nodos están organizados en una cuadrícula o rejilla bidimensional. Cada nodo está conectado a sus vecinos inmediatos en la cuadrícula.

h) Topología en Estrella Distribuida

Similar a la topología de estrella, pero con múltiples nodos centrales, cada uno conectado a sus propios hosts periféricos.

i) Topología en Anillo Doble

Es una variante de la topología de anillo en la que hay dos anillos separados, y algunos nodos pueden estar conectados a ambos anillos.

j) Topología en Isla

En esta topología, los nodos se agrupan en subredes o "islas" separadas, y algunos nodos actúan como enrutadores que conectan estas islas.

2.2.7. Miniedit

Jackson Emilio Martínez Copete en su libro: Estudio del funcionamiento de la herramienta Mininet (2015), nos dice que se trata de una interfaz gráfica de usuario (GUI) que se introdujo a partir de la versión 2.2.0.1 y fue desarrollada en código Python. Esta GUI incluye los componentes esenciales de una red, como hosts (computadoras), switches, controladores, enlaces (conexiones virtuales de cable), routers y otros elementos. Esto brinda a los usuarios la capacidad de diseñar la topología de red seleccionando los dispositivos que desean incluir y guardarla para continuar trabajando en ella en el futuro. Además, ofrece la posibilidad de exportar la topología creada en formato de código Python.

2.2.8. Definición de términos básicos

- API

Application Programming Interface o Interfaz de programación de aplicaciones; permite a componentes de software comunicarse entre sí mediante diferentes y múltiples protocolos.

- Flujo de datos

Es la transferencia continua y secuencial de información desde una fuente a un destino por medio de un canal de comunicación.

- Tráfico IPv4

Tráfico de datos que utiliza el Protocolo de Internet versión 4 (IPv4).

- Tráfico IPv6

Tráfico de datos que utiliza el Protocolo de Internet versión 6 (IPv6).

- Plano de datos (data plane)

Transfiere y procesa el reenvío de paquetes de datos en función de lo que decide el plano de control mediante que estos fluyen en la red.

- Plano de control (control plane)

Es el encargado de tomar las decisiones sobre cómo se deben administrar y enviar los paquetes de datos en la red.

- Tablas de flujo

Se usan para establecer cómo los dispositivos de red reenvían los paquetes de datos de acuerdo a las políticas y reglas establecidas por el controlador SDN.

- Políticas de red

Conjunto de reglas que determinan cómo se comporta una red, con la finalidad de cumplir diferentes objetivos y/o requisitos.

- Entornos de virtualización

Infraestructuras tecnológicas donde se puede crear y usar Máquinas Virtuales (VM).

- QoS

Quality of Service o Calidad de servicio.

- EPL

Eclipse Public License o Licencia Pública Eclipse.

- GUI

Graphical User Interface o Interfaz Gráfica de Usuario.

- SSL

Secure Sockets Layer o Capa de Conexión Segura, que es un protocolo de seguridad para

internet que protege la privacidad de los datos entre cliente y servidor.

- ODL (OpenDayLight)

Proyecto que busca crear una plataforma común y abierta que permita a los administradores de red gestionar y controlar dispositivos de red de manera más eficiente y flexible.

- Convergencia

Tiempo que se toman los routers para intercambiar información entre sí.

- Balance de carga

Acción para aumentar eficientemente la capacidad de una red.

- Traceroute

Comando que muestra por donde pasan los datos desde el origen hasta su destino.

- Ataque de denegación de servicios (DDoS)

Es un ciber ataque que impide y/o bloquea el funcionamiento normal de un servidor.

- MPLS

Multiprotocol Label Switching o Conmutación de etiquetas multiprotocolo.

2.2.9. Tipo y nivel de investigación

- ***Tipificación de la investigación***

Alfonso González Damián (2003) nos dice que es importante destacar que hay 2 tipos de estudios, el básico y el aplicado y que, en ambos tipos de estudios, se parte de una situación problemática, aunque en cada caso se entiende, desarrolla y expresa de manera diferente. En la investigación básica, la problemática se relaciona con deficiencias, lagunas o vacíos en la teoría, lo que implica que el investigador se enfrenta al límite del conocimiento. Por otro lado, la problemática en la investigación aplicada está vinculada a dificultades, obstáculos, conflictos, carencias o necesidades prácticas, lo que obliga al investigador a abordar las necesidades de la sociedad, sus instituciones, empresas e individuos.

Con estas aclaraciones, es evidente que desde la formulación del problema se pueden identificar los objetivos que persigue la investigación y, por ende, el tipo al que pertenece. La elección entre uno u otro tipo depende de las prioridades de los organismos, instituciones y grupos de investigación, así como de las preferencias, estilo y experiencia del investigador en cuestión.

Por lo tanto, la presente investigación se enmarca en el ámbito de la investigación aplicada.

- ***Nivel de investigación***

Según el autor Piscoya (1987), se identifican varios tipos de investigación tecnológica, que son los siguientes:

a) Investigación en tecnologías físicas

Esta área se enfoca en la creación, mejora y optimización de máquinas, equipos, instrumentos, mecanismos, procedimientos y sistemas en campos como la ingeniería civil, agrícola, agronomía, ambiental, pesquera, industrial, minera, geotécnica, de petróleo, entre otros. También abarca la medicina, especialmente en lo que se refiere a la curación de enfermedades y, principalmente, a la cirugía y rehabilitación de los pacientes. Otros campos relacionados son la astronáutica, la farmacia y la odontología, entre otros.

b) Investigación en tecnología social

Esta categoría engloba las técnicas aplicadas a la pedagogía, la economía (incluyendo las tecnologías de la información y la comunicación, TIC) y otros campos como la informática, la administración, la planificación y lo relacionado con aspectos técnicos y legales. Estas investigaciones se basan en disciplinas como la psicología, la sociología, la antropología y la lingüística.

c) Investigación en tecnologías formales

Este campo abarca áreas como la programación de computadoras, el análisis de sistemas, la investigación operativa y la cibernética. Aquí es donde se ha desarrollado la tecnología algorítmica, ya que se fundamenta en teorías matemáticas. Las disciplinas que sustentan estas investigaciones son el cálculo de probabilidades, la teoría de grafos, la teoría de juegos, el álgebra de Boole, entre otras.

De acuerdo con el autor Piscoya (1987), el nivel de investigación de nuestra tesis es de tipo investigación en tecnologías formales.

2.2.10. Diseño de la investigación

- ***Diseño actual de la red de la universidad Ricardo Palma***

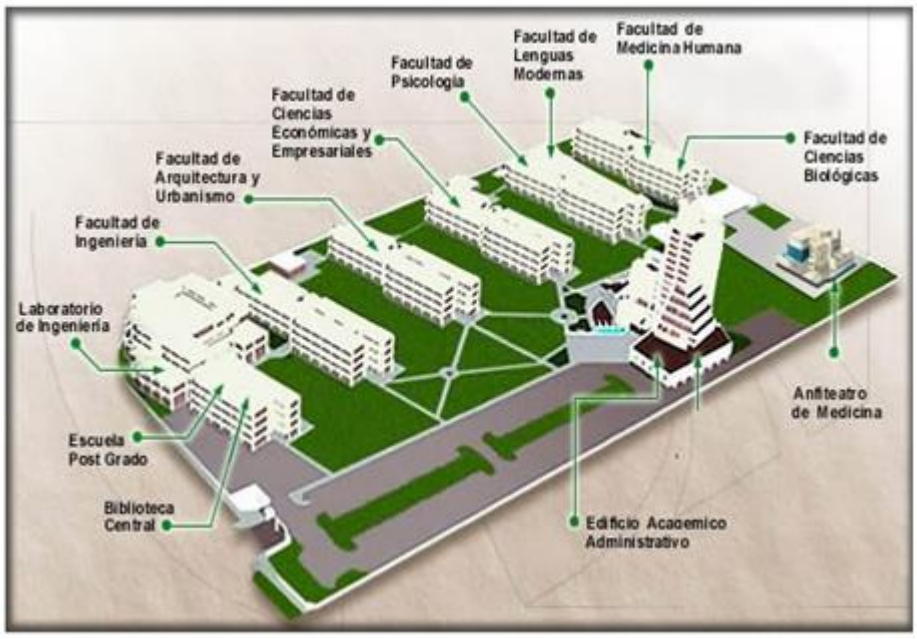
Tenemos unos 12.000 alumnos, cerca de 1.100 profesores y 600 trabajadores; además de miles de egresados que están en diferentes funciones dentro del Estado o el sector privado en el Perú y en el exterior.

Se cuenta con 8 facultades las cuales son Arquitectura, Ciencias Biológicas, Ciencias Económicas y Empresariales, Derecho y Ciencias políticas, Humanidades y lenguas modernas, Ingeniería, Medicina humana y Psicología. En cada facultad se cuenta con una biblioteca especializada y una biblioteca virtual.

En la figura 13 se puede apreciar la estructura de la universidad.

Figura 13

Estructura de la Universidad Ricardo Palma.



Nota. https://sites.google.com/a/spc.org.pe/cibse2015/home/universidad-ricardo-palma

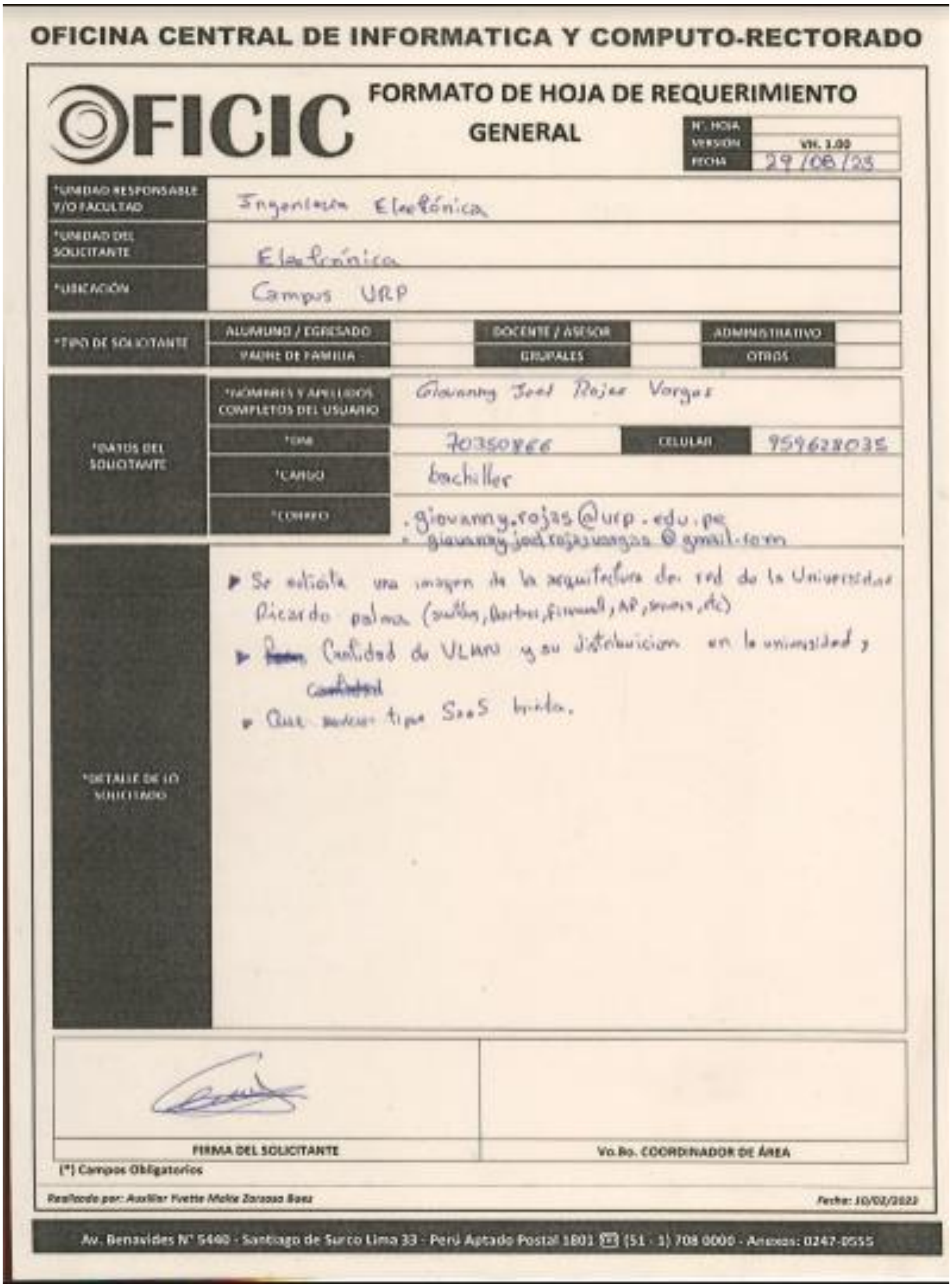
- ***Topología de red actual de la universidad Ricardo Palma***

Para tener acceso a la topología de red de la Universidad Ricardo Palma, se tuvo que solicitar el acceso al documento a la Oficina Central de Informática y Computo/Rectorado.

En la figura 14 se observa el documento de solicitud que se realizó.

Figura 14

Solicitud de acceso y/o permiso.



OFICINA CENTRAL DE INFORMATICA Y COMPUTO-RECTORADO

OFICIC

FORMATO DE HOJA DE REQUERIMIENTO GENERAL

N°. HOJA	
VERSION	VH. 1.00
FECHA	29/08/23

*UNIDAD RESPONSABLE Y/O FACULTAD	Ingeniería Electrónica
*UNIDAD DEL SOLICITANTE	Electrónica
*UBICACIÓN	Campus URP

*TIPO DE SOLICITANTE	ALUMUNO / EGRESADO		DOCENTE / ASESOR		ADMINISTRATIVO	
	PADRE DE FAMILIA		GRUPALES		OTROS	

*DATOS DEL SOLICITANTE	*NOMBRES Y APELLIDOS COMPLETOS DEL USUARIO	Giovanny Joel Rojas Vargas		
	*DNI	70350866	CELULAR	959628035
	*CARGO	bachiller		
	*CORREO	giovanny.rojas@urp.edu.pe		

*DETALLE DE LO SOLICITADO	Se solicita una imagen de la arquitectura de red de la Universidad Ricardo Palma (switches, routers, firewall, AP, servers, etc)

FIRMA DEL SOLICITANTE

Vo.Bo. COORDINADOR DE ÁREA

(*) Campos Obligatorios

Realizado por: Auxiliar Yvette Malca Zarzosa Baez

Fecha: 10/02/2022

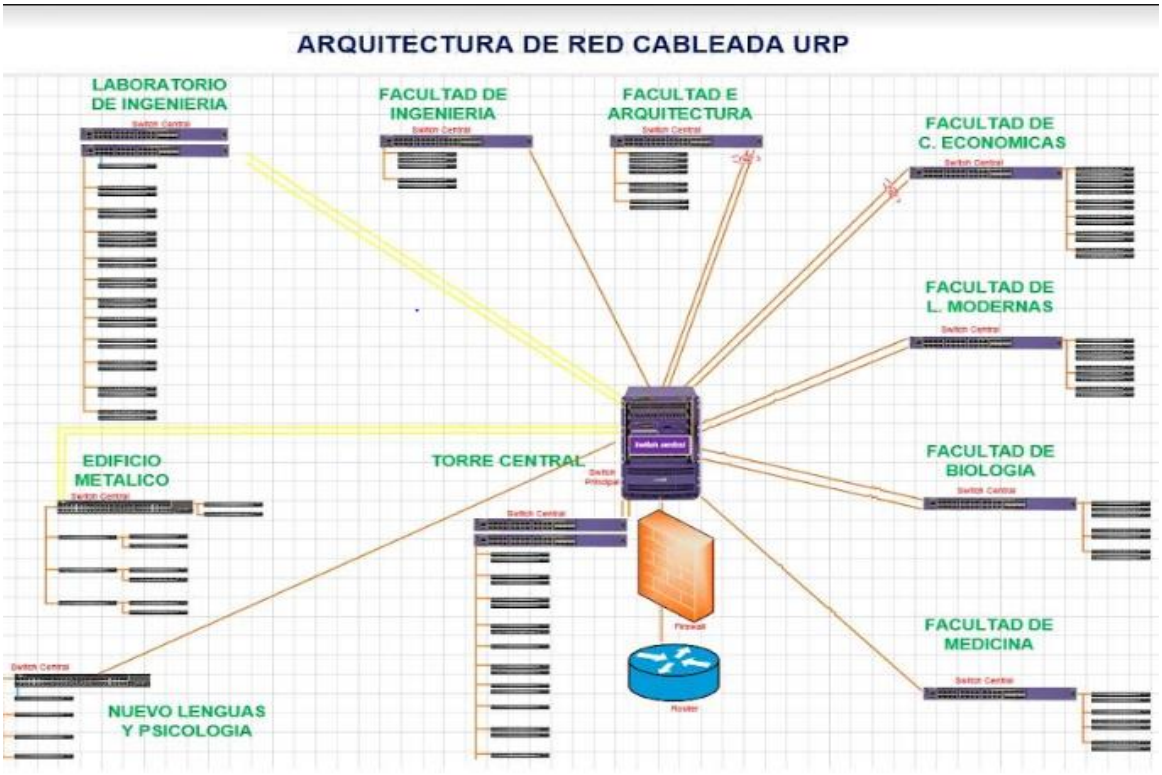
Av. Benavides N° 5440 - Santiago de Surco Lima 33 - Perú Aptado Postal 1801 (51 - 1) 708 0000 - Anexos: 0247-0555

Nota. Elaboración propia.

En la figura 15 se muestra la arquitectura de red cableada en el campus de la Universidad Ricardo Palma.

Figura 15

Arquitectura de red de la Universidad Ricardo Palma.



Nota. Elaboración propia.

- ***Vlans***

Distribuidas en 8 por cada facultad y 1 en el Edificio Metálico. En la Universidad existen 2 grupos de vlans: Vlans académicas y Vlans administrativas. Existen en total 37 vlans, siendo la distribución la siguiente: 24 vlans académicas y 13 vlans administrativas.

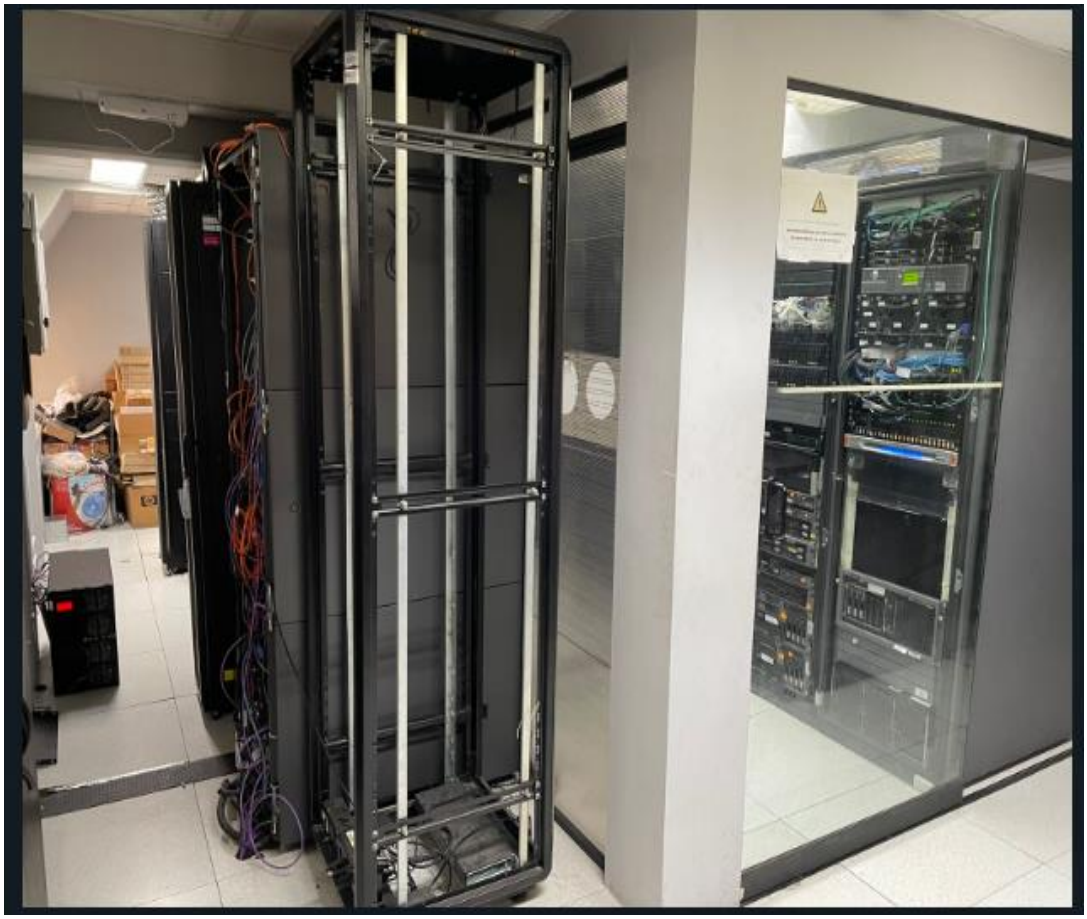
- ***Data Center***

La universidad cuenta con una gran tecnología en estructura de base de datos y en sus equipos y/o servidores que se encargan de administrar y gestionar la red del campus de la universidad.

En la figura 16, observamos un poco la forma física en el que está distribuido el data center de la Universidad Ricardo Palma.

Figura 16

Data center de la Universidad Ricardo Palma.



Nota. Elaboración propia.

CAPÍTULO III: METODOLOGIA DE LA INVESTIGACIÓN

3.1. Selección de controlador SDN

Para elegir el controlador adecuado se hizo un resumen de los 3 principales controladores de red controlado por software (SDN) con protocolo OpenFlow.

En el proyecto de tesis se eligió al controlador Opendaylight, debido a que tiene un dominio de aplicación objetivo y certero para la red de la Universidad Ricardo Palma, también cuenta con un lenguaje de programación conocido y potente el cual es JAVA, tiene integración con diferentes API's del exterior, se puede conectar desde una plataforma con sistema operativo tipo Linux , Windows y MAC en la cuales se basan los servidores y computadoras que conforman la arquitectura de red de capa 1 de la Universidad Ricardo Palma y también el tener información y documentación sobre el controlador Opendaylight fue un factor decisivo para la selección de dicho controlador.

En la figura 17 se ve una comparación de los controladores SDN, anteriormente mencionados, y la elección en amarillo del controlador elegido para esta tesis.

Figura 17

Comparación y selección del controlador SDN.

