
**UNIVERSIDAD TECNICA DE BABAHOYO**

**FACULTAD DE ADMINISTRACION FINANZAS E INFORMATICA**

**ESCUELA DE TECNOLOGIAS DE LA INFORMACION Y LA COMUNICACIÓN**

**SISTEMAS DE INFORMACION (REDISEÑADA)**

**TEMA**

Diseño de una infraestructura de SDN utilizando herramientas de código abierto como OpenFlow y Mininet en entornos de simulación para análisis de desempeño.

**ALUMNO**

Cercado Solórzano Christopher Elian

**TUTOR**

Ing. Peñaherrera Larenas Milton Fabian

**PERIODO**

Abril – Septiembre 2025

## DEDICATORIA

Dedico este trabajo a Dios, por guiarme y darme la fortaleza para alcanzar una meta más en mi vida. A mis padres, por su apoyo incondicional, su amor y sacrificios, que han sido el motor de mi perseverancia. A mis hermanos, quienes son mi inspiración y mi mayor motivación para seguir adelante.

# AGRADECIMIENTOS

Agradezco profundamente a Dios por haberme dado la fortaleza, la sabiduría y la perseverancia necesarias para culminar con éxito este importante proyecto. Sin su guía y bendiciones, este logro no habría sido posible.

Mi sincera gratitud a mis padres, quienes, con su apoyo incondicional, su amor, su paciencia y sus sacrificios a lo largo de mi vida, me han inculcado los valores de la dedicación y la constancia, siendo mi mayor inspiración y mi pilar fundamental en cada etapa de este camino.

Extiendo mi agradecimiento al Ing. Peñaherrera Larenas Milton Fabian, mi tutor, cuya valiosa orientación, conocimientos, paciencia y profesionalismo fueron clave para la realización de este trabajo. Su guía fue fundamental para superar cada desafío y alcanzar los objetivos propuestos.

Agradezco a la Universidad Técnica de Babahoyo, la institución que me abrió sus puertas y me brindó las herramientas, los conocimientos y las oportunidades para mi formación académica y profesional.

De igual manera, mi gratitud a mis profesores, por su dedicación, por compartir su experiencia y por contribuir al desarrollo de mis habilidades y competencias. Su labor ha sido fundamental en mi crecimiento como profesional en el campo de las Tecnologías de la Información y la Comunicación.

Finalmente, agradezco a mis compañeros de clase y a todas aquellas personas que, de una u otra forma, me brindaron su apoyo y colaboración durante el proceso de investigación y desarrollo de este proyecto

# RESUMEN

El proyecto de investigación se centra en el diseño de una infraestructura de red definida por software (SDN) en un entorno de simulación, utilizando herramientas de código abierto como OpenFlow y Mininet. El objetivo principal es evaluar el rendimiento de esta infraestructura, midiendo métricas como latencia, rendimiento (rendimiento) y pérdida de paquetes. La investigación surge de la problemática de la falta de equipos físicos especializados en universidades, que limita la enseñanza práctica de redes avanzadas. Se propone una solución económica y versátil a través de la simulación, que permite replicar y comparar resultados de forma confiable.

La metodología utilizada es de tipo aplicada, experimental y descriptiva. Se manipularon variables en un entorno controlado para evaluar el desempeño de la red, utilizando tres tipos de topologías (estrella, árbol, malla) y tres controladores SDN (ONOS, Floodlight, OpenDaylight). Los resultados muestran que ONOS tuvo los valores de rendimiento más altos en las topologías de árbol (3022) y malla (2231). En términos de confiabilidad, ONOS y OpenDaylight lograron una pérdida de paquetes del 0% en todas las topologías, demostrando su alta confiabilidad. Floodlight, aunque tuvo un desempeño sobresaliente en la topología de malla con 0% de pérdida, presentó pérdidas mínimas en las topologías de árbol y estrella. La latencia fue similar en los tres controladores, fluctuando alrededor de 11200 ms. El estudio concluye que la simulación es una herramienta eficaz para la formación en SDN y que el diseño propuesto puede ser replicado para mejorar la calidad de las prácticas académicas.



Palabras claves: SDN, OpenFlow, Mininet, Rendimiento, Latencia, Perdida de paquetes, Simulación.

# ABSTRACT

The research project focuses on the design of a software-defined network (SDN) infrastructure in a simulation environment, using open-source tools such as OpenFlow and Mininet. The main objective is to evaluate the performance of this infrastructure, measuring metrics such as latency, throughput (performance), and packet loss. The research arises from the problem of the lack of specialized physical equipment in universities, which limits the practical teaching of advanced networks. An economical and versatile solution is proposed through simulation, which allows results to be reliably replicated and compared.

The methodology used is applied, experimental, and descriptive. Variables were manipulated in a controlled environment to evaluate network performance, using three types of topologies (star, tree, mesh) and three SDN controllers (ONOS, Floodlight, OpenDaylight). The results show that ONOS had the highest performance values in tree (3022) and mesh (2231) topologies. In terms of reliability, ONOS and OpenDaylight achieved 0% packet loss in all topologies, demonstrating their high reliability. Floodlight, although it performed outstandingly in the mesh topology with 0% loss, showed minimal losses in the tree and star topologies. Latency was similar across all three controllers, fluctuating around 11200 ms. The study concludes that simulation is an effective tool for SDN training and that the proposed design can be replicated to improve the quality of academic practices.



Keywords: SDN, OpenFlow, Mininet, Performance, Latency, Packet Loss, Simulation.

# CONTENIDO

## INDICE DE TABLAS

INDICE DE FIGURAS

# CAPÍTULO I – INTRODUCIÓN

## 1.1 Contextualización de la situación problemática

### 1.1.1 Contexto internacional

Las Redes Definidas por Software (SDN) han transformado la forma en que se administran y operan las redes, gracias a la separación entre el plano de control y el plano de datos. Esta arquitectura permite una gestión centralizada, flexible y programable, mejorando la eficiencia y la adaptabilidad de las redes. Empresas como Google, Amazon y Facebook utilizan SDN en sus centros de datos para optimizar el uso del ancho de banda, reducir los costos operativos y aumentar la disponibilidad de los servicios.

El protocolo OpenFlow, uno de los más utilizados en entornos SDN, se ha consolidado como estándar para la comunicación entre controladores y dispositivos de red. Herramientas como Mininet permiten simular entornos complejos de forma económica y segura, lo que ha facilitado a universidades de prestigio como Stanford, MIT y Berkeley desarrollar prácticas y proyectos sin necesidad de equipos físicos costosos.

Diversos estudios han evaluado el desempeño de los controladores SDN y su aplicabilidad en distintos escenarios. Por ejemplo, (Singh et al., 2022)analizaron el desempeño de controladores como ONOS y OpenDaylight en entornos académicos, demostrando el interés creciente de instituciones ecuatorianas en consolidar prácticas experimentales con SDN. Asimismo, (Ruchel et al., 2022a)evaluaron la robustez de estos controladores frente a fallos en el plano de control y datos, resaltando su resiliencia en entornos críticos. Más recientemente, (Montazerolghaem & Imanpour, 2025)analizaron el controlador Ryu en distintos escenarios de red simulados con Mininet, confirmando su aplicabilidad en entornos educativos y de investigación. Estas

experiencias reflejan que la adopción de SDN a nivel global no solo responde a intereses industriales, sino también a la consolidación de un ecosistema académico y de estandarización que impulsa su evolución.

### 1.1.2 Contexto nacional

En Ecuador, la adopción de SDN aún es limitada, principalmente debido a la falta de especialistas y al alto costo de implementación en entornos reales. Sin embargo, algunas universidades han comenzado a incorporar herramientas de simulación como Mininet para acercar a los estudiantes a las tecnologías de redes programables. A pesar de estos avances, muchas de estas prácticas se realizan de forma aislada, sin un diseño previo que permita medir y comparar el rendimiento de las configuraciones de red.