	CONTROLADORES SDN		
	RYU	OPENDAYLIGHT	ONOS
INTERFACES HACIA ABAJO	• Openflow 1.0 • Openflow 1.2 • Openflow 1.4	• Openflow 1.0 • Openflow 1.2 • Openflow 1.4 • Openflow 1.5	• Openflow 1.0 • Openflow 1.2 • Openflow 1.4 • Openflow 1.5
DOMINIO DE APLICACIÓN	• CAMPUS	• Data center • Transporte-SDWAN	• Data center • Transporte-SDWAN
SOPORTE TLS	SI	SI	SI
LENGUAJE DE PROGRAMACIÓN	• PYTHON	• JAVA	• JAVA
REST API	si	si	si
PATROCINADOR	• Nippon • Telegraph • telephone (NTT)	• Linux fundation • CISCO • IBM • más de 40 compañías	• TELECOM • CISCO • ERICCSON • FUJITSU • HUAWEI • INTEL • NEC
VIRTUALIZACIÓN	• MININET • OVS	• MININET • OVS	• MININET • OVS
INFORMACIÓN // DOCUMENTACIÓN	• medio	• muy bueno	• bueno
PLATAFORMA SOPORTADA	• LINUX	• LINUX • MAC • WINDOWS	• LINUX • MAC • WINDOWS

Nota. Elaboración propia.

3.2. Simulación de máquinas virtuales con VirtualBox

3.2.1. Descarga e instalación de VirtualBox

Se ingresó a la página de VirtualBox (Figura 18) y se descargó el ejecutable más reciente, en este caso se descargó el archivo .exe VirtualBox-7.0.10-158379-Win (Figura 19).

Figura 18

Página de VirtualBox.

Nota. https://www.virtualbox.org/wiki/Downloads

Figura 19

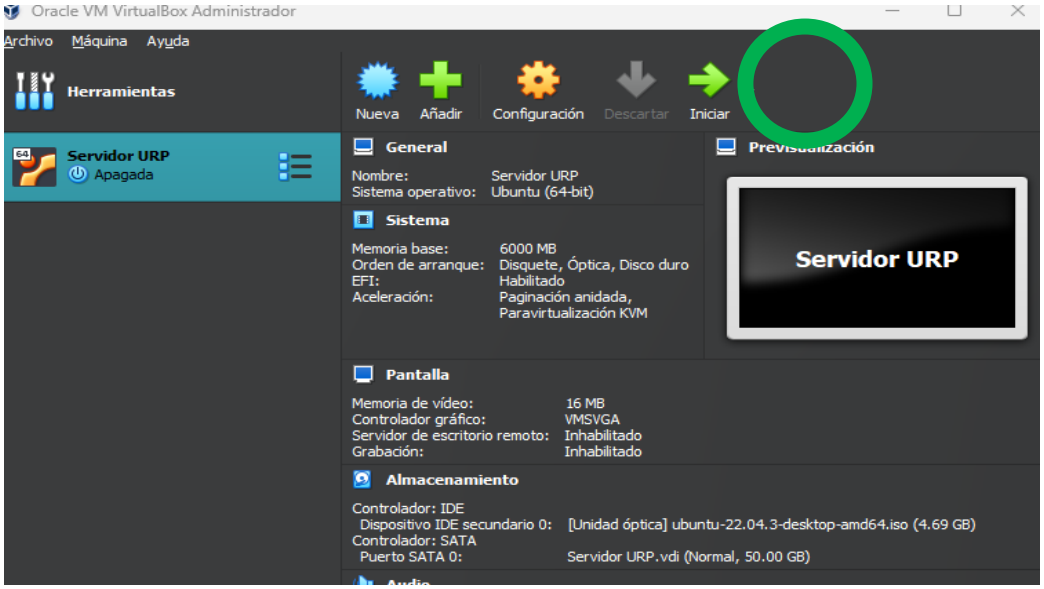
Ejecutable más reciente.

Nota. Elaboración propia.

Se ejecutó el archivo .exe y nos mostró una interfaz, donde podemos crear una nueva máquina virtual (Figura 20).

Figura 20

Interfaz donde creamos una nueva máquina virtual.



Nota. Elaboración propia.

3.2.2. Descarga e instalación de la imagen ISO Ubuntu 20.04.3-amd64 en VirtualBox.

Se descargó el Ubuntu-22.04.3-desktop-amd64 de la página oficial de Ubuntu, cual interfaz se ve en la figura 21.

Figura 21

Interfaz Ubuntu-22.04.3-desktop-amd64

Nota. https://cdimage.ubuntu.com/releases/focal/release/

Una vez descargado, se ve el archivo tipo .ISO (Figura 22), el cual se usó para ser instalado en la máquina virtual VirtualBox.

Figura 22

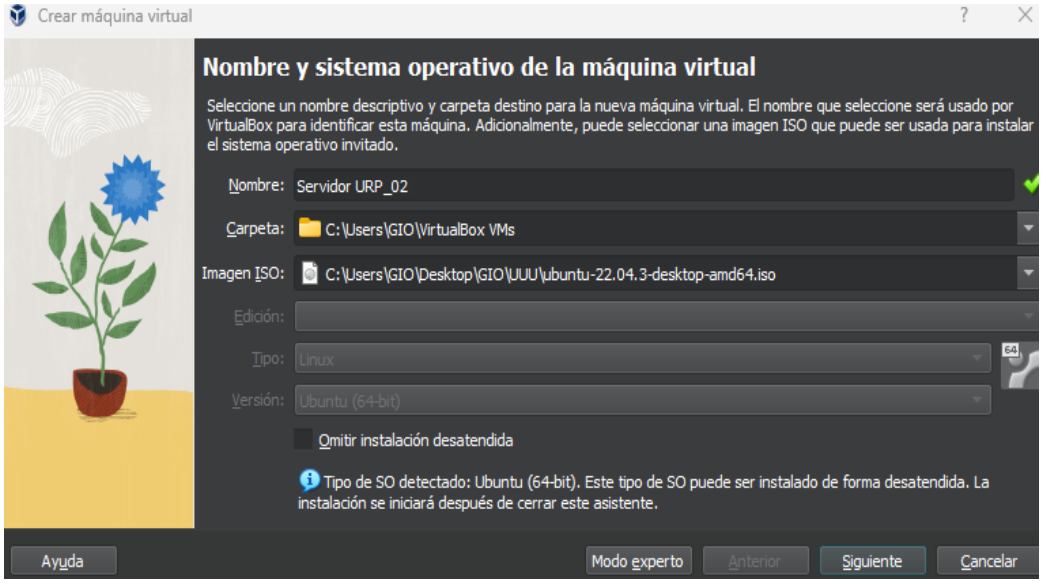
Archivo tipo .ISO

Nota. Elaboración propia.

En la figura 23 se creó la máquina virtual con sistema operativo Ubuntu, el cual cuenta con lenguaje Linux/GNU. El nombre del servidor fue “Servidor URP”, tipo Linux, versión Ubuntu (64-bit).

Figura 23

Máquina virtual con sistema operativo Ubuntu.

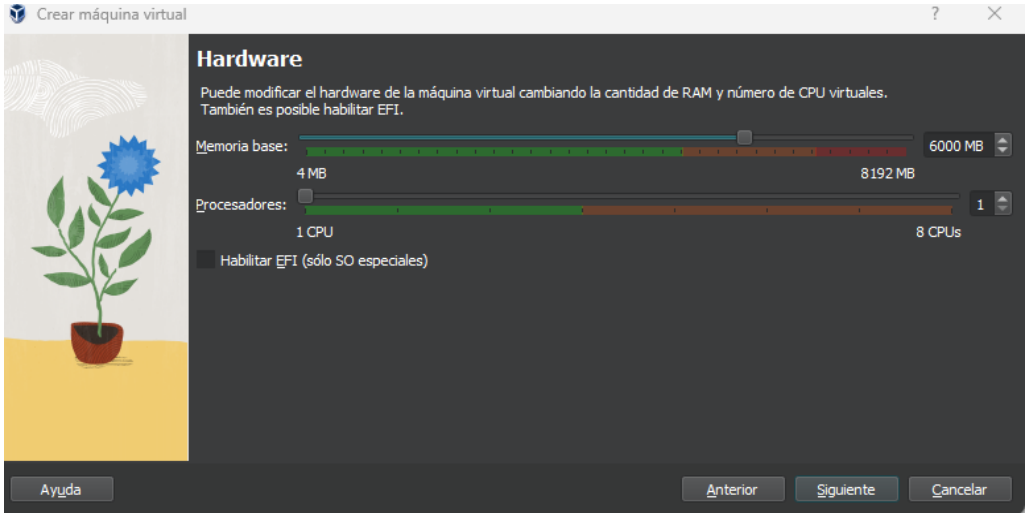


Nota. Elaboración propia.

La máquina virtual constó de una memoria base de 6000Mb y 1 procesador (CPU) y se habilitó el EFI (Figura 24).

Figura 24

Habilitación de EFI.

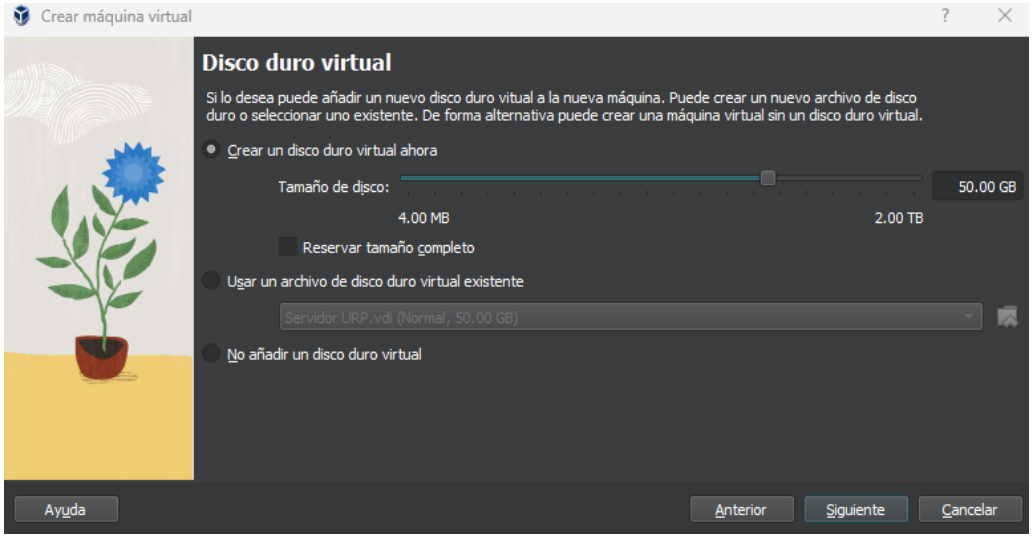


Nota. Elaboración propia.

Una vez especificado los términos de Hardware, se definió el tamaño del disco (Figura 25) que fue de 50.00 GB.

Figura 25

Tamaño del disco.

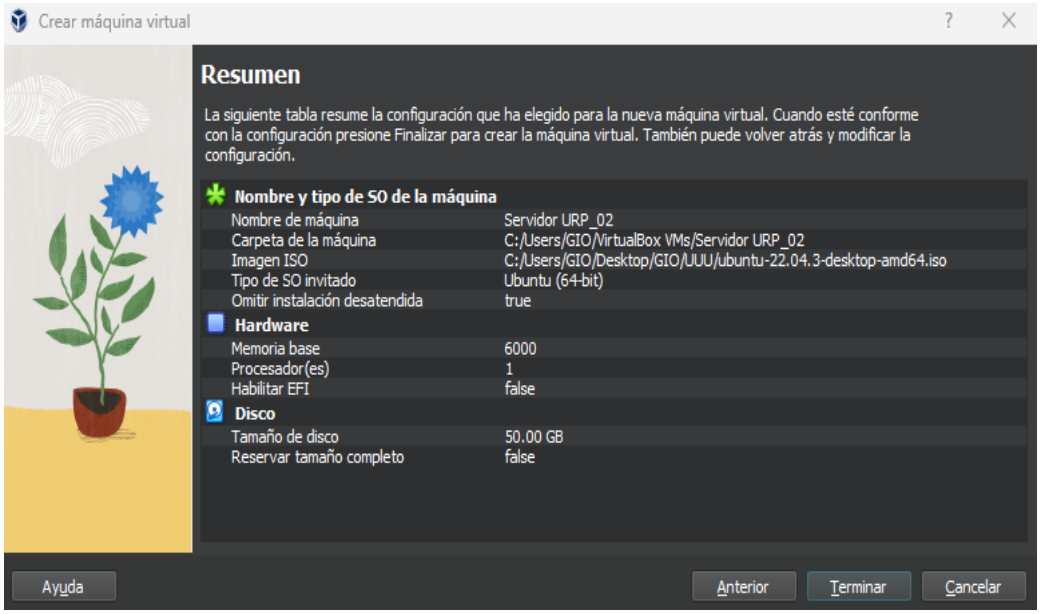


Nota. Elaboración propia.

Se verificó si tenemos los parámetros correctos y le damos en “Terminar” (Figura 26).

Figura 26

Verificación de los parámetros.



Nota. Elaboración propia.

3.2.3. Inicio de la máquina virtual "Servidor URP"

Se inició la máquina virtual (Figura 27), mientras se esperó la carga de la interfaz (Figura 28 y Figura 29) y finalmente se observó el escritorio de la máquina virtual que se creó (Figura 30).

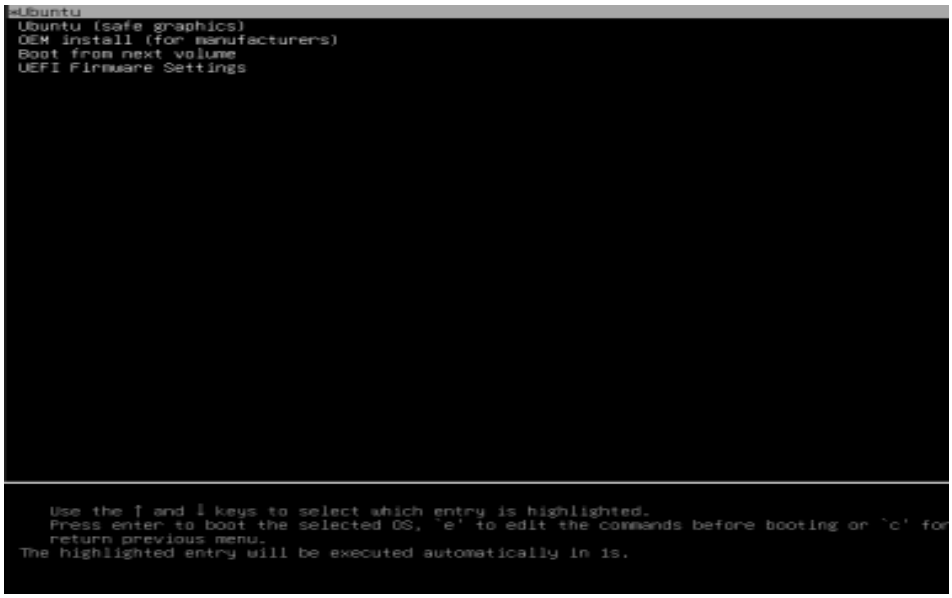
Figura 27

Inicio de máquina virtual.

Nota. Elaboración propia.

Figura 28

Carga de interfaz.



Nota. Elaboración propia.

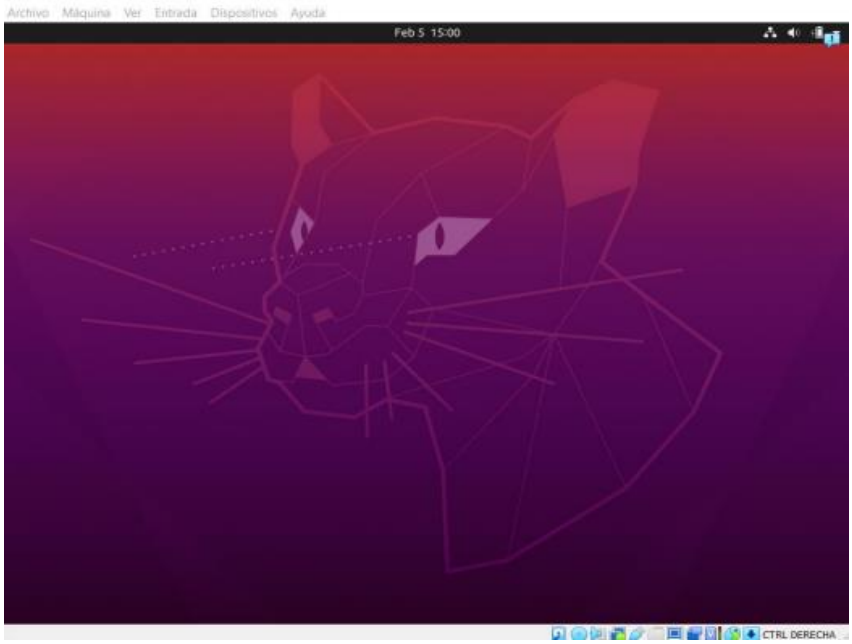
Figura 29

Carga de interfaz.

Nota. Elaboración propia.

Figura 30

Escritorio de la máquina virtual.



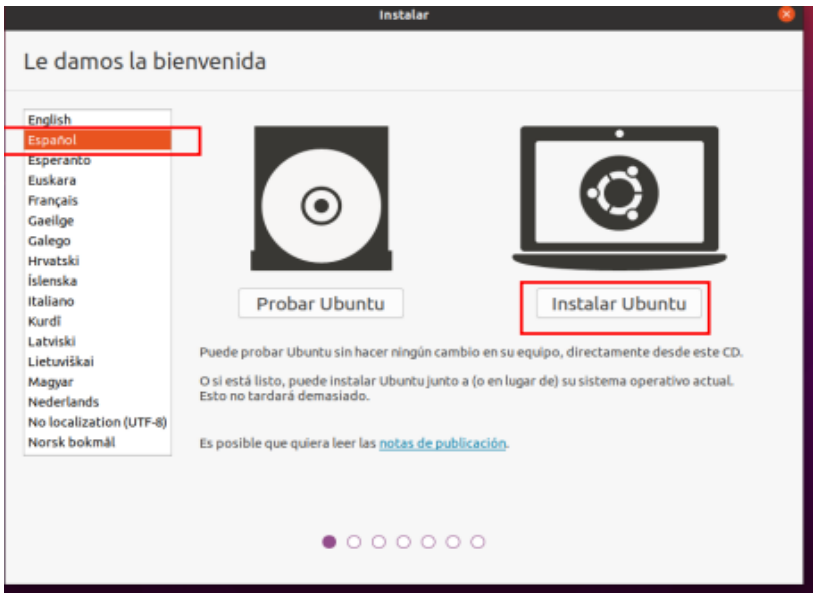
Nota. Elaboración propia.

3.2.4. Configuración de la máquina virtual.

Se inició la máquina virtual y se procedió a instalar el sistema operativo Ubuntu en español (Figura 31) y se seleccionó nuestra región Lima, Perú (Figura 32).

Figura 31

Sistema operativo Ubuntu en español.



Nota. Elaboración propia.

Figura 32

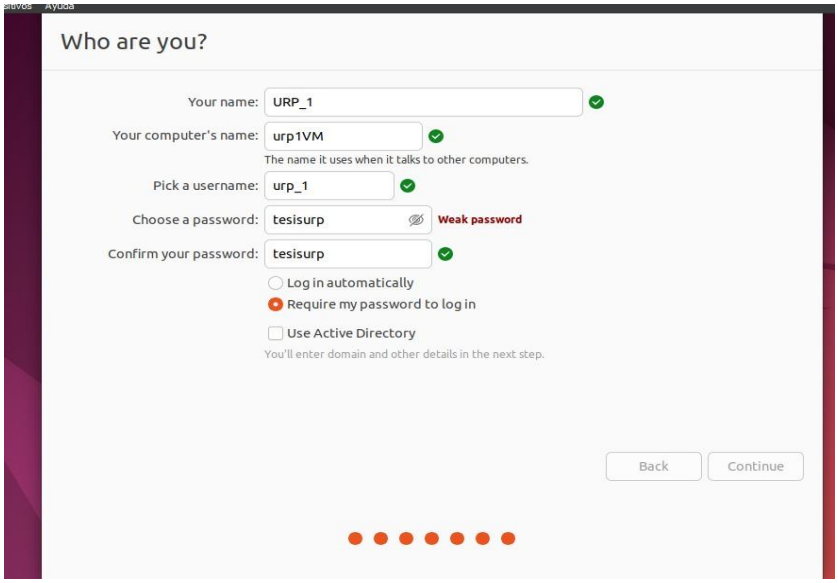
Selección de la región.

Nota. Elaboración propia.

En la figura 33 se procedió a seleccionar nuestro nombre, el nombre de la computadora, nombre de usuario y se eligió nuestra contraseña y a su vez se confirmó.

Figura 33

Selección de nombres y contraseña.

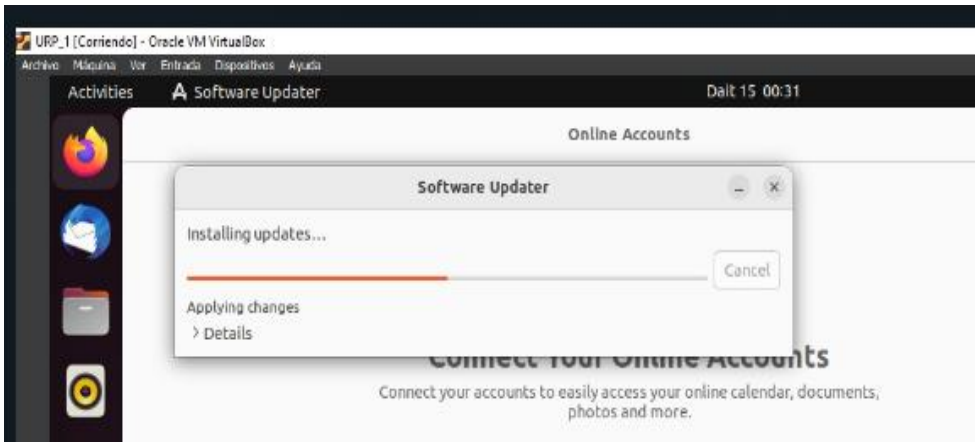


Nota. Elaboración propia.

Una vez configurado los requisitos que nos pide la máquina virtual, podemos ver el escritorio en la cual volvemos a instalar las actualizaciones (Figura 34).

Figura 34

Instalación de actualizaciones.