Esto significa que, si bien existe interés académico en SDN, no siempre se obtienen resultados claros y útiles para mejorar el aprendizaje o aplicar los conocimientos en proyectos más avanzados. En este sentido, algunos trabajos han marcado un precedente en el país: (López Huera, 2021) implementaron un entorno de simulación de SDN con Mininet en la Universidad Técnica del Norte, evidenciando su aplicabilidad como herramienta educativa. De igual manera, (Haro & Fernando, 2021) exploró el uso de SD-WAN como mecanismo de seguridad en accesos WAN, demostrando la aplicabilidad de tecnologías basadas en SDN en entornos corporativos. Por otro lado, (Medina Magües & Tenezaca Mawyin, 2023)analizaron el desempeño de controladores como ONOS y OpenDaylight en entornos académicos, demostrando el interés creciente de instituciones ecuatorianas en consolidar prácticas experimentales con SDN.

En conjunto, estas iniciativas muestran que, aunque incipiente, la investigación en SDN en Ecuador avanza desde el ámbito académico hacia aplicaciones prácticas, lo

cual abre un espacio importante para proyectos que fortalezcan tanto la enseñanza como el uso experimental de estas tecnologías.

### 1.1.3 Contexto local

En el contexto local, la mayoría de los laboratorios universitarios carecen del hardware necesario para implementar físicamente una red SDN. Esto obliga a recurrir a simuladores como Mininet, que ofrecen una alternativa económica y versátil para crear topologías de red y trabajar con protocolos como OpenFlow.

Sin embargo, se ha identificado que las prácticas realizadas en estos entornos carecen de un diseño estructurado que permita evaluar métricas clave como la latencia, la pérdida de paquetes y el rendimiento. Esto reduce la calidad de la experiencia de aprendizaje y limita el valor de los resultados obtenidos, ya que no son fácilmente repetibles ni comparables.

## 1.2 Planteamiento del problema

En muchas universidades y centros educativos con recursos limitados, la enseñanza de redes avanzadas se enfrenta a una barrera importante: la falta de equipos especializados para trabajar con tecnologías emergentes como las Redes Definidas por Software (SDN). Esto impide a los estudiantes realizar prácticas en entornos reales, lo que limita su comprensión y habilidades técnicas.

Aunque existen herramientas de simulación como Mininet y el protocolo OpenFlow, que permiten experimentar con SDN sin necesidad de hardware costoso, en la práctica estas simulaciones suelen configurarse de forma improvisada, sin un diseño previo que permita obtener resultados claros sobre el rendimiento de la red. Como consecuencia, estudiantes e investigadores carecen de datos precisos sobre métricas como la latencia, la pérdida de paquetes o el rendimiento, y las pruebas no siempre pueden repetirse para obtener los mismos resultados.

Esto crea una brecha entre el conocimiento teórico y la experiencia práctica, lo que dificulta la formación de profesionales capaces de aplicar SDN en entornos reales.

Ante esta situación, se plantea la necesidad de diseñar una infraestructura SDN en un entorno simulado, utilizando herramientas de código abierto como OpenFlow y Mininet, que permita implementar diferentes topologías de red y medir su comportamiento de forma fiable. Con esto, se busca optimizar el aprendizaje práctico, facilitar la comprensión de SDN y generar resultados útiles para futuros proyectos académicos.

### 1.3 Delimitación de la investigación

**Espacial:** Entorno de simulación utilizando Mininet en equipos locales.

**Temporal:** Abril – Septiembre de 2025.

**Contenido:** Diseño de infraestructura SDN con OpenFlow y Mininet, evaluación de métricas de rendimiento (latencia, rendimiento, pérdida de paquetes).

### 1.4 Justificación

El estudio es relevante porque ofrece una solución práctica para la enseñanza y experimentación con redes SDN en contextos donde no se dispone de infraestructura física especializada. Al utilizar herramientas de código abierto como Mininet y OpenFlow, el coste de implementación se reduce significativamente y se facilita el acceso a pruebas controladas y repetibles.