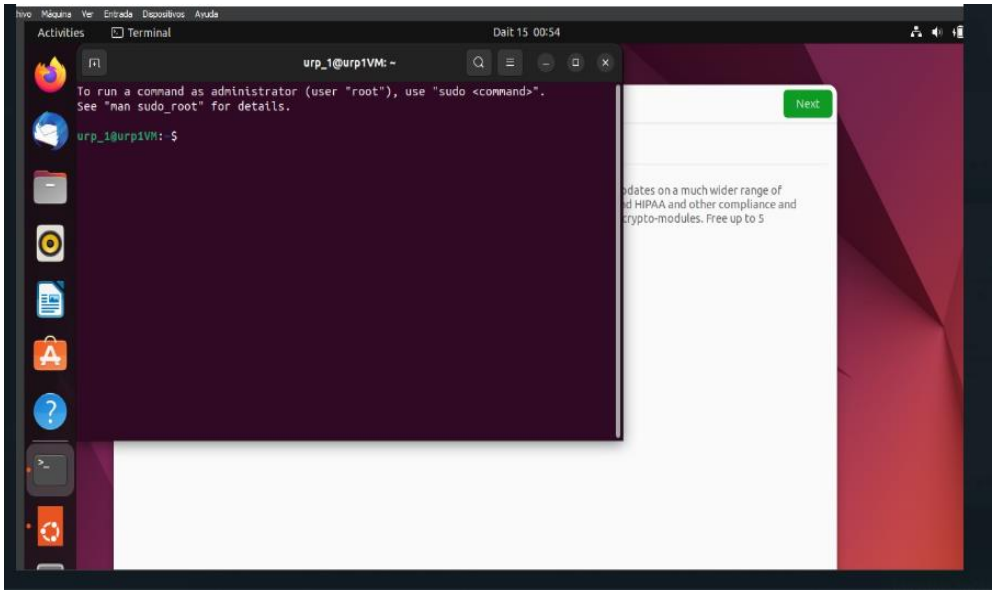


Nota. Elaboración propia.

Se activó la terminal de Ubuntu (Figura 35), la cual se usó para escribir los códigos de instalación de Mininet y del controlador Openflow.

Figura 35

Terminal de Ubuntu.

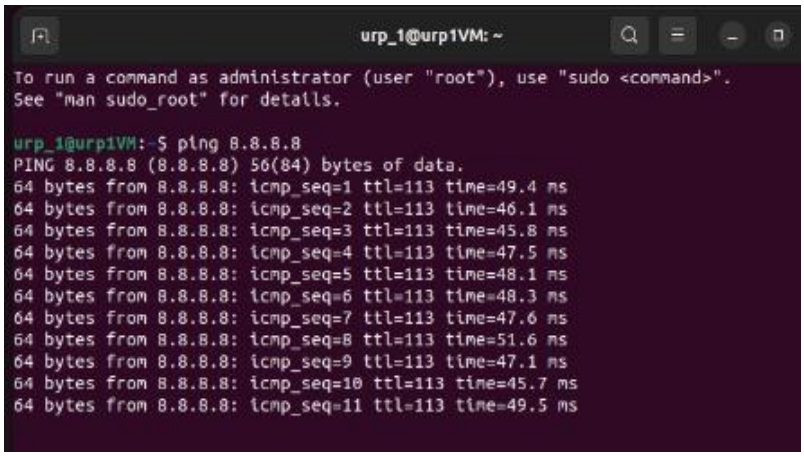


Nota. Elaboración propia.

En la figura 36 se observa que se hizo ping 8.8.8.8 para comprobar el acceso a internet desde la VM.

Figura 36

Comprobación del acceso a internet.



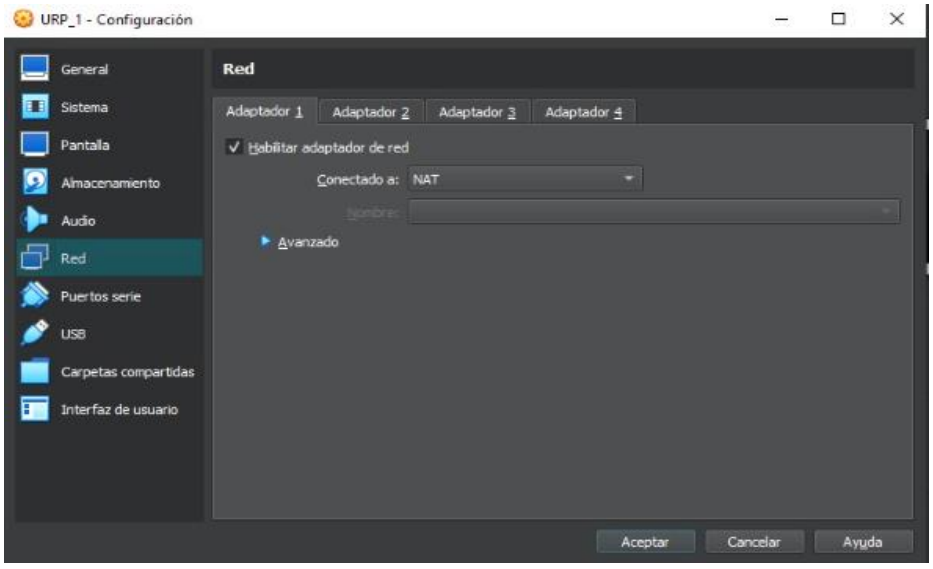
Nota. Elaboración propia.

3.2.5. Configuración de NAT

Se configuró el modo NAT desde la configuración de VirtualBox (Figura 37) la cual se necesitó para que los dispositivos en la red se comuniquen a través de internet.

Figura 37

Configuración de NAT.



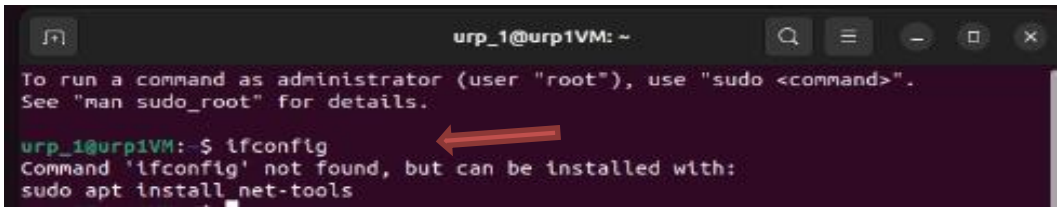
Nota. Elaboración propia.

3.2.6. Verificación de configuración de la máquina virtual.

Se instaló *ifconfig* en "Servidor URP" (Figura 38), el cual nos mostró todas las configuraciones de red en el sistema ya que por defecto *ifconfig* no está instalado en Ubuntu 20.04. por lo cual se usó el código *ifconfig* en la terminal.

Figura 38

Instalación de "ifconfig".

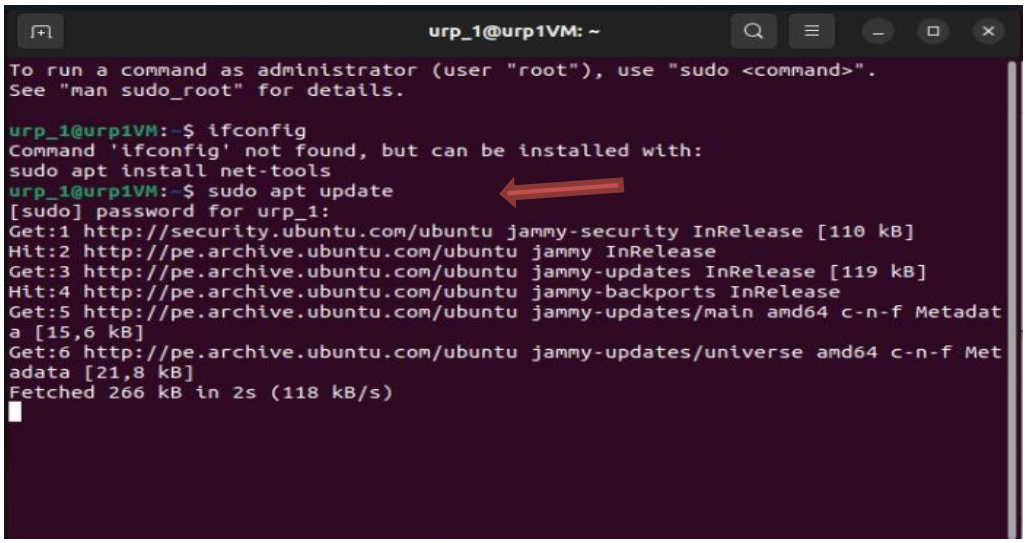


Nota. Elaboración propia.

Se ejecutó el código *sudo apt update* (Figura 39), el cual actualiza el programa del Ubuntu y *sudo apt install net-tools* (Figura 40) que nos permitió ver las configuraciones de red.

Figura 39

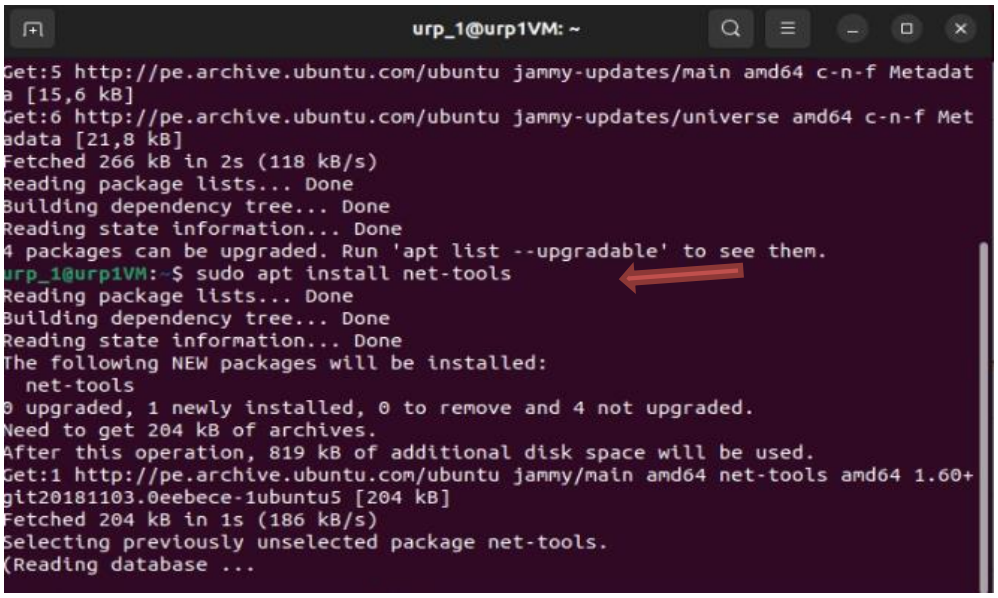
Ejecución del código sudo apt update.



Nota. Elaboración Propia.

Figura 40

Ejecución del código sudo apt install net-tools.

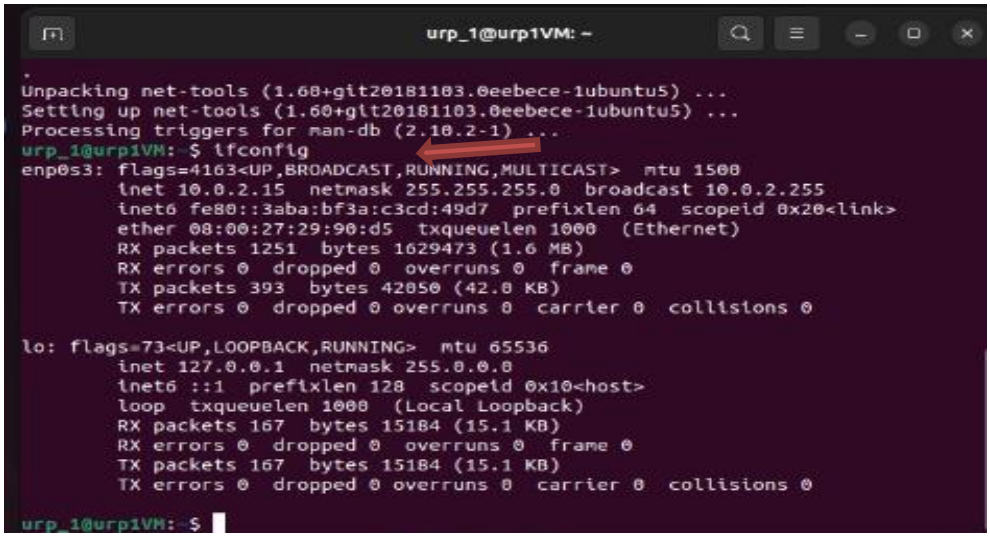


Nota. Elaboración propia.

Una vez ejecutado los 2 comandos, se ingresó el comando *ifconfig* para visualizar las direcciones ip de la interfaz de red (Figura 41). Desde este punto ya pudimos instalar Mininet.

Figura 41

Ejecución del comando "ifconfig".



Nota. Elaboración propia.

3.3. Instalación de Mininet

Mininet es el emulador que se usó para el despliegue de SDN. Se instaló mediante Linux en la máquina virtual creada.

3.3.1. Lista de comandos que se usó en la instalación de mininet en VirtualBox

- Apt

Se usó para administrar la instalación, actualización, eliminación y administración de paquetes de software en estos sistemas.

- Update

Se usó para actualizar la lista de paquetes disponibles en los repositorios de software

- Upgrade

Se usó para actualizar todos los paquetes instalados en el sistema a las versiones más recientes disponibles.

- Install

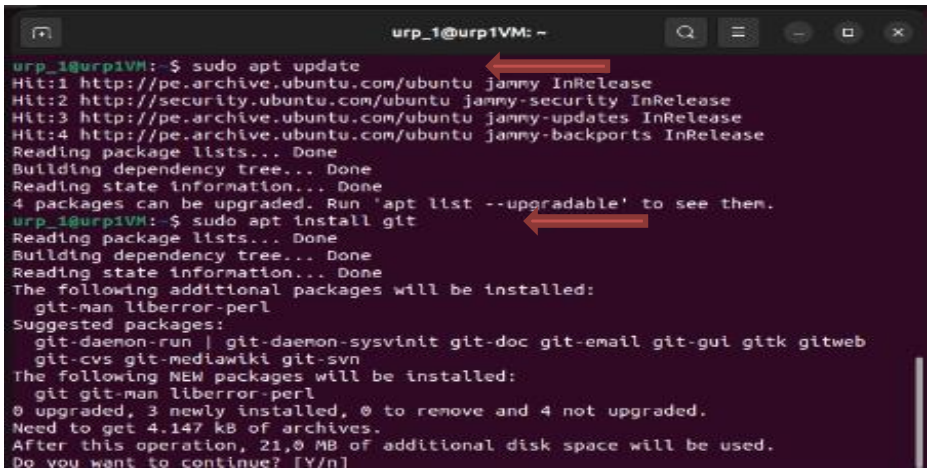
Se usó para instalar nuevos paquetes de software en el sistema. En el contexto de Mininet, se usó este comando para instalar Mininet y sus dependencias si aún no están instaladas.

3.3.2. Instalación de Mininet mediante el repositorio de GitHub

Git o GitHub nos ayudan a gestionar y versionar el código fuente del proyecto. Usaremos los códigos “Sudo apt update” y “Sudo apt install git” para instalar Git. En la figura 42 se muestra el proceso de instalación.

Figura 42

Proceso de instalación de Git.



Nota. Elaboración propia.

Se instaló Mininet de forma nativa desde GitHub mediante el código *sudo git clone* como se muestra en la figura 43.

Figura 43

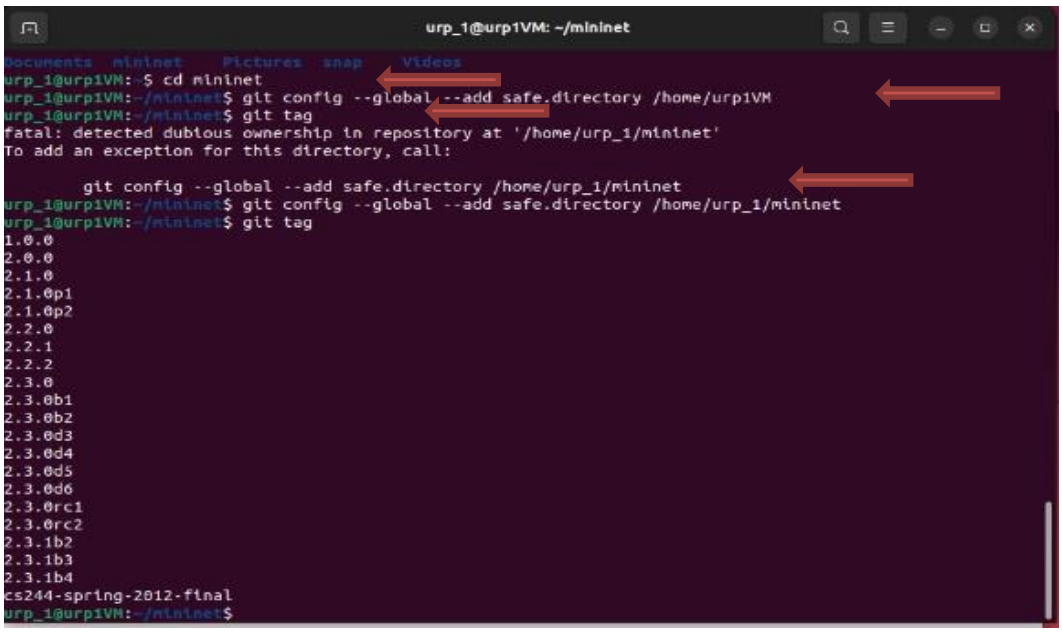
Instalación de Mininet desde GitHub.

Nota. Elaboración propia.

Luego, como se ve en la Figura 44, se ingresó al directorio mediante el código *cd mininet* y con el comando "*git config --global --add safe.directory /home/urp_1/mininet* se ingresó a mininet, posteriormente se ejecutó el código *git tag* y se visualizó la versión disponible que se usó para el proyecto.

Figura 44

Ingreso al directorio y a Mininet.

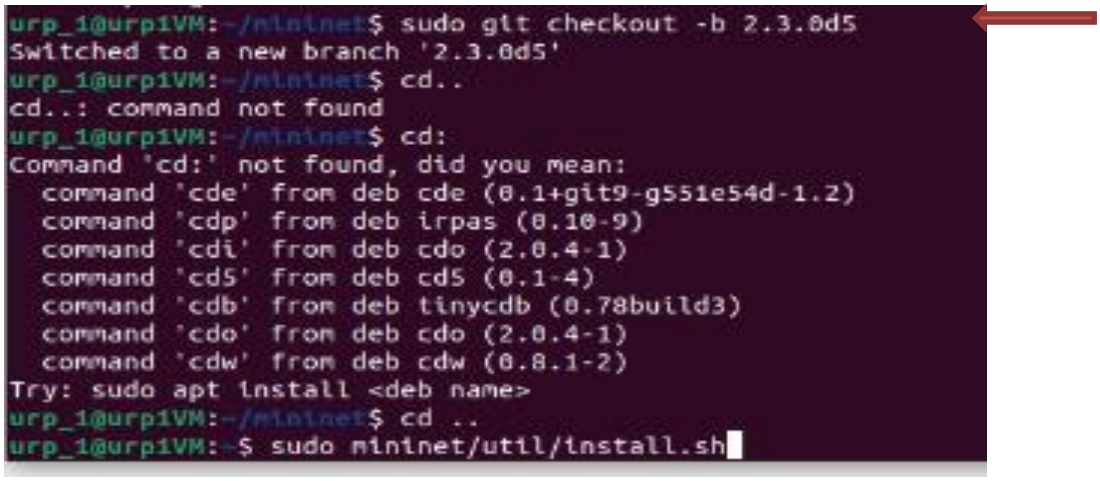


Nota. Elaboración propia.

Se instaló la versión 2.3.0d5, por lo cual se ejecutó el código *sudo git checkout -b 2.3.0d5.* (Figura 45)

Figura 45

Instalación de la versión 2.3.0d5

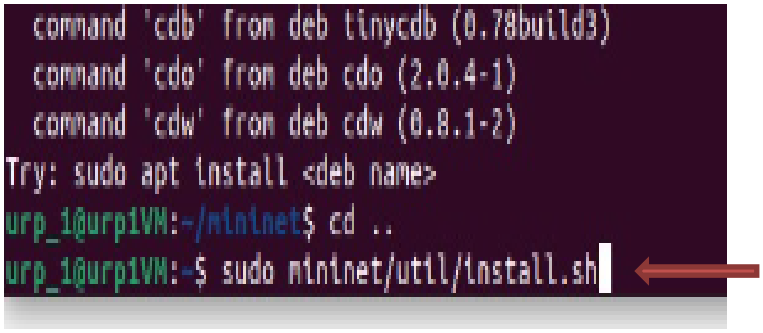


Nota. Elaboración propia.

Se volvió al directorio usando *cd* .. (Figura 46) y se instaló Mininet con el código *sudo mininet/util/install.sh* (Figura 47).

Figura 46

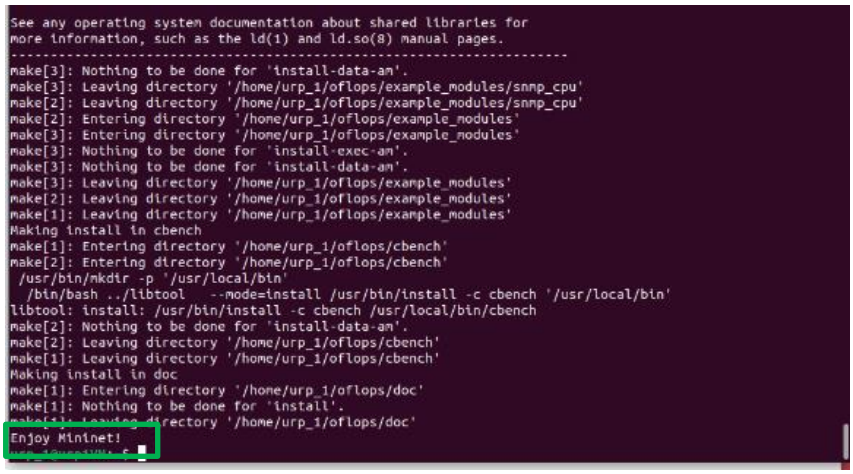
Regresando al directorio.



Nota. Elaboración propia.

Figura 47

Instalación de Mininet.



Nota. Elaboración propia.

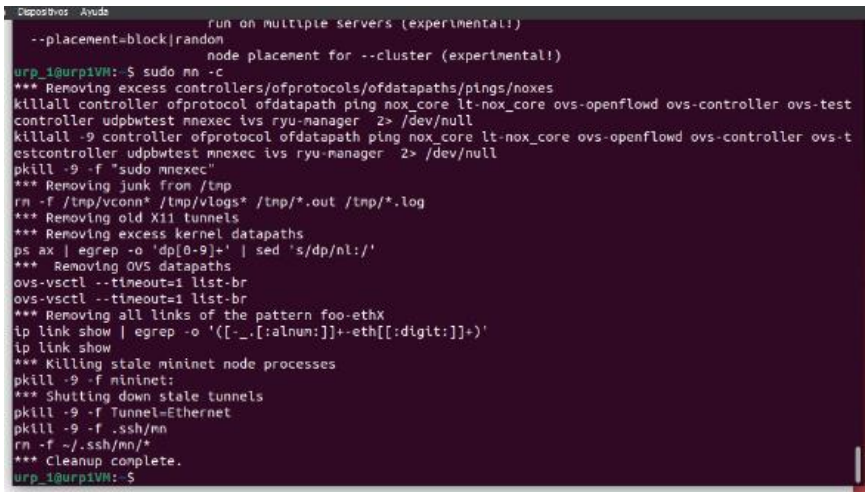
Como se observa en la figura 47, hay un aviso que dice “Enjoy Mininet”, el cual nos confirmó la instalación del emulador.

3.3.3. Sudo mn -c

El código *sudo mn -c* como se ve en la figura 48 es un comando que se usó en el sistema Linux para limpiar y reiniciar el entorno de emulación de red de Mininet. Esto se necesitó para ejecutar y poder simular nuestra topología de una manera limpia.

Figura 48

Comando sudo mn -c.



Nota. Elaboración propia

3.4. Miniedit

3.4.1. Instalación de Miniedit en Mininet

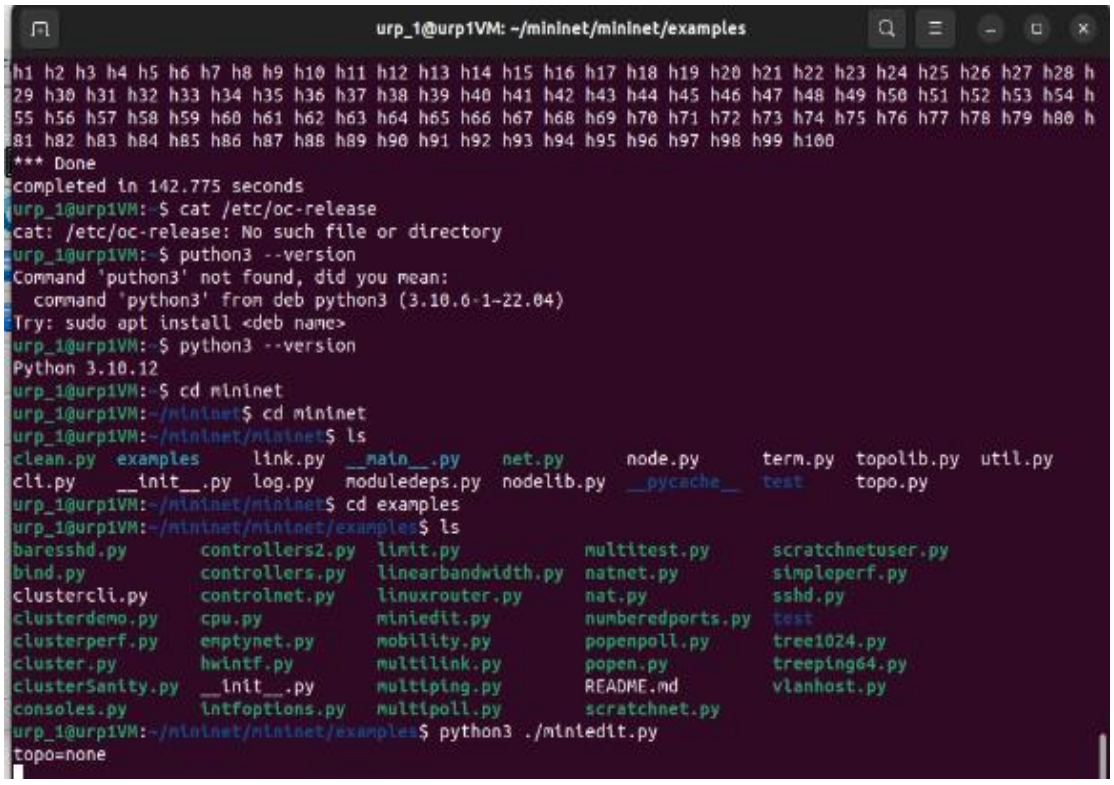
Se instaló MiniEdit que fue la interfaz gráfica donde se simuló la topología de red de la universidad Ricardo Palma con el controlador OpenDayLight SDN y los switch´s con el protocolo openFlow.

Se verificó la versión con el código *cat /etc/os-release* y la versión de Python con *python3 –version.* (Figura 49)

Se ingresó a la carpeta examples del escitorio de mininet usando *cd mininet, cd mininet, cd examples* (Figura 49) y se ejecutó MiniEdit *sudo python3 ./miniedit.py (Figura 50).*

Figura 49

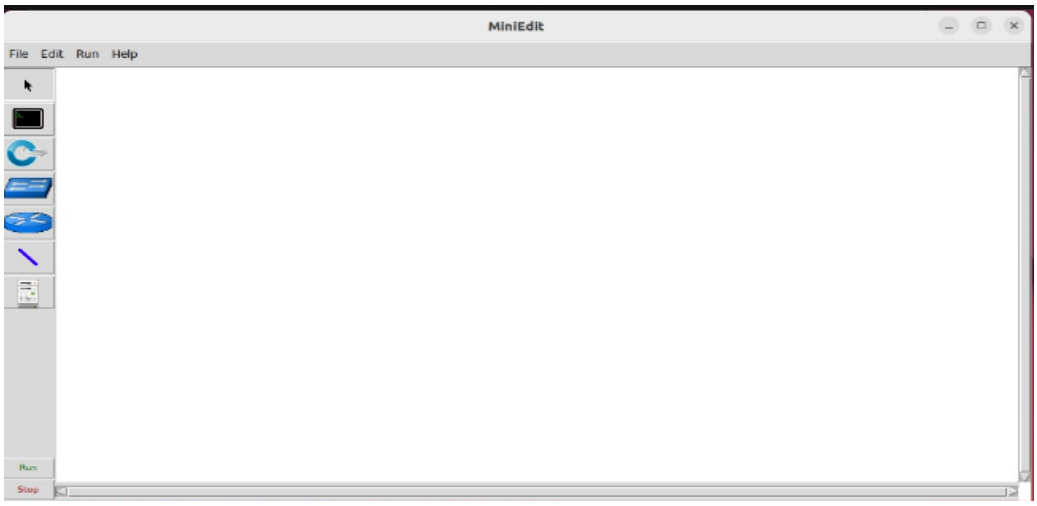
Verificación de versiones.



Nota. Elaboración propia.

Figura 50

MiniEdit ejecutado.



Nota. Elaboración propia.

Se vió que la interfaz de MiniEdit cuenta con unas pequeñas imágenes, las cuales son:

- Controlador SDN 
- Conmutador SDN 
- Host 
- Conexión en el plano de datos (azul) o plano de control (rojo) 

3.5. Instalación del controlador SDN (OpenDayLight)

3.5.1. Instalación de Java JRE versión 8

Se actualizó el sistema operativo usando *sudo apt-get -y update* y los paquetes usando *sudo apt-get -y upgrade*. Para que exista el sistema operativo de trabajo debe tener instalado las versiones de Java por lo cual se configuró la versión indicando el PATH respectivo, donde usamos los siguientes códigos en la consola:

cd /home/urp_1

sudo apt-get -y install openjdk-8-jre

sudo update-alternatives --config java

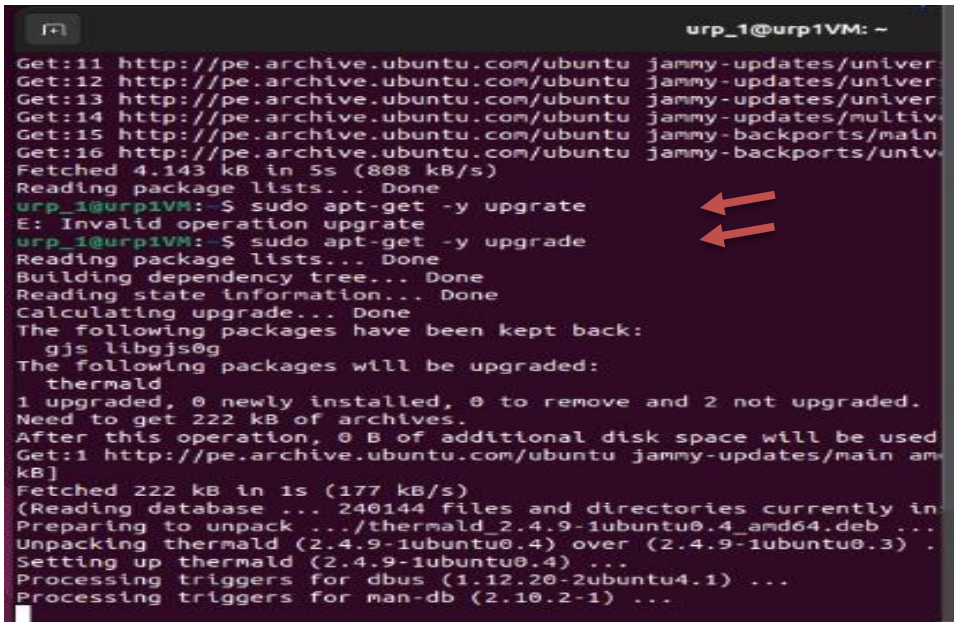
cd /home/urp_1

ls /usr/lib/jvm/java-8-openjdk-amd64/jre

Los ejecutamos uno después del otro como se observa en las figuras 51, 52 y 53.

Figura 51

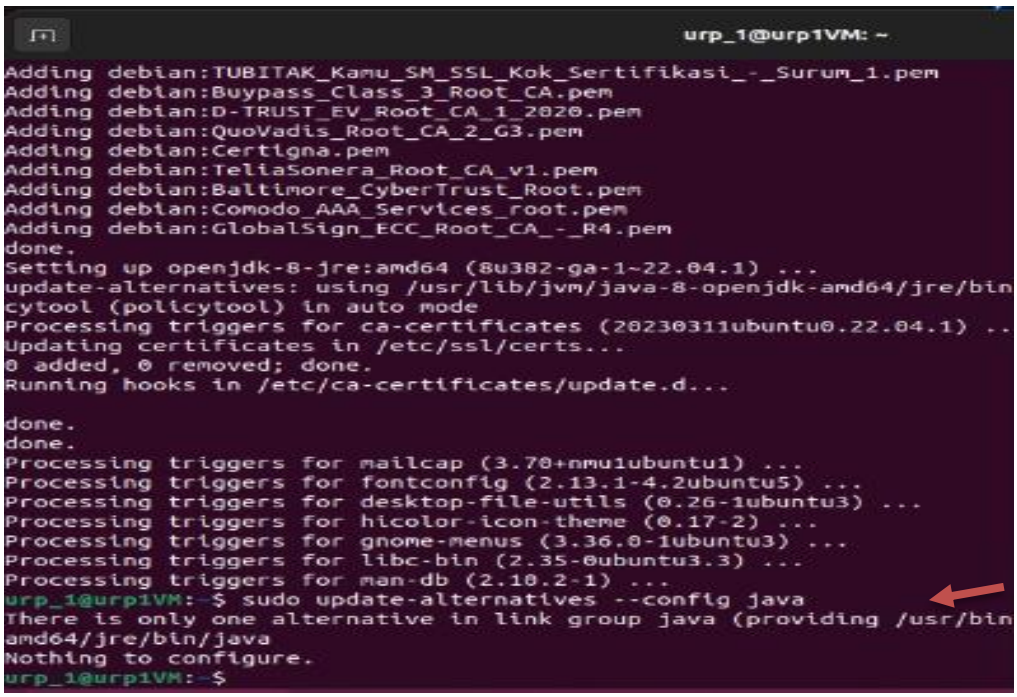
Ejecución de códigos.



Nota. Elaboración propia.

Figura 52

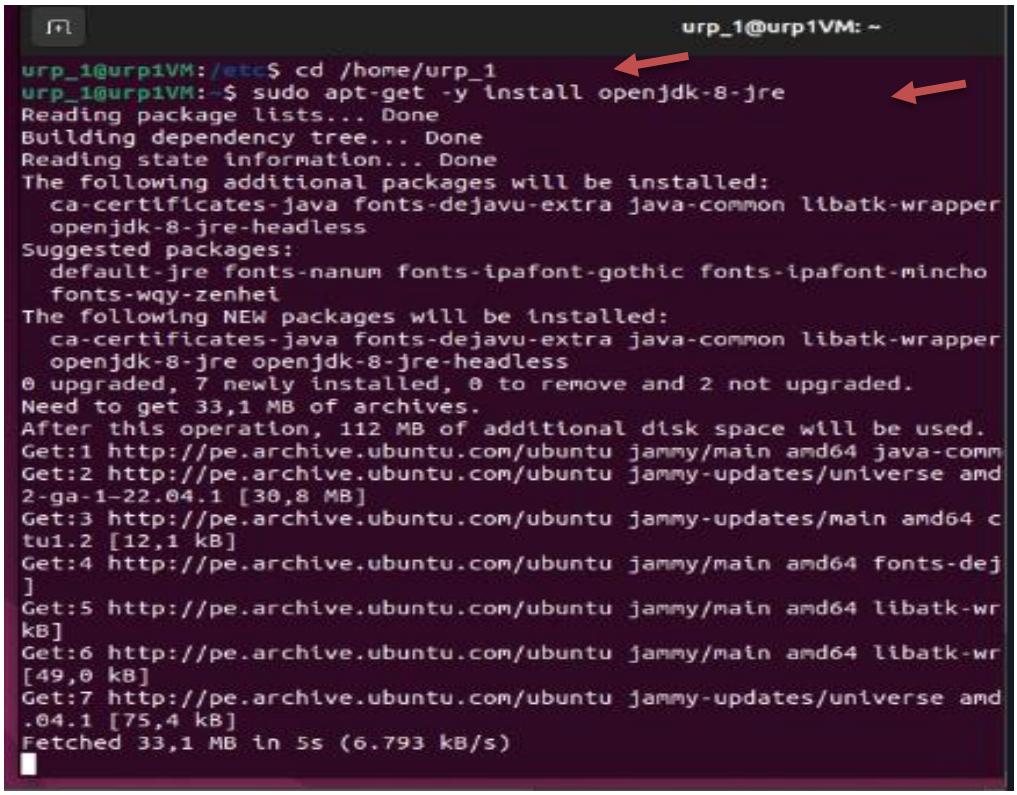
Ejecución de códigos.



Nota. Elaboración propia.

Figura 53

Ejecución de códigos.



Nota. Elaboración propia.

Se estableció la variable JAVA_HOME en Java versión 8 y 11 usando el siguiente código: *sudo nano /etc/environment* (Figura 54).

Figura 54

Código "sudo nano /etc/environment".

Nota. Elaboración propia.

sudo nano /etc/environment JAVA_HOME="/usr/lib/jvm/java-8- openjdk-amd64/jre"

source /etc/environment

echo $JAVA_HOME

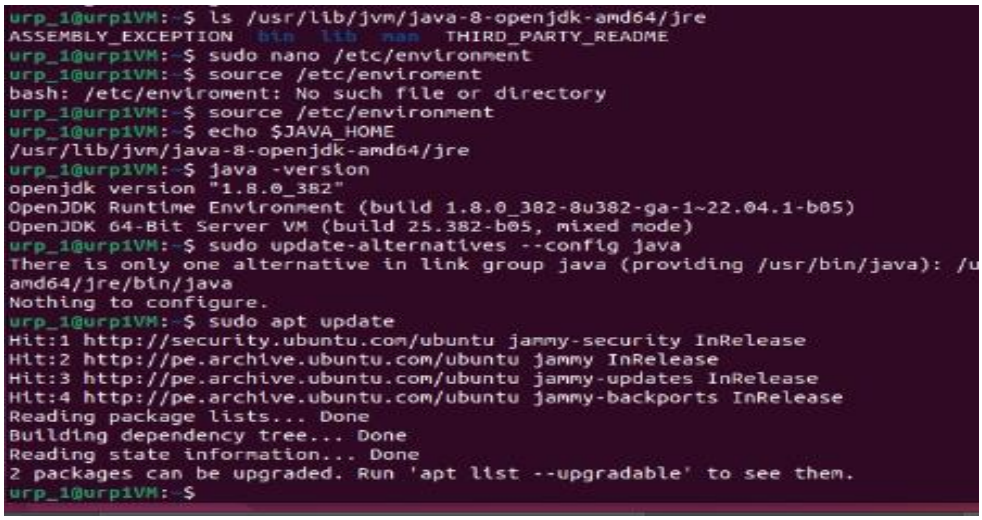
java -version

sudo update-alternatives --config java

sudo apt update

Figura 55

Continuación del código.

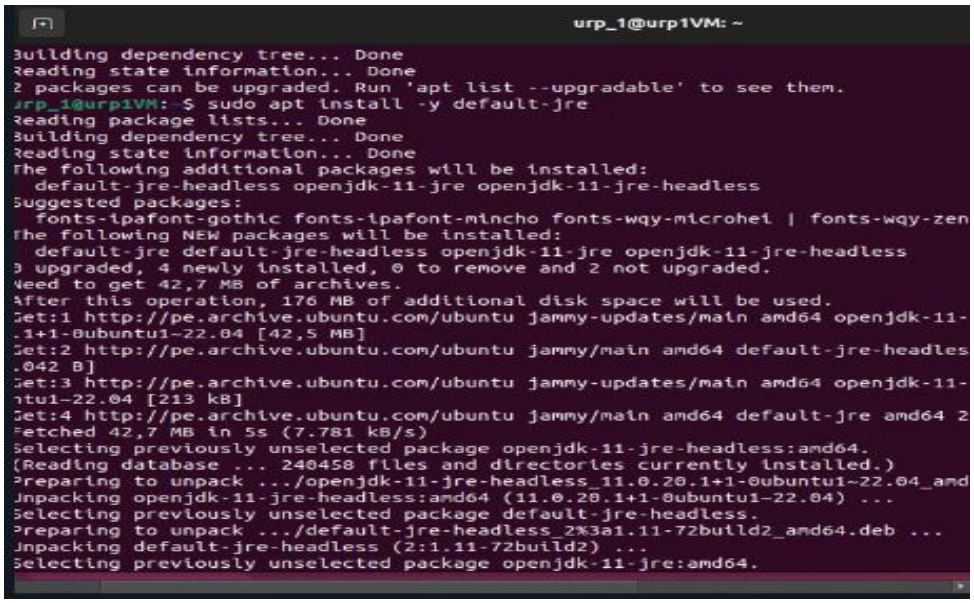


Nota. Elaboración propia.

Sudo apt install -y default-jre

Figura 56

Continuación del código.



Nota. Elaboración propia.

Sudo update-alternatives –config java

Figura 57

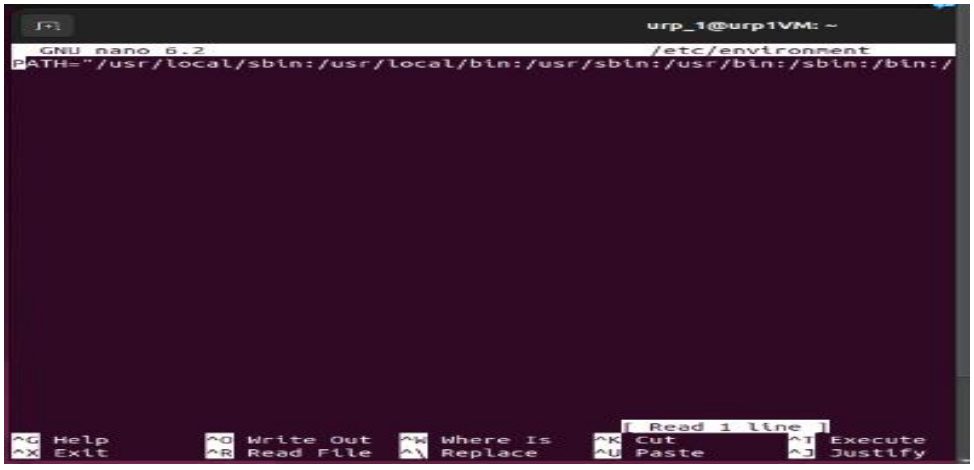
Continuación del código.

Nota. Elaboración propia.

Se aceptó la interfaz gráfica del Path (Figura 58).

Figura 58

Interfaz gráfica del Path.



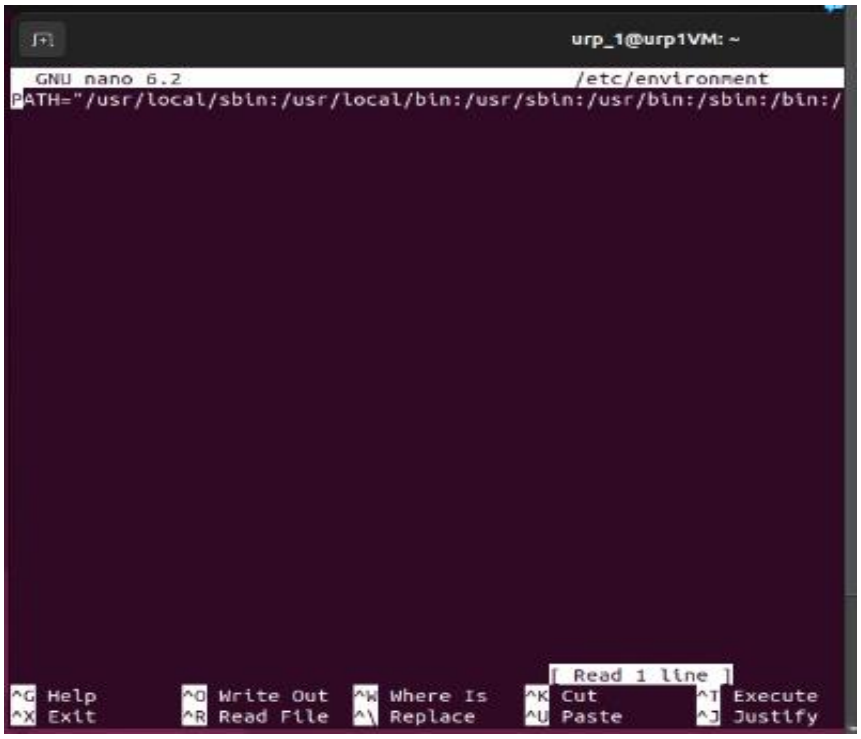
Nota. Elaboración propia.