Con un enfoque académico, este trabajo permitirá a estudiantes e investigadores desarrollar habilidades técnicas en el diseño y análisis de redes SDN, trabajando con métricas concretas que proporcionan datos fiables para la toma de decisiones. Desde un enfoque práctico, se obtendrá un diseño de infraestructura en simulación que podrá

replicarse en otras instituciones, mejorando así la calidad de las prácticas y proyectos en el área de redes.

### 1.5 Objetivos de la investigación

#### 1.5.1 Objetivo general

Diseñar una infraestructura de red definida por software (SDN) en un entorno de simulación, utilizando herramientas de código abierto como OpenFlow y Mininet, para evaluar métricas de rendimiento en diferentes topologías de red.

#### 1.5.2 Objetivos específicos

- Analizar los fundamentos técnicos de las redes definidas por software y el protocolo OpenFlow.
- Bosquejar las topologías de red en Mininet para simular diferentes escenarios.
- Evaluar el rendimiento de cada topología en términos de latencia, pérdida de paquetes y rendimiento.

### 1.6 Hipótesis

El diseño estructurado de una infraestructura SDN en entornos simulados, utilizando OpenFlow y Mininet para implementar diferentes topologías de red, permitirá obtener mediciones precisas y repetibles de métricas de rendimiento como latencia, pérdida de paquetes y throughput, reduciendo la brecha entre el conocimiento teórico y la experiencia práctica en la enseñanza de redes avanzadas en instituciones con recursos limitados

## CAPÍTULO II - MARCO TEÓRICO

### 2.1 Antecedentes

Diversos estudios han demostrado la eficacia de las Redes Definidas por Software (SDN) en la gestión de redes modernas. (Lasso Guamán & Puchaicela Condoy, 2021) implementaron un prototipo SDN utilizando OpenFlow y herramientas

de código abierto, y encontraron que el controlador OpenDaylight obtuvo una latencia promedio de 0.936 ms con solo un 1.97% de pérdida de paquetes, mientras que Floodlight se destacó en entornos con transferencias de datos masivos, logrando un throughput de 39.478 MB/s.

Por ejemplo, (Ruchel et al., 2022b)evaluaron la robustez de ONOS y OpenDaylight frente a fallos en planos de control y datos, revelando diferencias significativas en continuidad del servicio y tolerancia en despliegues distribuidos.

Adicionalmente, una comparativa publicada en 2022 examinó el desempeño de los controladores POX, Ryu, OpenDaylight y ONOS en redes inalámbricas emuladas (SDWN con Mininet Wi-Fi), midiendo latencia, jitter, pérdida y throughput, permitiendo seleccionar el controlador según requisitos del caso de uso.(Aslan & Al-Somaidai, 2022)

Aunque estos estudios ofrecen resultados positivos, muchos de ellos no cuentan con un diseño metodológico unificado, lo que dificulta la comparabilidad de resultados. En este sentido, el presente trabajo busca contribuir con un enfoque estructurado y replicable que permita evaluar métricas específicas de desempeño bajo diferentes condiciones de red simulada.

## 2.2 Bases teóricas

### 2.2.1 Fundamentos de Redes Definidas por Software (SDN)

SDN es una arquitectura de red que separa el plano de control (donde se toman las decisiones sobre el tránsito de datos) del plano de datos (donde se ejecutan esas decisiones). Esta separación permite un control más flexible, programable y centralizado sobre la red. La arquitectura SDN, sus ventajas (programabilidad, flexibilidad, centralización) y sus retos actuales (interoperabilidad, resiliencia,

seguridad) han sido revisados de forma exhaustiva en estudios recientes, confirmando su papel en redes modernas multi-cloud, edge y 5G.(Rakissaga et al., 2025)

La arquitectura general está compuesta por tres capas principales:

- Capa de Aplicación: Incluye programas que definen políticas de red, monitoreo y seguridad.
- Capa de Control: Ejecuta las decisiones a través de controladores como OpenDaylight, ONOS o Floodlight.
- Capa de Infraestructura: Incluye los dispositivos de red físicos o virtuales, encargados de ejecutar las instrucciones provenientes del controlador.