3.5.2. Activación de JAVA para OpenDayLight con Karaf 0.8.4

El valor de la variable JAVA_HOME se restauró con el valor */usr/lib/jvm/java-8-openjdk-amd64/jre. P*ara realizar este cambio se ingresó el comando *sudo nano /etc/environment* y luego se modificó el valor de la variable JAVA_HOME. En este punto se tuvo java instalado en la versión 8 y 11, activado la versión 8 para trabajar con karaf-0.8.4 (Figura 59).

Figura 59

Visualización de instalación de karaf.



Nota. Elaboración propia.

3.5.3. Descarga del archivo tar.gz OpenDayLight

Se descargó el archivo TAR.GZ OPENDAYLIGHT que lo encontramos en la página web de OpenDayLight: https://opendaylight.org, como se observa en la figura 60.

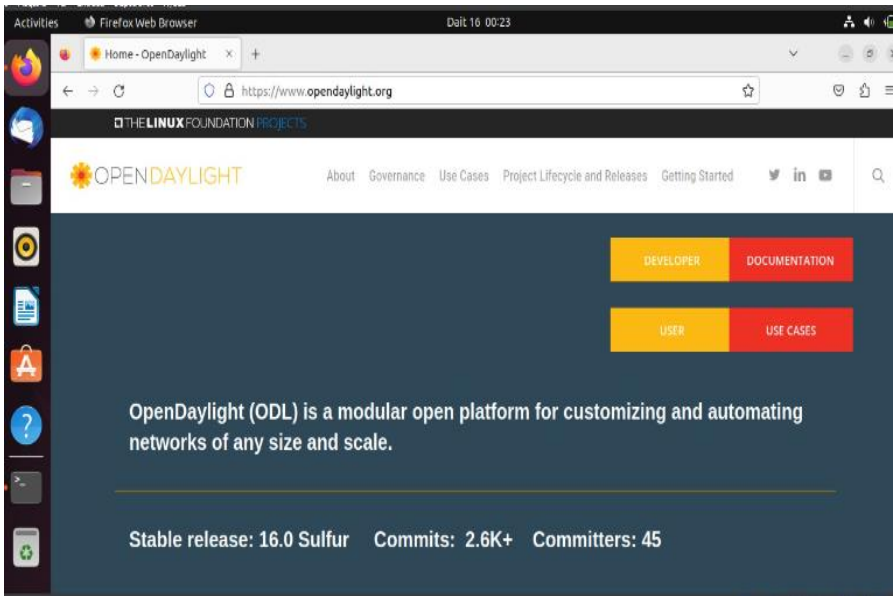
Se seleccionó la opción OpenDayLight Downloads, como se observa en la figura 61.

Luego se seleccionó la opción OpenDayLight (Nitrogen and Oxygen), como se observa en la figura 62.

Finalmente se seleccionó la opción donde se encuentra el archivo karaf-0.8.4.tar.gz y la descarga se inició como se observa en la figura 63.

Figura 60

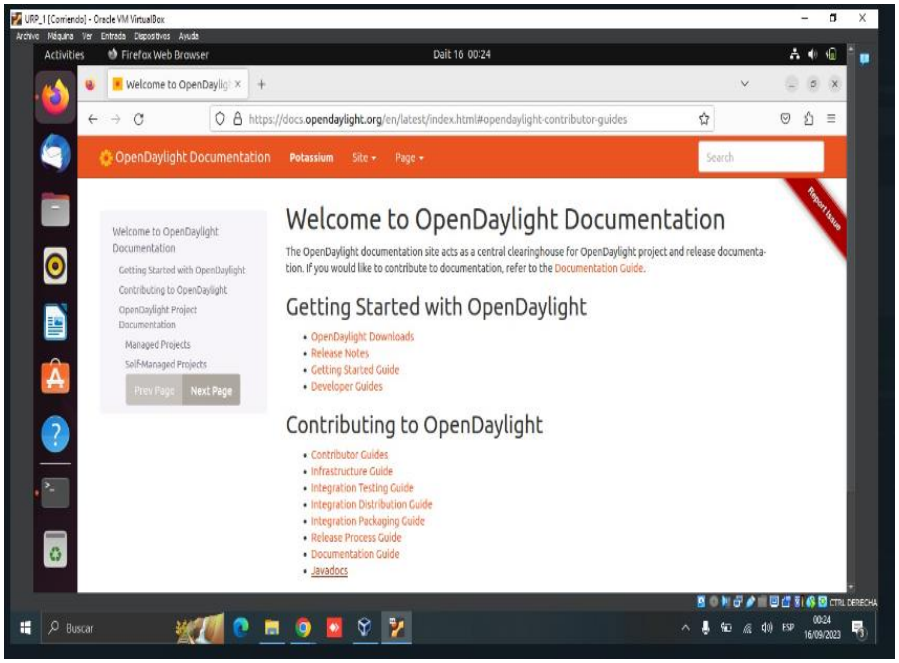
Página web de OpenDayLight.



Nota. https://opendaylight.org

Figura 61

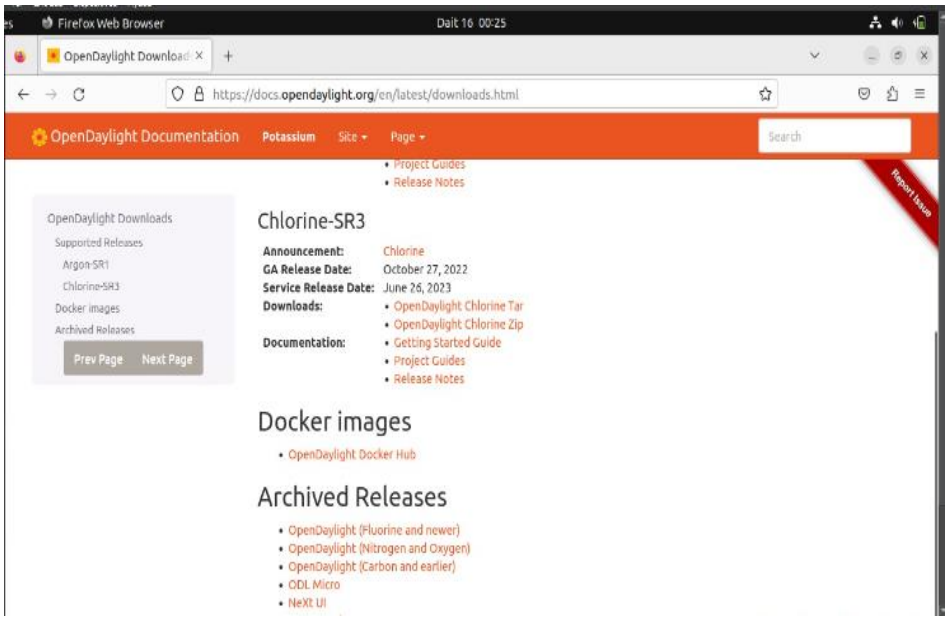
Selección de la opción OpenDayLight Downloads.



Nota. https://opendaylight.org

Figura 62

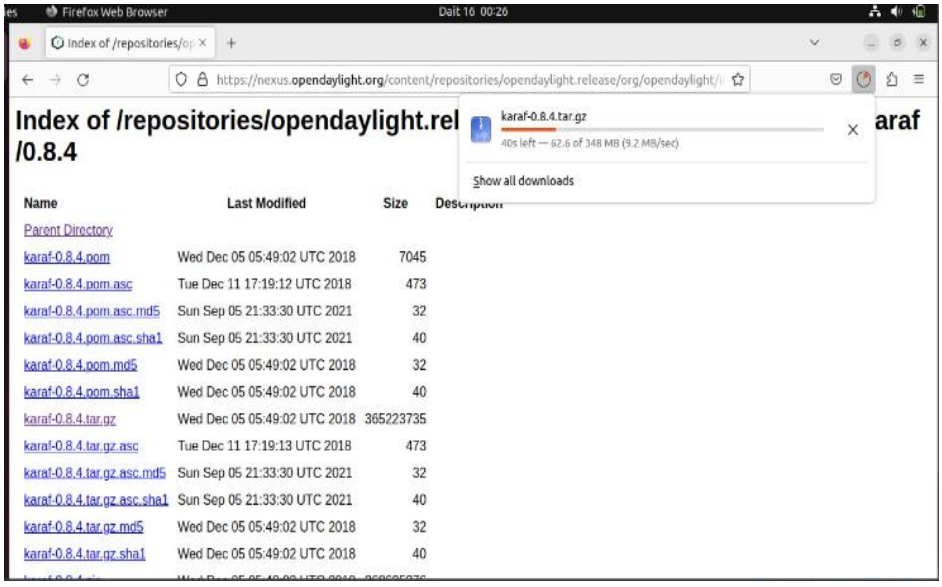
Selección de la opción OpenDayLight (Nitrogen and Oxygen).



Nota. https://opendaylight.org

Figura 63

Repositorio Opendaylight.



Nota. https://opendaylight.org

3.5.4. Instalación de OpenDayLight

Una vez obtenido el archivo Karaf-0.8.4.tar.gz, se instaló el controlador opendaylight mediante el siguiente código en la consola, los cuales se pueden observar en las figuras 65, 66, 67 y 68.

ls

mkdir opendaylight

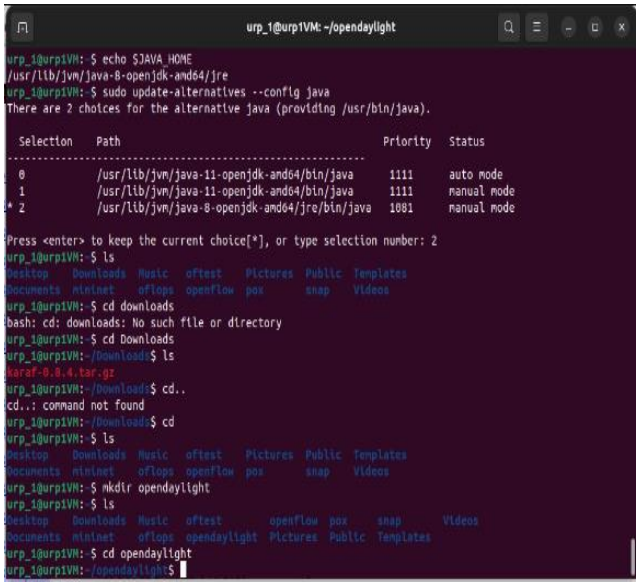
ls

cd opendaylight

cp /home/urp_1/Descargas/karaf-0.8.4.tar.gz karaf-0.8.4.tar.gz ls

Figura 64

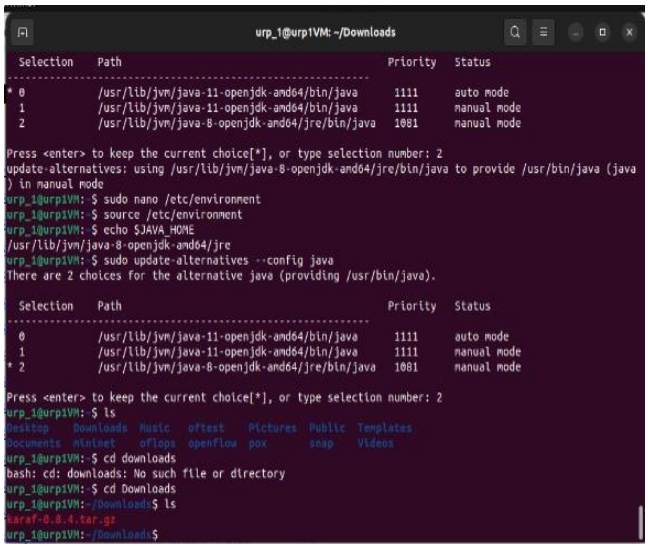
Código para instalar el controlador OpenDayLight.



Nota. Elaboración propia.

Figura 65

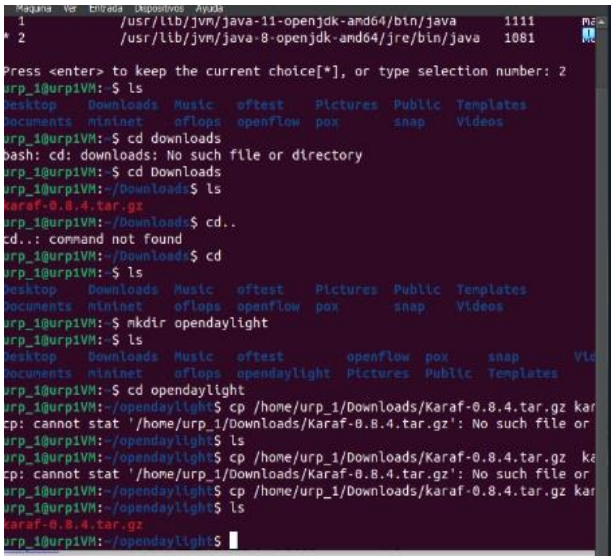
Código para instalar el controlador OpenDayLight.



Nota. Elaboración propia.

Figura 66

Código para instalar el controlador OpenDayLight.



Nota. Elaboración propia.

Luego se ejecutaron los códigos (Figura 67 y 68):

tar -zxvf karaf-0.8.4.tar.gz

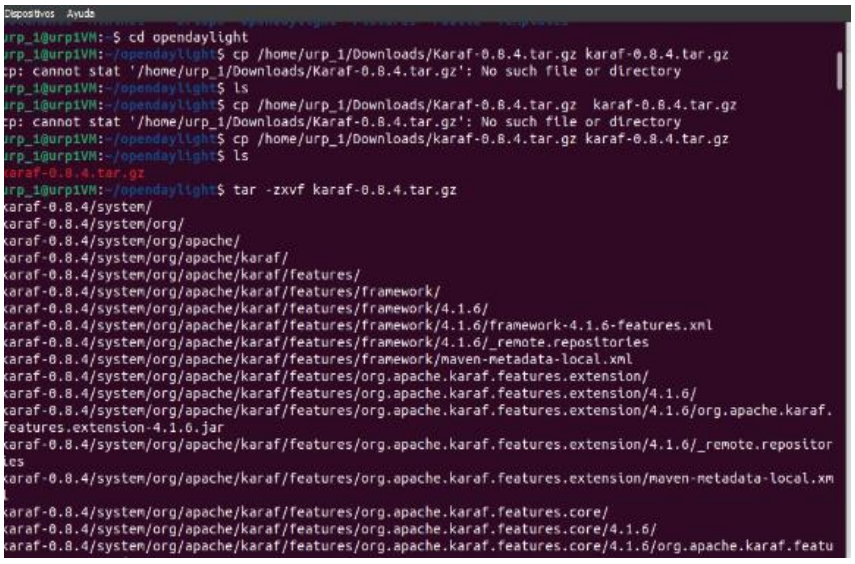
ls

cd karaf-0.8.4

ls

Figura 67

Código para instalar el controlador OpenDayLight.



Nota. Elaboración propia.

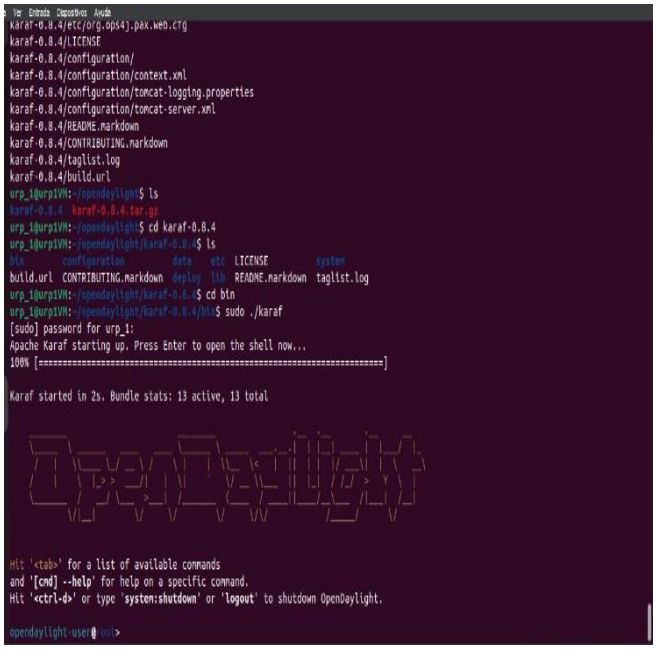
Se dio inicio al controlador Opendaylight mediante el código (Figura 68):

cd bin

sudo ./karaf

Figura 68

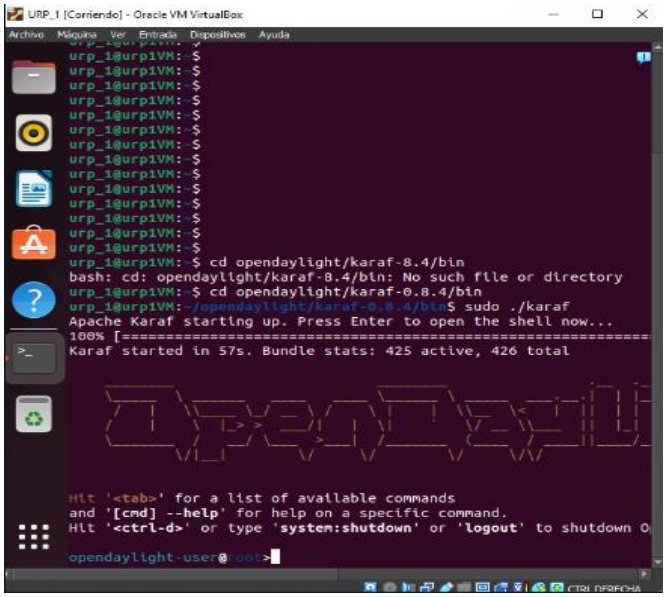
Código para iniciar el controlador OpenDayLight.



Nota. Elaboración propia.

Figura 69

Inicio de controlador OpenDayLight.



Nota. Elaboración propia.

3.5.5. Instalación de las características básicas de Karaf

Se instaló las características de Karaf que cumplirán las siguientes funciones:

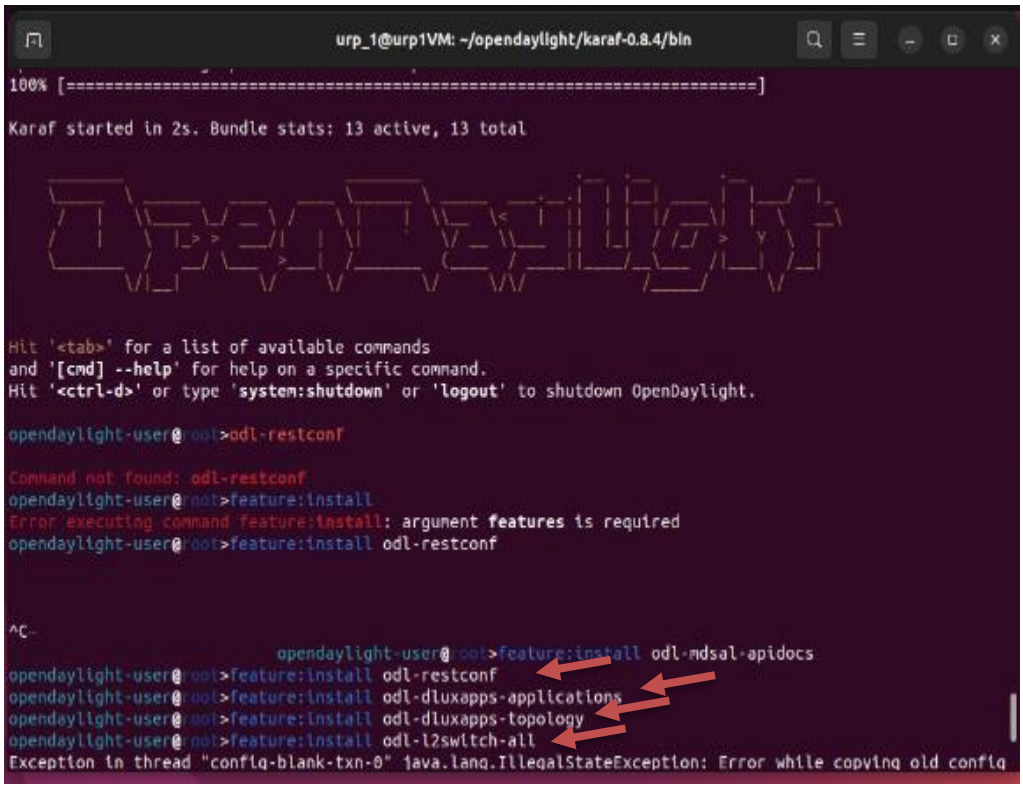
- odl-restconf: Facilitó la disponibilidad de la API REST para acceder a MD-SAL, lo que incluye la recuperación de datos almacenados.
- odl-l2switch-switch-ui: Ofreció la capacidad de dirigir el tráfico de capa 2 a través de conmutadores OpenFlow conectados, además brindó soporte para el seguimiento de dispositivos finales.
- odl-mdsal-apidocs: Dado que OpenDaylight (ODL) ha adoptado la arquitectura MD-SAL (Model Driven Service Abstraction Layer: Capa de Abstracción de Servicio Basada en Modelos), algunos ajustes estáticos, como la creación y gestión de flujos estáticos, se han reemplazado por un enfoque más abstracto y flexible.
- odl-dluxapps-applications: Proporcionó una interfaz de usuario gráfica intuitiva para OpenDaylight.
- odl-l2switch-switch: Garantizó que la topología muestre todos los detalles necesarios, por lo que fue necesario habilitar la función odl-l2switch-switch en Karaf.
- odl-dluxapps-topology: Es una aplicación básica de topología que mostró los nodos y enlaces de la topología de OpenFlow.

Para instalar estas características antes mencionadas, se usó el siguiente código (Figura 70 y Figura 71):

feature:install odl-restconf
feature:install odl-mdsal-apidocs
feature:install odl-dluxapps-applications
feature:install odl-dluxapps-topology
feature:install odl-dluxapps-nodes
feature:install odl-l2switch-all
feature:install odl-l2switch-switch-ui
feature:install odl-l2switch-switch
feature:install odl-l2switch-all

Figura 70

Instalación de características básicas de Karaf.



Nota. Elaboración propia.

Figura 71

Instalación de características básicas de Karaf.

Nota. Elaboración propia.

CAPÍTULO IV: PRESENTACIÓN Y ANÁLISIS DE RESULTADOS

4.1. Simulación de la red SDN con OpenDayLight en la URP

Como ya se configuró OpenDayLight con las funciones necesarias, se procedió a iniciar el controlador OpenDayLight, activándolo nuevamente. Para hacerlo, se ejecutó el siguiente comando: sudo ./karaf desde la ubicación en el directorio opendaylight/karaf-0.8.4/bin, como se muestra en la figura 72.

Figura 72

Controlador SDN OpenDayLight.

Nota. Elaboración propia.

4.1.1. Creación de la topología de red de la Universidad Ricardo Palma

La topología URP contó con 12 switchs que simulan 6 de los switch core que están en las facultades , 1 switch core que está en el laboratorio de ingeniería, 1 switch core en el edifico nuevo de Lenguas y psicología ,1 switch core que se encuentra en el edificio metálico, comúnmente llamado aulario, 1 switch core en la torre central de administración, 1 Switch core central que es el núcleo de conexión de toda universidad Ricardo Palma y 1 Switch OpenFlow que cuenta con el controlador OpenDaylight mientras los otros switch cuentan con una computadora que simula una consola para la verificación de conexión .

En la figura 73 se ve la distribución de los dispositivos.

Figura 73

Distribución de dispositivos.

Ubicación del Switch	SWITCH	HOST	CONTROLADOR SDN
Switch central	SWC1	h1	Protocolo OpenFlow
Faculta de Ingeniería	FI2	h2	Protocolo OpenFlow
Laboratorio de ingeniería	LI3	h3	Protocolo OpenFlow
Edificio metálico (Aulario)	EM4	h4	Protocolo OpenFlow
Nuevo Lenguas y Psicología	NLP5	h5	Protocolo OpenFlow
Torre central	TC6	h6	Protocolo OpenFlow
Facultad de medicina	FM7	h7	Protocolo OpenFlow
Facultad de biología	FB8	h8	Protocolo OpenFlow
Faculta de lenguas modernas	FLM9	h9	Protocolo OpenFlow
Facultad de ciencias económicas	FCE10	h10	Protocolo OpenFlow
Facultad de arquitectura	FA11	h11	Protocolo OpenFlow
SwitchOpenFlow	SWOF12	c0	Controlador OpenDaylight

Nota. Elaboración propia.

Se ingresó a la interfaz de MiniEdit al ejecutar el siguiente código en la consola de la máquina virtual como se ve en la figura 74.

cd

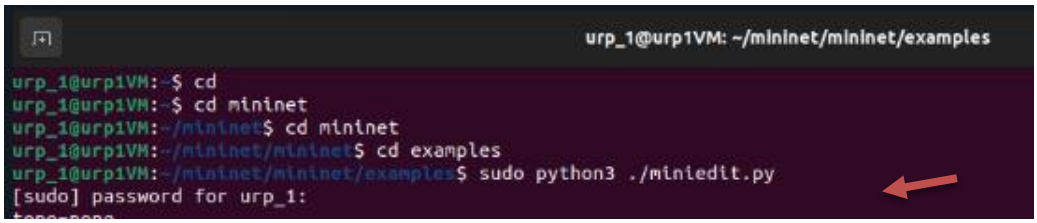
cd mininet

cd examples

sudo python3 ./miniedit.py

Figura 74

Ejecución de MiniEdit.

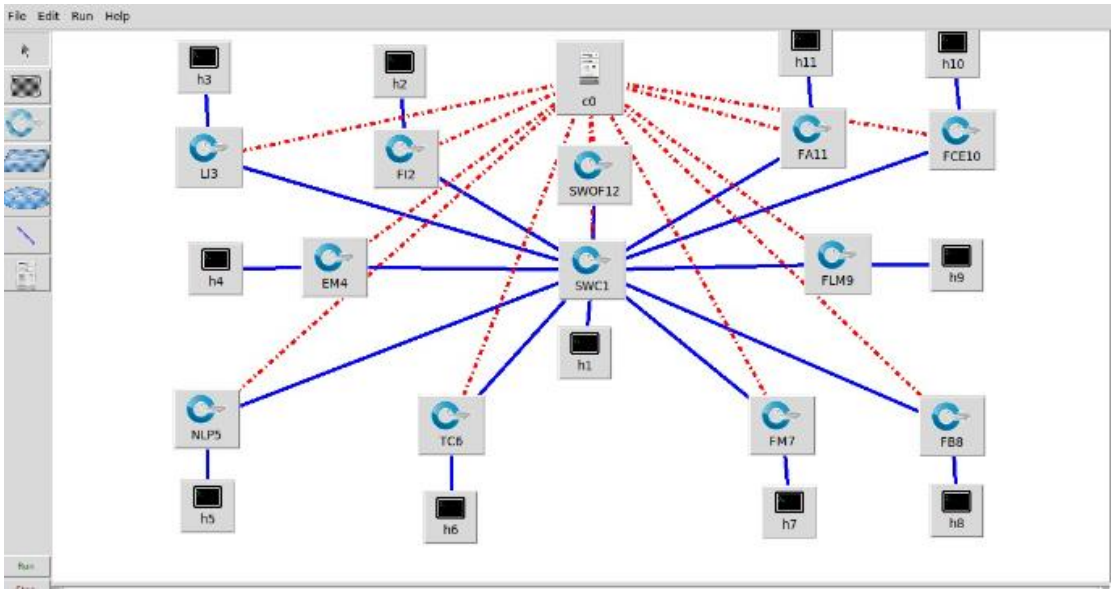


Nota. Elaboración propia.

Se usó MiniEdit y se configuró la topología de la red definida por software (SDN) en la Universidad Ricardo Palma estableciendo OpenDaylight como el controlador. Tal como se muestra en la figura 75 vemos que la arquitectura está conformada por 1 controlador c0 que es donde está hospedado el controlador OpenDaylight y cuenta con 12 switch´s Openflow, cada uno con su respectivo host, los cuales podemos ver el nombre de cada uno de los dispositivos en la figura 73, mencionada anteriormente.

Figura 75

Topología de la red de la Universidad Ricardo Palma.



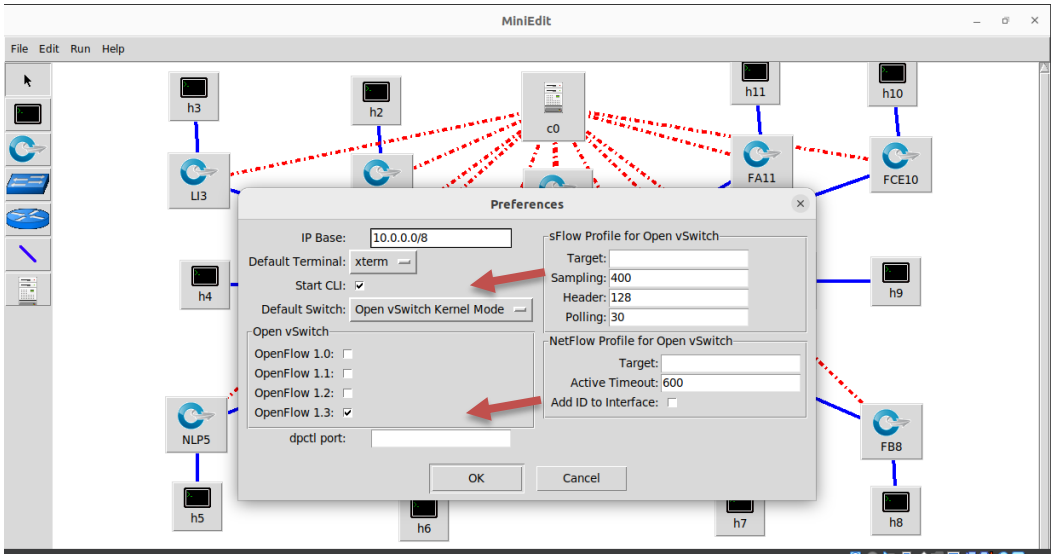
Nota. Elaboración propia.

4.1.2. Configuración del controlador OpenFlow

Se le dio click al botón Edit, luego click en preferences y nos apareció una interfaz, luego activamos Start CLI y verificamos que OpenFlow 1.3 está activado, todo esto como se observa en la figura 76.

Figura 76

Configuración del controlador OpenFlow.

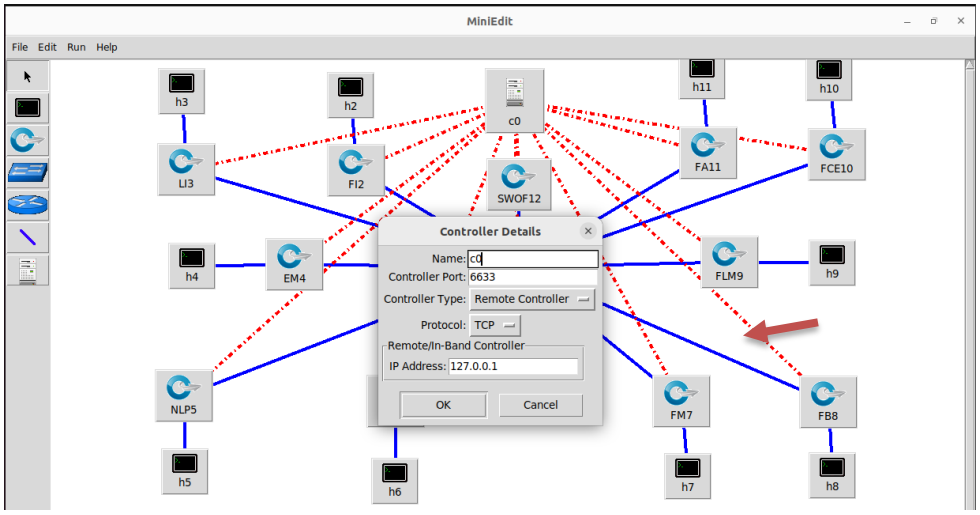


Nota. Elaboración propia.

Luego se le dio click derecho al Controlador Remoto y se puso la IP address de simulación 127.0.0.1 y se seleccionó Remote Controller en Controller Type (figura 77).

Figura 77

Configuracion de IP address.



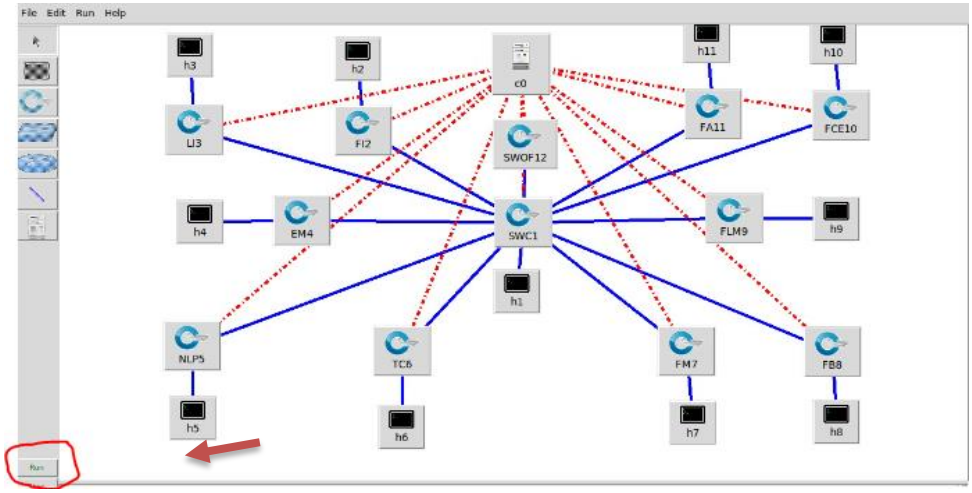
Nota. Elaboración propia.

4.1.3. Ejecución y verificación de conectividad de la red

Una vez activadas las opciones que se nombraron anteriormente, se le dio click en la opción “Run”, el cual se encuentra en la parte izquierda inferior (Figura 78).

Figura 78

Click en la opción Run.

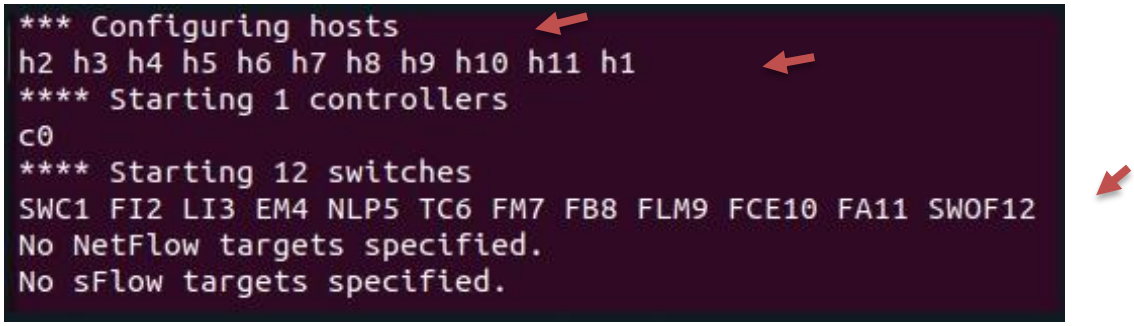


Nota. Elaboración propia.

Se verificó en la consola donde se nos muestra los 11 host, 12 switches y el controlador, han sido configurados de manera exitosa como se muestra en la figura 79.

Figura 79

Verificación de la configuración.



Nota. Elaboración propia.

Se ejecutó el comando pingall y vemos que todos los hosts tienen conexión entre sí, como se ve en la figura 80.

Figura 80

Comando pingall.

Nota. Elaboración propia.

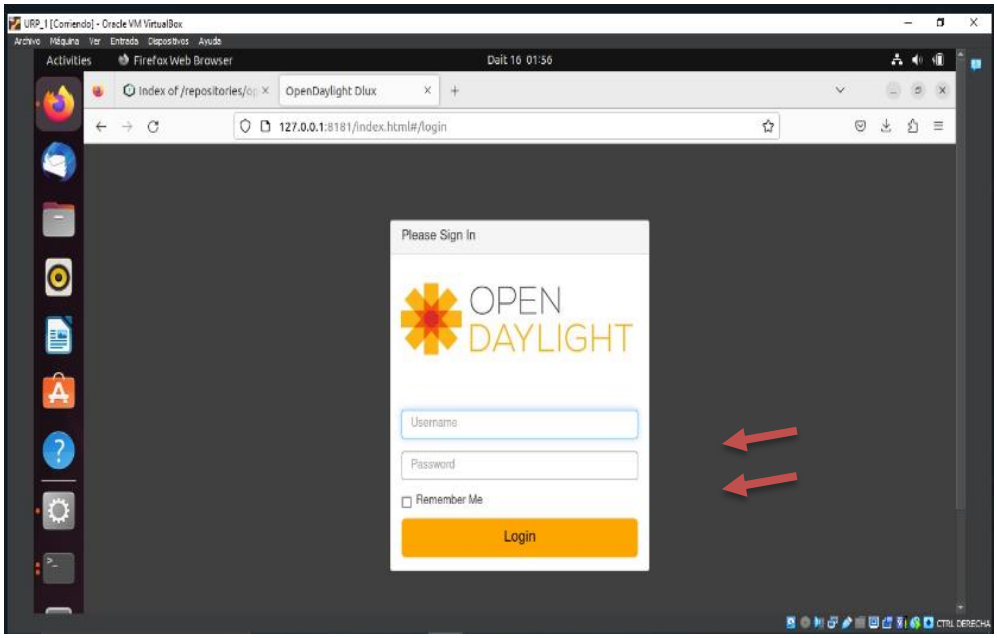
4.1.4. Uso de la plataforma de administración para el controlador OpenDaylight

El servidor OpenDayLight incluye una interfaz web que permitió la observación de la red SDN implementada. Para acceder a este servidor desde un cliente, se utiliza el navegador web que se encuentra en la máquina virtual que aloja el controlador OpenDayLight. Se accedió ingresando la siguiente dirección en la barra de direcciones del navegador: http://127.0.0.1:8181/index.html. En esta dirección, "127.0.0.1" representa la ubicación del controlador y "8181" es el puerto del servidor web.

Al ingresar al servidor web de OpenDayLight, se mostró una ventana similar a la que se presenta en la figura 79 para acceder, se debe utilizar "admin" como nombre de usuario (login) y contraseña (password).

Figura 81

Credenciales OpenDayLight.

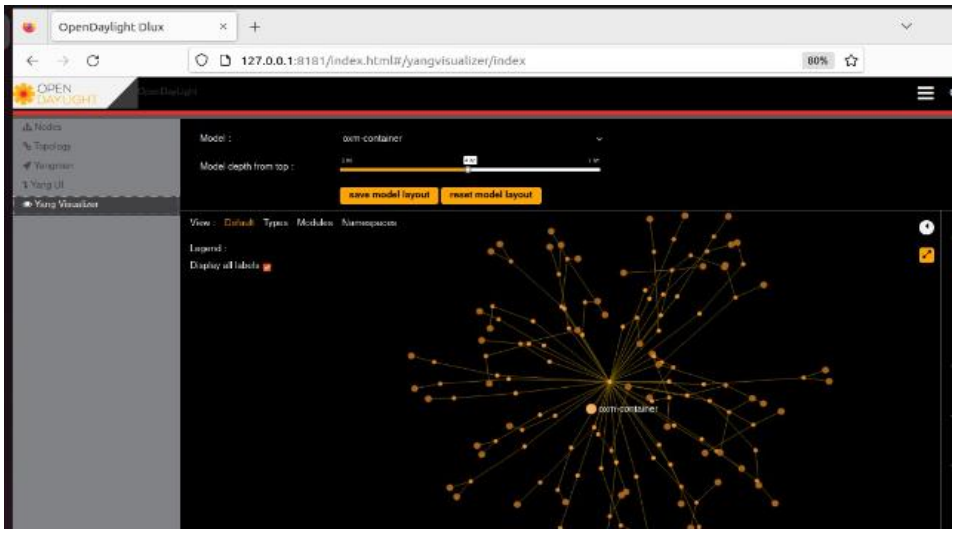


Nota. Elaboración propia.

Después de ingresar las credenciales, se vio la interfaz del controlador OpenDaylight, donde en la parte izquierda tenemos 4 opciones, las cuales son Nodes, Topology, Yangman, YangUI y Yang como se muestra en la figura 82.

Figura 82

Interfaz del controlador OpenDaylight.



Nota. Elaboración propia.

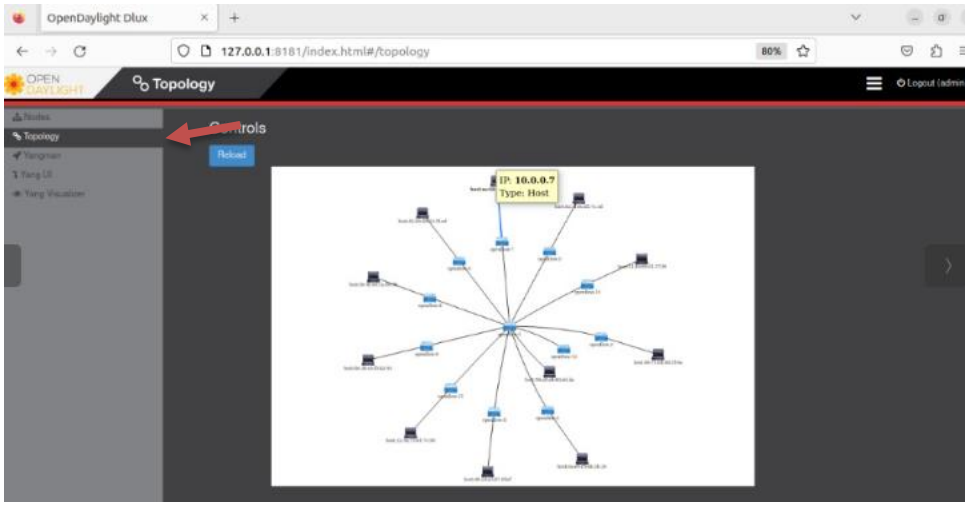
a) Automatización en la Topology

Una vez dentro de la consola de OpenDaylight, se le dio click a la opción "TOPOLOGY" que nos presentó la información acerca de la estructura de la red de la Universidad Ricardo Palma. Se vio una representación gráfica de la interconexión de los dispositivos de red, donde se pudo ver en detalle los enlaces y las relaciones que hay entre los nodos. Esta característica resultó fundamental para la gestión y supervisión de la infraestructura de red, ya que nos brindó una visión clara y rápida de la organización y funcionamiento de la red en un instante dado. Además, esta característica resulta de ayuda en la identificación de problemas de conectividad que es muy común en campus universitarios como es el caso de la Universidad Ricardo Palma y en la toma de decisiones relacionadas con la configuración de la red.

En la figura 83, podemos ver de forma automatizada la topología de la URP.

Figura 83

Control de la topología de la URP.



Nota. Elaboración propia.

b) Administración de nodos

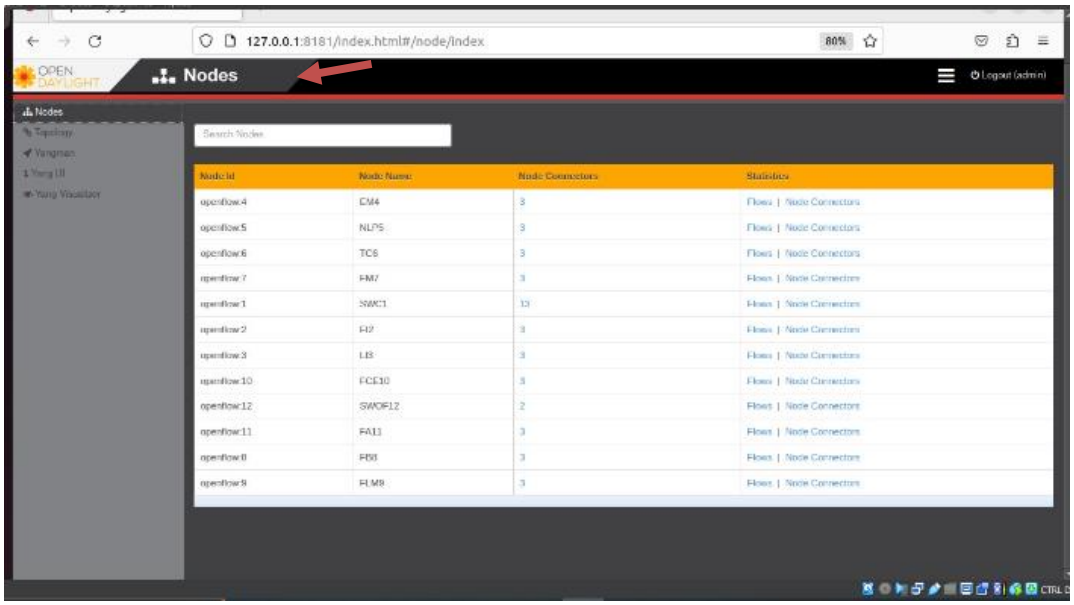
Dentro de la consola de OpenDaylight, se accedió a la opción "NODES" que cumple la función de presentar información acerca de los nodos de la red. Esta función despliega un listado que abarca todos los dispositivos o nodos de la red que han sido registrados y están siendo administrados por el controlador urp_1, controlador OpenDaylight. Dicha información incluye detalles exhaustivos acerca de cada nodo, tales como la dirección IP, su tipo (por ejemplo, switch, router, host), su identificador único y otros atributos pertinentes.