Esta arquitectura permite que las redes sean más dinámicas y adaptables a los cambios en el tráfico, y reduce los costos al utilizar hardware genérico (commodity hardware) en lugar de equipos especializados.

### 2.2.2 Protocolo OpenFlow

OpenFlow es el protocolo fundamental que facilita la comunicación entre el plano de control y el plano de datos en arquitecturas SDN, actuando como el canal estandarizado por el cual los controladores pueden programar y gestionar el tráfico de los switches de forma dinámica y centralizada. La estructura de OpenFlow está compuesta típicamente por un controlador (que determina las políticas y reglas de reenvío), switches OpenFlow (encargados de ejecutar estas políticas) y un canal seguro cifrado que garantiza integridad y autenticidad en la comunicación. (Merayo et al., 2021)

Sus componentes clave son:

- **Controlador OpenFlow**: Toma decisiones sobre el enrutamiento de paquetes.

- **Switch OpenFlow**: Recibe instrucciones del controlador y ejecuta las acciones especificadas.
- **Canal Seguro**: Se encarga de que la comunicación entre controlador y switches esté cifrada y protegida.

Las versiones recientes de OpenFlow (desde la 1.3 en adelante) han añadido soporte para nuevos protocolos (IPv6, MPLS, Q-in-Q), múltiples tablas de flujo, opciones de meter y group table, y capacidades de extensibilidad que lo han hecho apto no solo para entornos experimentales, sino también para la automatización de redes de misión crítica y la integración en escenarios de 5G, edge y redes residenciales avanzadas.(Shamsoddini et al., 2025)

### 2.2.3 Herramienta de simulación Mininet

Mininet es un emulador de última generación, ampliamente adoptado para simular topologías de red, switches, controladores y hosts virtuales en una sola máquina. Esta herramienta ha sido clave en investigación y docencia por su bajo consumo de recursos y su capacidad para replicar experimentos de manera precisa y flexible.(Khudhair & Athab, 2025)

En los años más recientes, Mininet ha integrado nuevas funciones para la emulación a gran escala y el soporte de servicios diferenciados, permitiendo la simulación realista de tráfico avanzado, colas múltiples y experimentos automatizados. Estas características facilitan el análisis mediante métricas como latencia, pérdida de paquetes y rendimiento, utilizando iperf, ping y scripts en Python para automatización y repetibilidad.(Raca et al., 2022)

### 2.2.4 Controladores SDN

El controlador es el "cerebro" de la red SDN. Entre los más utilizados se encuentran:

**OpenDaylight (ODL)** es una plataforma modular respaldada por la Linux Foundation que se ha consolidado como un controlador versátil y extensible, soportando múltiples protocolos además de OpenFlow. Singh et al. (2022) destacan que ODL ofrece ventajas en entornos donde se requiere compatibilidad amplia y flexibilidad en la administración de red, mientras que Yurekten y Demirci (2022) resaltan su robustez frente a fallos y su capacidad de recuperación en escenarios de producción.

**ONOS (Open Network Operating System)** fue diseñado con un enfoque en alta disponibilidad y escalabilidad, orientado a operadores y entornos de telecomunicaciones. Investigaciones recientes, como las de(Addad et al., 2022), analizan la eficiencia de sus interfaces “Intent” para la gestión de servicios en redes 5G, mientras que (Li et al., 2022)identifican retos en la estabilidad de su rendimiento bajo ciertas condiciones mediante pruebas de fuzzing. (Agnew et al., 2022)muestran además su aplicabilidad en arquitecturas distribuidas para redes críticas como las de energía, reforzando la resiliencia frente a ataques.