El propósito esencial de la función "NODES" radica en proporcionar a los administradores de red de la Universidad Ricardo Palma una perspectiva general de los dispositivos presentes en la red que están bajo el control y supervisión de OpenDaylight. Este recurso resulta invaluable para llevar a cabo tareas de supervisión, configuración y gestión de la red, ya que permite la identificación y gestión eficaz de los nodos, además de la posibilidad de rastrear su estado y comportamiento dentro de la infraestructura de red dirigida por OpenDaylight.

En la figura 84, se muestra la interfaz desde donde se controla los nodos de la URP.

Figura 84

Control de nodos de la universidad Ricardo Palma.

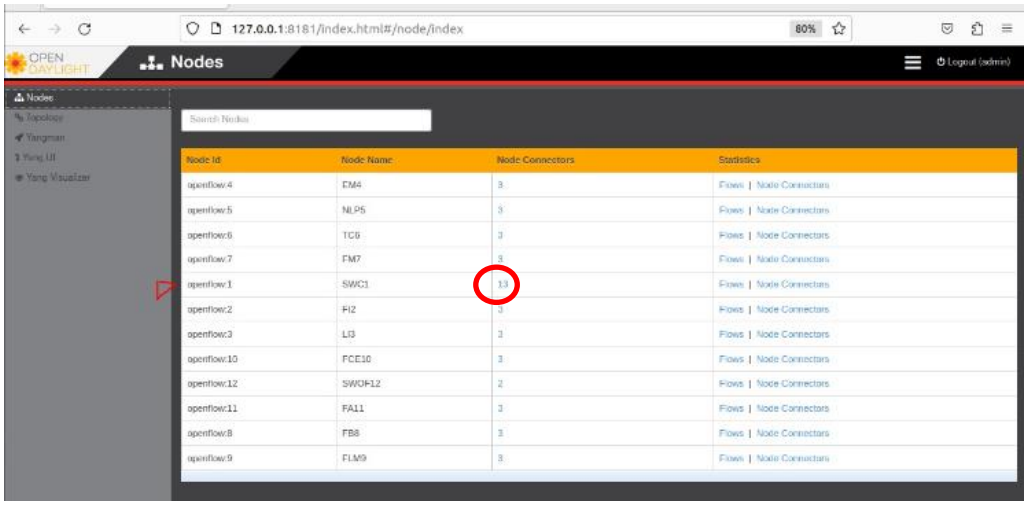


Nota. Elaboración propia.

Se le dio click al número 13 del SWC1, como se ve en la figura 85 y nos muestra las conexiones que tenemos en ese switch, como también la MAC address y el número de puerto de cada dispositivo conectado al switch (figura 86).

Figura 85

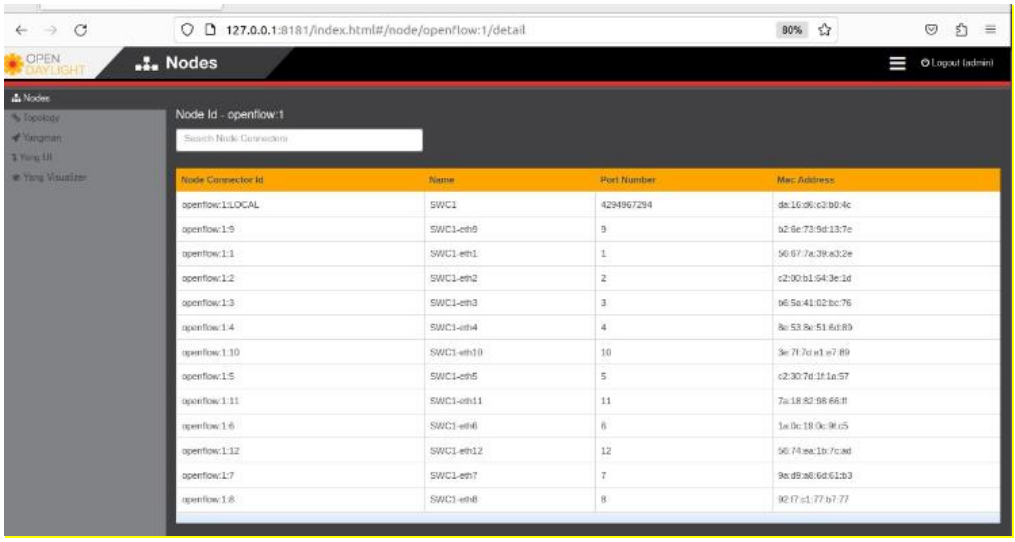
Selección del nodo colector SWC1.



Nota. Elaboración propia.

Figura 86

Nodos ID del SWC1.

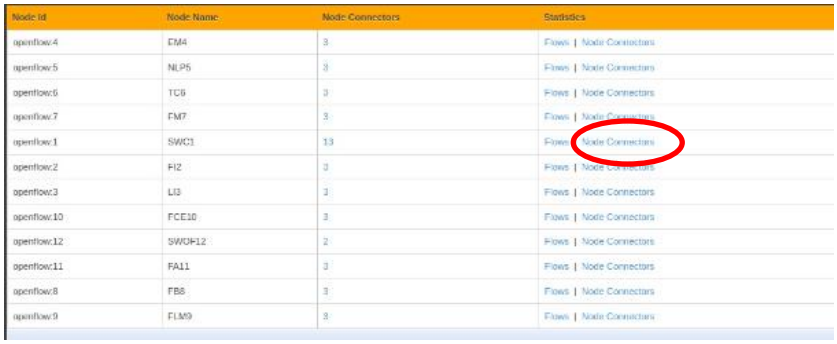


Nota. Elaboración propia.

Después se le dio click a “Node Conection” del mismo switch central (figura 87) y nos mostró los bytes, drops, picts , errores, frame error, over run error, colisiones de cada uno de los dispositivos conectados al switch central (figura 88).

Figura 87

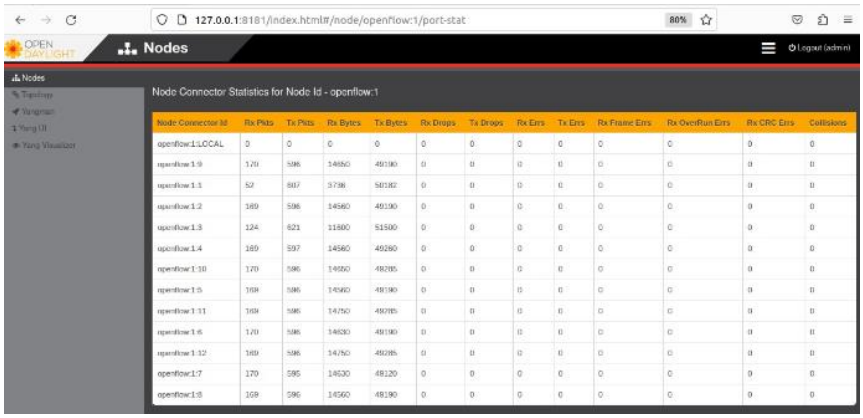
Selección de Node Connectores del SWC1.



Nota. Elaboración propia.

Figura 88

Control de características de los nodos conectador al SWC1.



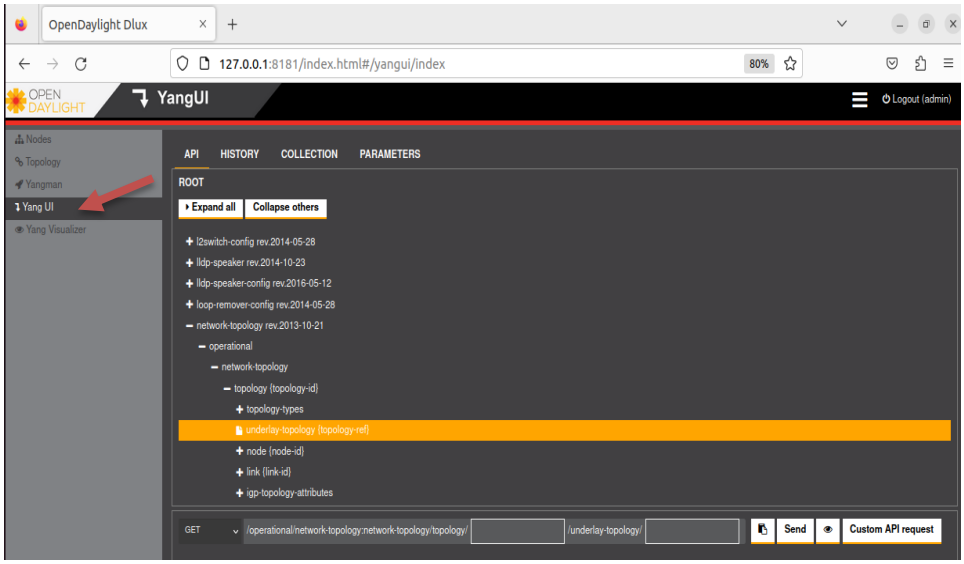
Nota. Elaboración propia.

c) Módulo de inventario

El módulo de inventario, denominado Yang UI (figura 89) o Yangman (figura 90), que forma parte de los dos nodos finales en la aplicación, ofrece la capacidad de examinar o ajustar el estado de cada interruptor en la topología. Desde este módulo, accedimos a información exhaustiva sobre cada nodo (figura 91) y también se realizó configuraciones que influyen en el funcionamiento de la topología (figura 92). A través de esta interfaz, los usuarios podrán llevar a cabo acciones como establecer parámetros de red en la universidad Ricardo Palma, monitorear el estado de los dispositivos y administrar servicios de red y poder verlos desde la interfaz Yaung Visualizer (figura 93).

Figura 89

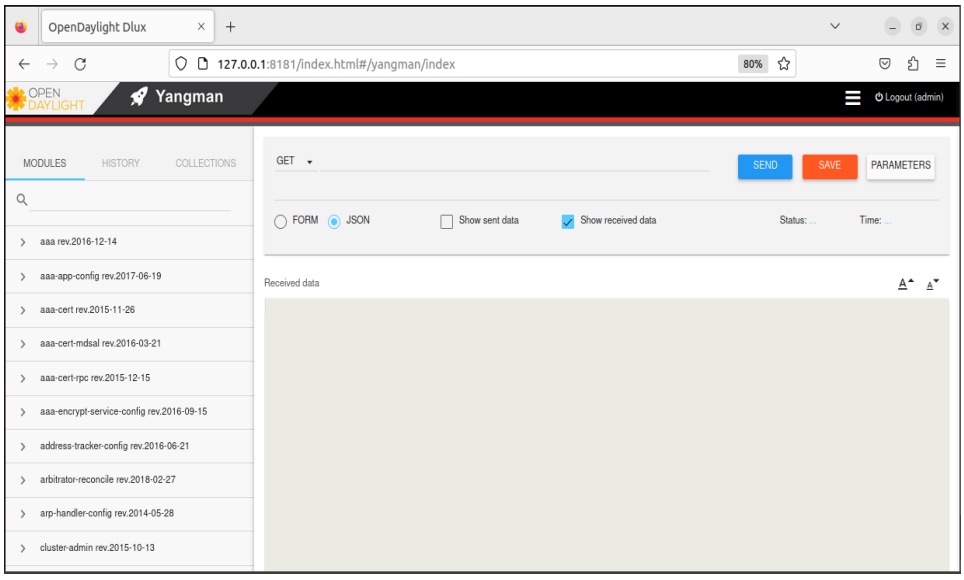
Interfaz YangUI.



Nota. Elaboración propia.

Figura 90

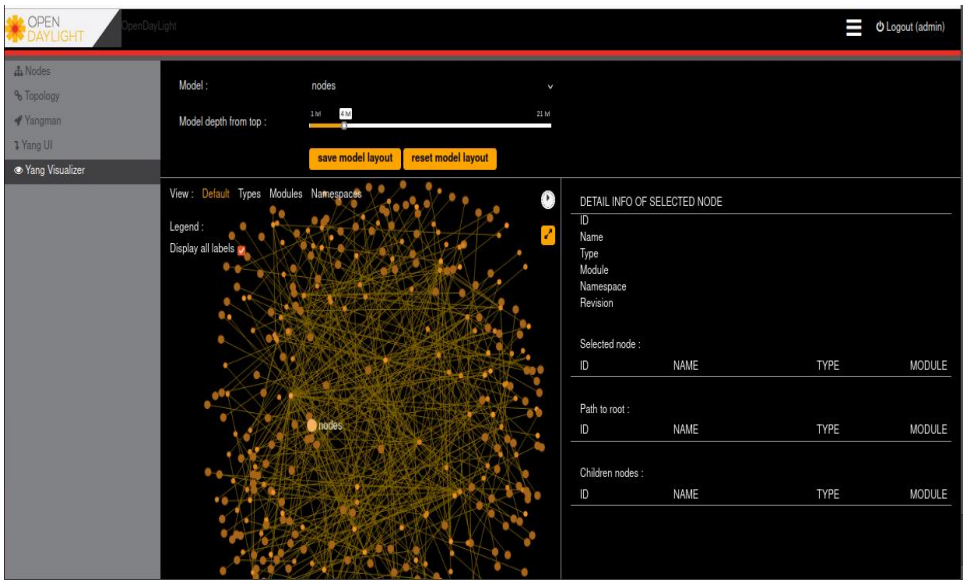
Interfaz de la opcion Yangman.



Nota. Elaboración propia.

Figura 91

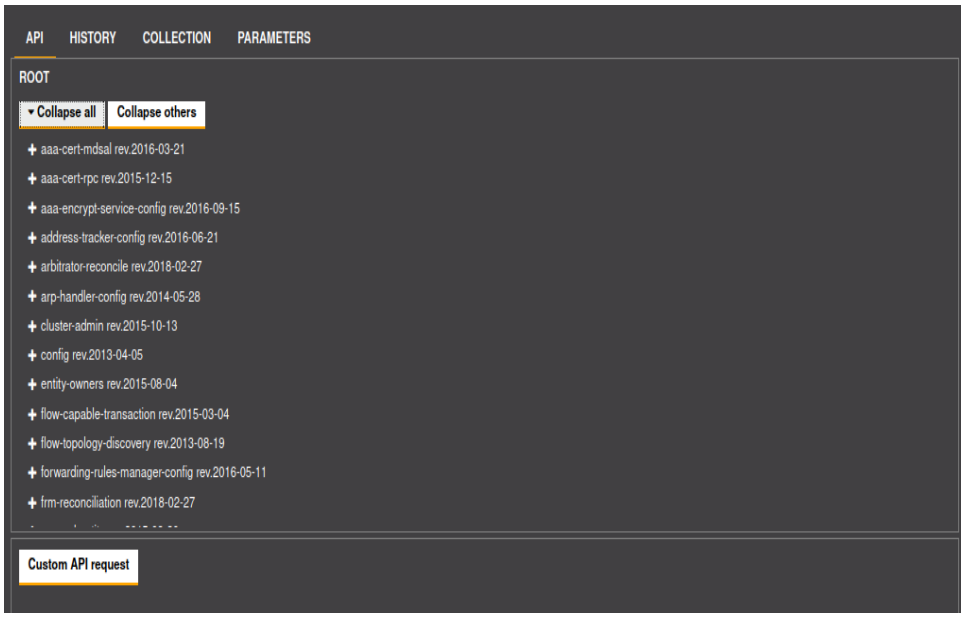
Visualización de nodos.



Nota. Elaboración propia.

Figura 92

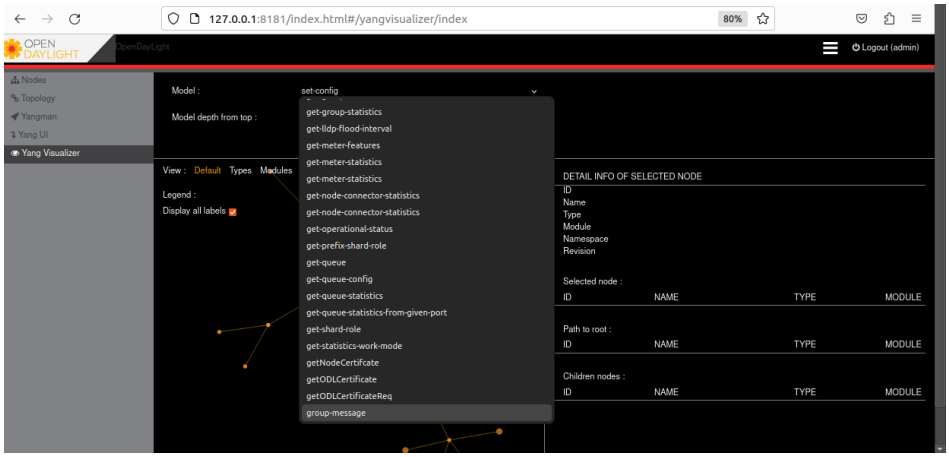
Configuraciones de la topología mediante API.



Nota. Elaboración propia.

Figura 93

Interfaz Yang Visualizer.



Nota. Elaboración propia.

4.2. Presupuesto de implementación

4.2.1. Precios de hardware y software

- Licenciamiento de software del controlador.

En lo que respecta al controlador OpenDaylight, se basta con acceder al repositorio de descargas en la página oficial de ODL y descargar el instalador en formato ova o .img de manera inmediata. En cuanto a los switches que formaron parte de la red de la Universidad Ricardo Palma, su licencia se encuentra incluida en el paquete de software; solo es necesario activarla en los switches propuestos mediante una actualización, siempre y cuando seamos socios o clientes de Cisco.

En cuanto al servidor, es necesario contar con VirtualBox o VMware, y es posible montar directamente el paquete de software de ODL en el servidor propuesto. Los simuladores son gratuitos y no conllevan ningún costo.

- Elección del hardware servidor.

Como se pudo observar en la simulación y los resultados, se requiere un servidor para hospedar el controlador SDN, que es el OpenDaylight. El controlador virtual no exige una cantidad significativa de recursos de hardware, solo necesita ser instalado y asegurarse de que cuente con suficiente espacio de almacenamiento para los scripts y configuraciones necesarios para satisfacer las necesidades del campus universitario de la Universidad Ricardo Palma. Por esta razón, se optó por el servidor Cisco Blade UCS de

la serie B como se ve en la figura 94, el cual se caracteriza por ofrecer un buen rendimiento y una gran eficiencia operativa tanto para cargas de trabajo físicas como virtuales. Este servidor tiene la capacidad de integrarse con el monitoreo a nivel UCS y está diseñado principalmente para ser programable desde un entorno virtual, lo que es ideal para trabajar con el controlador OpenFlow, como se ve en las especificaciones técnicas en la figura 95.

Figura 94

Servidor Cisco Blade UCS, serie B.

Nota. https://www.cisco.com/c/en/us/products/collateral/servers-unified-computing/ucs-b-series-blade-servers/datasheet-c78-2368888.html

Figura 95

Especificaciones técnicas del servidor Cisco.

Item	Specifications
Processors	Up to 2 3rd Gen Intel Xeon Scalable processors (1 or 2)
Memory	32 DDR4 DIMM slots: 16, 32, 64, 128 at up to 3200 MHz
Intel Optane DC persistent memory	16 DIMM slots: 128, 256, and 512 GB at up to 3200 MHz
mLOM	mLOM slot for Cisco UCS VIC 1440
Mezzanine adapter (rear)	1 rear mezzanine adapter for: • Cisco UCS VIC 1480 mezzanine card • Cisco port expander mezzanine card
Mezzanine adapter (front)	1 front mezzanine adapter for: • Cisco FlexStorage 12-Gbps SAS RAID Controller • Cisco FlexStorage 12-Gbps SAS RAID Controller with 2-GB cache • Cisco FlexStorage M.2 Boot RAID Controller
Internal storage	2 hot-pluggable front-access 7 mm form factor drives: • **SSD:** Enterprise performance and value SSDs with up to 7.6 TB per drive • **NVMe:** Up to 7.7 TB per drive **Or 4** • **M.2:** Upto 960GB per drive **Note:** Drives require a RAID or pass-through controller in the front mezzanine adapter slot.
Management	Cisco Intersight software

Nota. https://bladesmadesimple.com/2021/04/cisco-announces-the-ucs-b200-m6-blade-server/

En la figura 96 podemos observar el precio del servidor Cisco UCS B200 M6 Blade Server.

Figura 96

Precio del servidor Cisco.

Nota. https://itprice.com/es/cisco/n20-b6625-1.html

- Elección de Switch OpenFlow.

El equipo seleccionado es el Cisco Catalyst 3850 series, que desempeña un papel fundamental en la creación y la incorporación de la solución de red definida por software (SDN) en la Universidad Ricardo Palma. Se trata de una unidad central, de distribución y acceso de 24 puertos que están equipados con la interfaz programable ASIC y la programación del APIC mediante una actualización de firmware.

En la figura 97 se puede apreciar el switch cisco seleccionado.

Figura 97

Switch Cisco Catalyst 3850 Series.

Nota. https://tienda.dnpcorp.pe/inicio/138-switch-cisco-ws-c3850-24p-s.html#:~:text=%242%2C966.10%20%2BIGV.%20Cisco%20Catalyst%20WS-C3850-24P-S%20es%20un%20conmutador,255%20VLAN%20y%20soporte%20est%C3%A1ndar%20IEEE802.3at%20POE%20%2B.

Cuenta con 24 puertos Ethernet de 10/100/1000 y se puede rackear sin usar mucho espacio ya que necesita 1 RU, como se especifica en la figura 98 y también cuenta con poder POE disponible de 435 W y una fuente de alimentación de 715W AC, como se puede observar en la figura 99.

Figura 98

Características técnicas del Switch Cisco Catalyst 3850 Series.

CÓDIGO DE PRODUCTO	WS-C3850-24P-S
TIPO DE CAJA	1 RU
CONJUNTO DE CARACTERÍSTICAS	Base IP
SELECCIÓN DEL MÓDULO DE ENLACE ASCENDENTE SFP DE RED	C3850-NM-4-1G C3850-NM-2-10G
PUERTOS	24 puertos Ethernet 10/100/1000
PODER POE DISPONIBLE	435W
NÚMERO MÁXIMO DE APILAMIENTO	9 9
ANCHO DE BANDA DE LA PILA	480 Gpbs
CAPACIDAD DE CONMUTACIÓN	92 Gpbs
REENVÍO DE RENDIMIENTO	68.4Mpps
RAM	4 GB
MEMORIA FLASH	2 GB
DIMENSIONES	4,45 cm x 44,5 cm x 44,5 cm
PESO DEL PAQUETE	17,49 kg

Nota. https://tienda.dnpcorp.pe/inicio/138-switch-cisco-ws-c3850-24p-s.html#:~:text=%242%2C966.10%20%2BIGV.%20Cisco%20Catalyst%20WS-C3850-24P-S%20es%20un%20conmutador,255%20VLAN%20y%20soporte%20est%C3%A1ndar%20IEEE802.3at%20POE%20%2B

Figura 99

Consumo de energía del switch.

Modelo	Total de puertos 10/100/1000 o SFP o SFP+	Fuente de alimentación de CA predeterminada	Alimentación PoE disponible	Presupuesto POE con PS secundario de 1100 W	StackWise-480	Potencia de pila
WS-C3850-24P	24 PoE+	715WAC	435W	1535W		

Nota. https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-3850-series-switches/datasheet_c78-720918.html

En la tabla 1 se observa el precio total que engloba el hardware y el software para este proyecto. Donde se escogen 11 switches, uno que es el Switch Core y los otros 10 que representan a cada facultad escogida.

Tabla 1

Precio de hardware y software para el proyecto.

ITEM	Producto	Cantidad	Precio unitario	Total
1	Switch Cisco WS-C3850-24P-S	11	$ 2,966.10	$ 32,627.10
2	Cisco UCS B200 M6 Blade Server	1	$ 2,954.00	$ 2,954.00
3	Licencias OpenDaylight	11	$ 0	$ 0
			TOTAL	**$ 35,581.10**

Nota. Elaboración propia.

4.2.2. Rentabilidad de una red definida por software en la Universidad Ricardo Palma.

- Debido a las limitaciones de política de la Universidad Ricardo Palma no se puede hacer una comparación de la nueva red definida por software (SDN) con la antigua red de la Universidad Ricardo Palma debido a que los datos del hardware perteneciente a la red son privados.
- Según nuestra experiencia como egresados de la facultad de ingeniería electrónica, se

estima que la solución SDN permitirá ahorrar en cuanto a infraestructura un 30% a comparación de la implementación de la antigua red, cuya red tiene un tiempo aproximado de 15 años implementado y no cuenta con muchas funciones y/o características nuevas y actualizadas con la que contaría la nueva red definida por software.

- Se ha propuesto un organigrama de planificación y ejecución del proyecto de la implementación de red SDN en la Universidad Ricardo Palma.

En la tabla 2 se muestra la propuesta de planificación de tiempo del proyecto.

Tabla 2

Organigrama de planificación y ejecución de proyecto.

ORGANIGRAMA	Mes 1	Mes 2	Mes 3	Mes 4	Mes 5
FASE DE PLANIFICACIÓN					
Procesamiento de información técnica					
Fase de diseño detallado de la red SDN					
Elaboración de planos de datos					
Creación de planos de red eléctrica					
Diseño de topología de red y asignación de direcciones IP					
Elaboración de diagrama de interconexión y disposición de equipos en racks					
Adquisición de activos (software y hardware)					
Implementación de licencias					
FASE DE EJECUCIÓN					
Etapa de programación y configuración del controlador OpenDaylight					
Integración de componentes de software					
Fase de pruebas de funcionamiento					
Validación del protocolo Open Flow					
Etapa de operación y puesta en marcha					
FASE DE CIERRE					
Capacitación de empleados					
Conclusión del proyecto					

Nota. Elaboración propia.

- También se ha propuesto un monto para cada rol que desempeñara el personal a cargo de ejecutar el proyecto.

En la tabla 3 se puede observar el rol, función y el pago que se le otorgará al personal que se propone para ejecutar el proyecto.

Tabla 3

Personal ejecutable del proyecto.

ROL	FUNCION	PAGO MENSUAL
Ingeniero director del proyecto	Esta persona tendrá la responsabilidad de liderar, diseñar y supervisar la ejecución del proyecto en sus etapas de planificación, implementación y conclusión. Su función principal será garantizar que todos los miembros del equipo desempeñen sus funciones de manera óptima.	S/ 5,000.00
Ingeniero líder	Ocupará la segunda posición en términos de responsabilidad en el proyecto y su función principal será garantizar la integración efectiva de toda la red con el controlador, asegurando que este último pueda gestionar la red en su totalidad.	S/ 3,000.00
Ingeniero preventa	El responsable de compras se encargará de garantizar la adquisición de los activos necesarios.	S/ 3,000.00
Ingeniero Residente	Tendrá la responsabilidad de garantizar que todas las tareas relacionadas con la instalación e implementación del sistema físico se lleven a cabo sin problemas, lo que implica la instalación de cableado estructurado, ajustes eléctricos, y la instalación de equipos de comunicaciones y servidores.	S/ 2,500.00
Ingeniero Networking	Se seleccionará un equipo de ingenieros expertos en cada una de las áreas vinculadas al proyecto, tales como networking, programación y administración de servidores.	S/ 2,000.00
Técnicos de telecomunicaciones	Tendrán la responsabilidad de llevar a cabo la instalación e implementación del sistema físico, lo cual engloba la instalación de cableado estructurado, modificaciones eléctricas, así como la instalación de equipos de comunicaciones y servidores.	S/ 1,500.00
	TOTAL (S/) **TOTAL ($)** **$1 = 3.77 soles**	**S/ 17,000.00** **$ 4,505.00**

Nota. Elaboración propia.

- Teniendo en cuenta la tabla 1 del presupuesto de hardware y software, la tabla 2 de organización de proyecto y la tabla 3 de roles del equipo propuesto para el proyecto, podemos obtener un presupuesto (aproximado) general del proyecto como se ve en la tabla 4.

Tabla 4

Presupuesto general del proyecto.

ITEM		Cantidad	Monto Inicial	TOTAL
1	Equipo de ingeniería	6 meses	**$ 4,505.00**	**$ 22,525.00**
2	Hardware y software	-	**$ 35,581.10**	**$ 35,581.10**
				$ 58,106.10

Nota. Elaboración propia.

- Por último, se puede decir que la nueva red SDN (que se realizó en esta tesis) depende de la red actual de la Universidad Ricardo Palma, ya que tiene la misma distribución en cuanto a arquitectura, pero no los mismos switches, ya que estos se cambiaron para tener una mejoría en la escalabilidad, en los costos y en la automatización que reduce el tiempo de respuesta y acción ante cualquier incidente.

CONCLUSIONES

1. Se diseñó la arquitectura de la topología de una red definida por software (SDN) en la Universidad Ricardo Palma usando el software Mininet y Miniedit.

2. Se simuló la red definida por software (SDN) en la Universidad Ricardo palma usando el protocolo Openflow llamado OpenDaylight.

3. Se concluye que las redes definidas por software (SDN) posibilitan la separación y centralización del plano de control de la infraestructura subyacente en una red de la universidad Ricardo Palma. Esto brinda soluciones para una gestión eficaz y automatización de redes, especialmente en entornos como los centros de datos.

4. La implementación de SDN ofrece beneficios que incluyen la capacidad de programar la red de manera personalizada, aumentar la seguridad, agilizar las operaciones y habilitar la automatización a gran escala en toda la red.

5. Concluimos que la principal distinción entre la red SDN y la red convencional del campus de la Universidad Ricardo Palma, radica en su infraestructura: la red SDN se basa en software, mientras que la red actual URP se basa en hardware. Debido a que el plano de control opera con software, la red SDN es mucho más flexible en comparación con la red actual URP. Esto permite a los administradores gestionar la red, ajustar la configuración, asignar recursos y expandir la capacidad de la red, todo desde una interfaz centralizada sin necesidad de agregar más componentes físicos como se vio en la simulación en el capítulo 3.

6. Existen diferencias en cuanto a la seguridad entre la SDN y las redes URP. La SDN ofrece una mejor seguridad en muchos aspectos gracias a una mayor visibilidad y la capacidad de establecer rutas seguras. Sin embargo, dado que las redes definidas por software dependen de un controlador centralizado, proteger este punto es fundamental para mantener la seguridad de la red, ya que representa un posible punto único de falla que podría convertirse en una vulnerabilidad en las SDN por lo cual esta tecnología puede ser integrada con diferentes soluciones mediante agente como por ejemplo los EDR que dan seguridad al punto final mediante software.

RECOMENDACIONES

1. Tener una portátil o computador, con requerimientos de memoria RAM mínima de 8GB y 500GB de almacenamiento, así podrá soportar el controlador OpenDaylight.

2. Mantener una capacitación constante sobre redes definidas por software (SDN) para el/los encargados de supervisar la red de la Universidad Ricardo Palma.

3. Capacitar al encargado de la gestión de la red de la Universidad Ricardo Palma, sobre el controlador OpenDaylight, como también el uso de su plataforma, debido a que tiene muchos componentes y aplicaciones para poder lograr una buena automatización de la red definida por software.

4. Tener una capacitación sobre seguridad informática, cloud y ciberseguridad, así tener una infraestructura segura.

5. Tener una solución de ciberseguridad llamado EDR, el cual nos protege los endpoints de la infraestructura de la Universidad Ricardo Palma.

REFERENCIAS BIBLIOGRÁFICAS

Cáceres, J. & Casilimas, C. (s.f.). Arquitectura y funcionamiento de redes definidas por software (SDN). Disponible: https://repository.udistrital.edu.co/bitstream/handle/11349/29727/CasilimasFajardoCarlosAlexis2022.pdf?sequence=2&isAllowed=y

Conectrónica, "SDN: El futuro de las redes inteligentes", 2014. Disponible: https://www.ramonmillan.com/documentos/sdnredesinteligentes.pdf

Cuba, G. y Becerra, J. (2015). *Diseño e implementación de un controlador SDN/OpenFlow para una red de campus académica.* [Tesis de licenciatura, Pontificia Universidad Católica del Perú]. Disponible: https://tesis.pucp.edu.pe/repositorio/handle/20.500.12404/7149.

Esteban, N. (2018). Tipos de investigación. Disponible: https://core.ac.uk/display/250080756?utm_source=pdf&utm_medium=banner&utm_campaign=pdf-decoration-v1

González, A. (2004). Investigación básica y aplicada en el campo de las ciencias económico administrativas. En Revista Ciencia Administrativa, Universidad Veracruzana. Núm 1. 39-50

Guanoluisa E.D. (2019). *Diseño de la arquitectura de una red SDN mediante el protocolo Openflow con simulación en el software mininet para la infraestructura de una PYMES.* [Tesis de licenciatura, Universidad de las Américas]. Disponible: https://dspace.udla.edu.ec/bitstream/33000/10884/1/UDLA-EC-TIRT-2019-05.pdf

Piscoya, l. (1987). Investigación científica y educacional. Lima: Amaru editores

Rentería, B. L. y Paucar, A. A. (2022). *Análisis e implementación de un prototipo de red SDN en el campus norte de la UNACH.* [Tesis de licenciatura, Universidad Nacional de Chimborazo]. Disponible:

http://dspace.unach.edu.ec/bitstream/51000/9210/3/Informe%20Final%20del%20Investigaci%c3%b3n_Andres%20Paucar%20%281%29.pdf

Rodriguez, D.; Talay, C. & Gonzales, C. (s.f.). Explorando las redes definidas por software (SDN). Disponible: http://sedici.unlp.edu.ar/bitstream/handle/10915/103546/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y

Rodríguez, E. (2020). *Diseño y simulación de una red definida por software (SDN) para la implementación de un laboratorio avanzado de datos para la EP de Telecomunicaciones de la Facultad de Ingeniería Electrónica y Eléctrica de la Universidad Nacional Mayor de San Marcos.* [Tesis de licenciatura, Universidad Nacional Mayor de San Marcos]. Disponible: https://cybertesis.unmsm.edu.pe/bitstream/handle/20.500.12672/16021/Rodriguez_ge.pdf?sequence=1&isAllowed=y

Roncero, O. (2014). Software Defined Networking. [Master, Universidad Politécnica de Catalunya]. Disponible: https://upcommons.upc.edu/bitstream/handle/2099.1/21633/Memoria.pdf

Ruipérez, J. (2021). *Seguridad em Redes definidas por software (SDN).* [Tesis de Licenciatura, Universidad Politécnica de Valencia]. Disponible: https://riunet.upv.es/bitstream/handle/10251/165154/Ruip%c3%a9rez%20-%20Seguridad%20en%20Redes%20definidas%20por%20software%20%28SDN%29.pdf?sequence=1&isAllowed=y#page=8&zoom=100,109,915

Serrano, D. (2015). *Redes definidas por software SDN: Openflow.* [Tesis de Licenciatura, Universidad Politécnica de Valencia]. Disponible: https://riunet.upv.es/bitstream/handle/10251/62801/SERRANO%20-%20Redes%20Definidas%20por%20Software%20(SDN):%20OpenFlow.pdf?sequence=3

Ulloa, L. (2009). *La virtualización y su impacto en las ciencias computacionales.* Disponible: https://www.redalyc.org/pdf/6139/613965348014.pdf

Valles W. (2022). *Diseño de una red definida por software (SDN) para brindar una gestión centralizada de las configuraciones y una adecuada gestión de crecimiento de los nodos a la red de un operador de servicios.* [Tesis de licenciatura, Universidad Peruana de Ciencias Aplicadas]. Disponible: https://repositorioacademico.upc.edu.pe/bitstream/handle/10757/660126/Valles_RW.pdf?sequence=3&isAllowed=y

Yagüez, P. (2015). *Programación de redes SDN mediante el controlador POX*. [Tesis de Licenciatura, Universidad Politécnica de Cartagena]. Disponible: https://repositorio.upct.es/bitstream/handle/10317/5254/tfg729.pdf?sequence=1

Zhuma, E.R. (2019). *Análisis del desempeño de redes definidas por software (SDN) frente a redes con arquitectura TCP/IP*. [Tesis de licenciatura, Universidad Técnica Estatal de Quevedo]. Disponible: https://repositorio.uteq.edu.ec/server/api/core/bitstreams/90c2e774-2c12-4fab-993a-d4bc5894bc49/content

ANEXOS

Anexo A: Documento de solicitud para el uso de información.

UNIVERSIDAD RICARDO PALMA
LICENCIAMIENTO INSTITUCIONAL RESOLUCIÓN DEL CONSEJO DIRECTIVO Nº 040-2016-SUNEDU/CD
Facultad de Ingeniería

Santiago de Surco, 14 de diciembre de 2023

Oficio N° 029-2023-FI-TITES-CG

Doctora
OFELIA ROQUE PAREDES
Directora de la Oficina Central de Informática y Cómputo
Universidad Ricardo Palma
Presente.

Asunto: Solicito brindar facilidades a tesista del IX Programa de Programa de Titulación por Tesis

De mi especial consideración:

Es grato dirigirme a usted para expresarle mi cordial saludo y al mismo tiempo presentar a los tesistas: **Giovanny Joel Rojas Vargas**, con DNI 70350866 y **Freddy Alonso Cruz Guerrero**, con DNI N° 76256722, participantes de la IX Edición del Programa de Titulación por Tesis 2023, modalidad virtual, de la Escuela Profesional de Ingeniería Electrónica, quienes han desarrollado la tesis titulada: "Diseño de una red definida por software (SDN) mediante el protocolo Openflow para la Universidad Ricardo Palma, Lima 2023", bajo la asesoría del Ing. Cuadrado Lerma, Luis Alberto.

Para culminar la fase de revisión de tesis, es necesario que se incluya una carta de autorización de su oficina, autorizando el uso de información, toda vez que la tesis se basa en la Arquitectura de red de la Universidad Ricardo Palma.

Sin otro particular, agradezco el especial apoyo que brinde a nuestros futuros ingenieros.

Atentamente,

Dr. Ing. SANTIAGO FIDEL ROJAS TUYA
Decano

SFRT/rch.

Anexo B: Documento de compromiso para el buen uso de datos.

UNIVERSIDAD RICARDO PALMA

DOCUMENTO DE COMPROMISO DE BUEN USO DE DATOS PERSONALES DE LA BASE DE DATOS DE ESTUDIANTES, DOCENTES Y NO DOCENTES

Yo, Giovanny Joel Rojas Vargas con número de documento (DNI / CE): 70350866, que soy colaborador de la Unidad Orgánica (Facultad/ Oficina / Departamento)_____ con la responsabilidad: _____. Me comprometo a cumplir con

PRIMERO. – Contribuir al cumplimiento de las disposiciones establecidas en la Ley N° 29733, Ley de Protección de Datos Personales, su Reglamento, aprobado por Decreto Supremo N° 003-2013-JUS, y sus normas complementarias.

SEGUNDO. – Respetar el Marco Normativo:

Las principales normas son las siguientes:

- Constitución política de Perú de 1993
- Ley N° 29733, Ley de Protección de Datos Personales
- Decreto Supremo N° 003-2013-JUS, por medio del cual se aprueba el Reglamento de la Ley N° 29733, Ley de Protección de Datos Personales.

TERCERO. - Respetar el Compromiso de la Universidad Ricardo Palma, con domicilio en Av. Benavides 5440, distrito de Santiago de Surco, declara ser la titular del Banco de Datos Personales e informa que los destinatarios de los datos personales serán las unidades académica o administrativa de la Universidad, las cuales conservará los datos personales permanentemente o hasta que sean modificados dependiendo de la naturaleza de los mismos; estos datos se utilizarán para ***contactarse*** y ***enviar información sobre ofertas educativas, seguimiento de los procesos académicos,*** para ***envío de publicidad mediante cualquier medio y soporte, fines estadísticos*** y en general para el ***cumplimiento de la prestación de servicios*** y cubrir las necesidades de sus interesados. Asimismo, estos datos se mantendrán mientras sean útiles para que la Universidad pueda prestar y ofrecer servicios.

UNIVERSIDAD
RICARDO PALMA

CUARTO. – Cumplir con las prohibiciones que se detallan continuación:

Los colaboradores de la Universidad Ricardo Palma no deberán:

- Utilizar los datos personales obtenidos para una finalidad distinta de por la cual fueron recabados.
- Usar la información para beneficio propio o de terceros.
- Tratar los datos personales sin respetar los principios señalados en el numeral 3 del presente documento.
- Incumplir su deber de confidencialidad.

En señal de conformidad, firmo el presente Documento de compromiso de buen uso de datos personales de la base de datos de estudiantes, docentes y no docentes.

Surco, 19 de diciembre de 2023

UNIVERSIDAD RICARDO PALMA
DECANO
FACULTAD DE INGENIERÍA

Rojas Santiago

Firma

Nombres y Apellidos: Giovanny Joel Rojas Vargas

Cargo: Tesista de la escuela de Ingeniería Electrónica

Anexo C: Solicitud de información firmada.

OFICINA CENTRAL DE INFORMATICA Y COMPUTO-RECTORADO

OFICIC **HOJA DE SERVICIO GENERAL**

N°. HOJA	3200-2023-OFICIC-D
VERSIÓN	VH. 1.00
FECHA	15/09/2023

REQUERIMIENTO: SOLICITUD

INCIDENCIA	
MEJORA	SOLICITUD DE INFORMACIÓN

TIPO DE SOLICITANTES	AREAS INTERNAS		AREAS EXTERNAS		ADMINISTRATIVO	
OFICIC - ÁREA	ÁREA DE SISTEMAS DE INFORMACIÓN Y COMUNICACIÓN				
UNIDAD QUE ATIENDE	UNIDAD DE GESTIÓN DE REDES Y CONECTIVIDADES				
NOMBRE DEL PERSONAL	NOMBRES Y APELLIDOS	WALTER GONZÁLES CRISANTO			
	CARGO	COORDINADOR DE UNIDAD			
FECHA DE RECEPCIÓN	29/08/2023		FECHA DE ENTREGA	15/09/2023	

SOLICITUD:

DETALLE DEL SERVICIO	Buenas tardes Giovanny: De acuerdo al requerimiento que registró el dias 29/08/23 y que se adjunta en el presente correo en el cual solicita información sobre la red URP por el tema de sus tesis que fue validado por la directora de la OFICIC, procedo a enviarle la información que solicito: Solicitamos: 1- arquitectura de la red URP (switch's , router , firewall , arquitectura capa 1 y 2) 2- distribución de VLANS **RTA a 1** Archivos adjuntos *"Arquitectura de red cableada URP.jpg"* y *"Arquitectura de red inalámbrica URP.jpg"* **RTA a 2** Se le indica que: ▪ En la Universidad existen 2 grupos de vlans: Vlans académicas y Vlans administrativas. ▪ - Existen en total 37 vlans, siendo la distribución la siguiente: 24 vlans académicas y 13 vlans administrativas
RECOMENDACIONES	Buen uso de los datos brindado.

Ing. Walter Gonzáles Crisanto	Ing. Walter Gonzáles Crisanto	Dra. Ofelia Roque Paredes
Firma del Personal que atendió	Vo Bo Coordinador de Unidad	Vo Bo de Dirección

Realizado por: Auxiliar Yvette Maite Zarzosa Baez *Fecha: 10/02/2023*

Av. Benavides N° 5440 - Santiago de Surco Lima 33 - Perú Aptado Postal 1801 ☎ (51 - 1) 708 0000 - Anexos: 0247-0555

OFICINA CENTRAL DE INFORMATICA Y COMPUTO-RECTORADO

OFICIC **HOJA DE SERVICIO**

N°. HOJA	-2023-AIT-UGS
VERSIÓN	VH. 1.02
FECHA	15/09/2023

ÁREA DE INFRAESTRUCTURA TECNOLÓGICA

SOPORTE DE EQUIPOS INFORMÁTICOS

REQUERIMIENTO:	-2023-OFICIC-D	FECHA DE RECEPCIÓN	29/08/03.
INCIDENCIA		MEJORA	X

UNIDAD RESPONSABLE	Facultad de Ingeniería Electrónica	PISO	2	N°	
UNIDAD INTERNA	Escuela de Ingeniería Electrónica				
USUARIO	Santiago Rojas Tuya	CARGO	Decano de la Facultad		
UBICACIÓN					

				N.° PC	PC
DATOS DE EQUIPO					
MARCA/MODELO		SERIE		Cod. Patrimonio	

SOLICITUD:

DETALLE DEL SERVICIO:
- Se entrego el diagrama de la arquitectura de red de datos de la Universidad Ricardo Palma.
- Se indicó la topologia de red de los equipos sin proporcionar mayor detalle.
- Se indico la cantidad de vlans en la universidad y su distribución sin proporcionar mayor detalle
RECOMENDACIONES:

PARTES A SOLICITAR	REQUERIDOS ○	INSTALADOS ○

CANTIDAD	DESCRIPCIÓN	N° PARTE

SOLUCIONADO ○ RETIRO DE EQUIPO ○ COMPRA DE REPUESTOS ○

CAMBIO DE EQUIPO ○ MIGRACION DE EQUIPO ○ DEVOLUCIÓN DE EQUIPO ○

Giovanny Joel Rojas Vargas	Walter González Crisanto	V°. Bo.
Usuario Solicitante	**Técnico del Soporte**	**Soporte – AIT**

Realizado por: Téc. T.I. Alan Rodriguez H. *Ver. HS-01 Fecha: 09/02/2023*