**Floodlight**, por su parte, es un controlador ligero, de fácil instalación y ampliamente usado en entornos educativos y de simulación. Aunque no cuenta con la misma escalabilidad de ONOS ni la modularidad de ODL, estudios como el de Altangerel et al. (2021) evidencian que logra un desempeño competitivo en escenarios de laboratorio con Mininet, lo que lo hace adecuado para fines académicos y de experimentación.

En términos comparativos, las investigaciones de (Zhu et al., 2019) y (Rahim et al., 2024) evidencian que ONOS se posiciona como el controlador más robusto en términos de escalabilidad, OpenDaylight como el más versátil e interoperable y Floodlight como el más accesible y práctico para entornos de aprendizaje y pruebas rápidas. Estos hallazgos justifican la selección de los tres controladores en el presente trabajo, pues permiten contrastar comportamientos en diferentes topologías bajo un mismo entorno de simulación, aportando una visión amplia del rendimiento en redes definidas por software.

### 2.2.5 Métricas de desempeño en redes SDN

Para evaluar el comportamiento de una red SDN es necesario medir ciertos indicadores claves, entre ellos:

- **Latencia**: Tiempo que tarda un paquete en llegar de origen a destino, medido en milisegundos.
- **Throughput (rendimiento)**: Cantidad de datos transmitidos exitosamente en un tiempo determinado, medido en Mbps.
- **Pérdida de paquetes**: Porcentaje de paquetes que no llegan a su destino.

Estas métricas permiten establecer comparaciones entre diferentes topologías y controladores, y son fundamentales para validar el diseño de la infraestructura.

### 2.2.6 Calidad de servicio (QoS) en SDN

Una de las ventajas de SDN es la posibilidad de definir políticas de calidad de servicio a nivel centralizado. Esto permite priorizar ciertos tipos de tráfico, como videoconferencias o transmisiones en tiempo real, sobre otros menos críticos. Gracias a esta capacidad, se puede mejorar la experiencia del usuario y la eficiencia de la red. Los

mecanismos de QoS pueden ser implementados mediante reglas OpenFlow y gestionados desde los controladores SDN (Ramírez & Contreras, 2019).

### 2.2.7 Seguridad en redes SDN

Las redes definidas por software han reforzado la seguridad gracias a la capacidad de aplicar políticas dinámicas, firewalls configurables y sistemas de monitoreo flexibles. Sin embargo, siguen existiendo riesgos como la dependencia de un único punto de control, que puede ser objetivo de ataques. Las soluciones actuales se centran en la redundancia, segmentación del plano de control, canales seguros y la incorporación de algoritmos de inteligencia artificial y monitorización predictiva para detectar y mitigar amenazas de manera proactiva. (Kumar et al., 2025)

### 2.2.8 Casos de uso y aplicaciones

Hoy día, las SDN se aplican en centros de datos, telecomunicaciones 5G, edge computing y la nube, debido a su capacidad de orquestación dinámica, optimización de recursos y gestión centralizada. En educación y laboratorio, el uso de Mininet y OpenFlow posibilita la formación práctica en configuración, análisis y simulación de redes complejas, facilitando la experimentación y la comparación de resultados sin requerir costosa infraestructura física.(Ahmad et al., 2025)

## 2.3 Limitaciones y oportunidades de mejora

Aunque SDN ofrece muchas ventajas, existen algunas limitaciones que deben considerarse:

- El rendimiento de herramientas como Mininet puede verse afectado por las capacidades del hardware local.
- La configuración de controladores complejos puede requerir un nivel técnico avanzado.

- Las simulaciones no siempre reflejan con exactitud el comportamiento de una red física.

A pesar de ello, el uso de herramientas de código abierto y la posibilidad de automatizar experimentos ofrecen un entorno propicio para el aprendizaje, el desarrollo de proyectos y la investigación. La estandarización de metodologías de simulación podría ser una línea de mejora para aumentar la confiabilidad de los resultados y facilitar su replicabilidad en otros entornos académicos.

# CAPÍTULO III - METODOLOGÍA

## 3.1 Tipo y diseño de investigación

Esta investigación es de tipo aplicada, ya que busca resolver un problema práctico mediante la implementación de una solución tecnológica simulada. Según su diseño, es experimental, debido a que se manipularán variables dentro de un entorno controlado para observar los efectos que estas producen en el desempeño de la red. Según el nivel de estudio, es descriptiva, porque se pretende describir y analizar el comportamiento de una infraestructura SDN bajo distintas configuraciones. En cuanto a su dimensión temporal, es transversal, ya que la recolección de datos se realiza en un periodo definido y limitado.

Método de investigación: Se empleará un enfoque mixto entre el método inductivo y el analítico. El método inductivo permitirá, a partir de observaciones particulares de las métricas de desempeño en diferentes topologías, obtener conclusiones generales sobre el comportamiento de la infraestructura SDN. Por su parte, el método analítico se aplicará al descomponer las configuraciones en sus elementos constituyentes (topología, controlador, métrica) para examinar sus relaciones y efectos.

Esta investigación adopta un enfoque cuantitativo, experimental y descriptivo. Se considera cuantitativa porque se medirán métricas numéricas como latencia, pérdida de paquetes y throughput. Es experimental ya que se crearán entornos controlados para probar diferentes configuraciones de red simuladas. Y es descriptiva porque se documentarán las características de las topologías implementadas, los controladores utilizados y los resultados obtenidos.

El diseño experimental busca reproducir condiciones similares en cada configuración, para garantizar la comparabilidad de resultados y permitir la replicabilidad del estudio. Este enfoque metodológico se ha recomendado en estudios previos como los de (Lopez-Gomez et al., 2025) y (Canedo et al., 2020), quienes destacan la necesidad de estructuras experimentales bien definidas en evaluaciones de redes SDN.

### 3.2 Operacionalización de variables

**Tabla 1. Operacionalización de variables**

| **Variable** | **Definición conceptual** | **Dimensión** | **Indicadores** | **Instrumento** |
|---|---|---|---|---|
| **Diseño de infraestructura SDN (VI)** | Estructura lógica de red compuesta por topologías y controladores SDN en un entorno simulado. | Tipo de topología<br>Tipo de controlador | Selección de la topología (estrella, árbol, malla).<br>Selección del controlador (ONOS, Floodlight, OpenDaylight). | Entorno de simulación |
| **Evaluación de desempeño** | Proceso de medición de parámetros de | Latencia<br>Throughput | Tiempo promedio de | Entorno de simulación |

| Variable | Definición conceptual | Dimensión | Indicadores | Instrumento |
|---|---|---|---|---|
| | calidad de servicio en redes SDN, relacionados con la eficiencia, confiabilidad y rapidez en la transmisión de datos. | Pérdida de paquetes | retardo (ms). Tasa promedio de transferencia (Mbps). Porcentaje de paquetes perdidos (%) | |

*Nota* elaboración propia

### 3.3 Población y muestra de investigación

#### 3.3.1 Población

La población corresponde al conjunto de todas las configuraciones posibles de infraestructura SDN simulada con herramientas de código abierto.

#### 3.3.2 Muestra

Para delimitar el estudio, se seleccionará una muestra representativa compuesta por 3 tipos de topologías (estrella, árbol y malla) combinadas con 3 controladores SDN (OpenDaylight, ONOS y Floodlight), resultando en 9 escenarios base.

### 3.4 Técnicas e instrumentos de medición

#### 3.4.1 Técnicas

- Emulación de redes mediante Mininet.
- Generación de tráfico sintético mediante iperf y ping.

#### 3.4.2 Instrumentos

- Software Mininet instalado en Ubuntu 24.0.
- Controladores SDN: OpenDaylight, ONOS y Floodlight.

- Iperf y ping

Estas herramientas son ampliamente utilizadas en investigaciones similares (Shafiq et al., 2024), y permiten llevar a cabo experimentos reproducibles con bajo costo.

### 3.5 Procesamiento de datos

Los datos serán analizados utilizando estadísticas descriptivas (media, desviación estándar, mínimo y máximo) para cada métrica. Además, se aplicarán pruebas de ANOVA para identificar si existen diferencias significativas entre los resultados de distintas topologías y controladores. Esta técnica ha sido recomendada por Shafiq et al. (2024) para comparar entornos de red.

### 3.6 Aspectos éticos

Se garantizará la transparencia del proceso mediante la documentación clara de todos los pasos realizados, la utilización de software de código abierto con licencias válidas, y la correcta citación de autores y fuentes de referencia. Además, se tomarán medidas para asegurar la integridad de los datos y la reproducibilidad de los experimentos.

### 3.7 Limitaciones de la metodología

- El rendimiento de las simulaciones puede verse afectado por las capacidades del equipo físico.
- La ausencia de tráfico real limita la validez externa de los resultados.
- Algunas herramientas como OpenDaylight pueden requerir configuraciones complejas y consumir muchos recursos.

A pesar de estas limitaciones, la metodología propuesta permite una evaluación objetiva y replicable de las configuraciones SDN en un entorno de simulación controlado.

# CAPÍTULO IV - RESULTADOS Y DISCUSIÓN

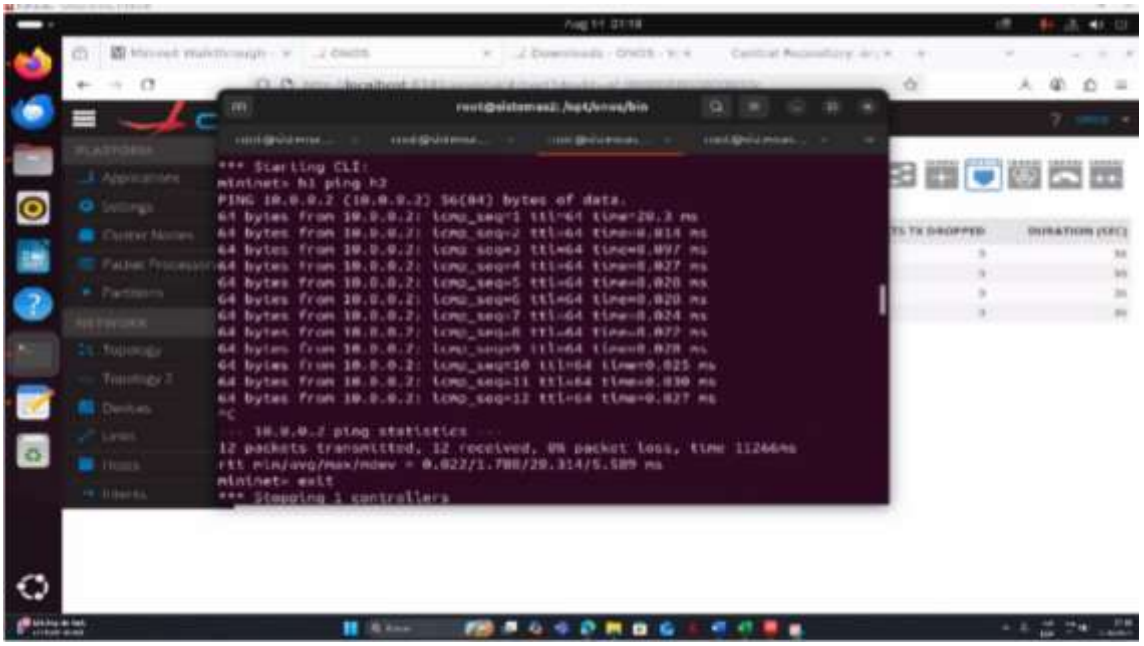
## 4.1 Resultados

Para la aplicación del laboratorio se utilizó una VPS de Azure con una capacidad de 4 núcleos, 8GB de RAM y 64 gigas de disco solido para el almacenamiento, con el sistema operativo Ubuntu versión 24.0. Además, en la intervención se usaron dos tipos de pruebas una donde se usa el pingall para enviar paquetes a todos los equipos y otra donde el host1 envía paquetes al host2.

## ONOS

### Árbol

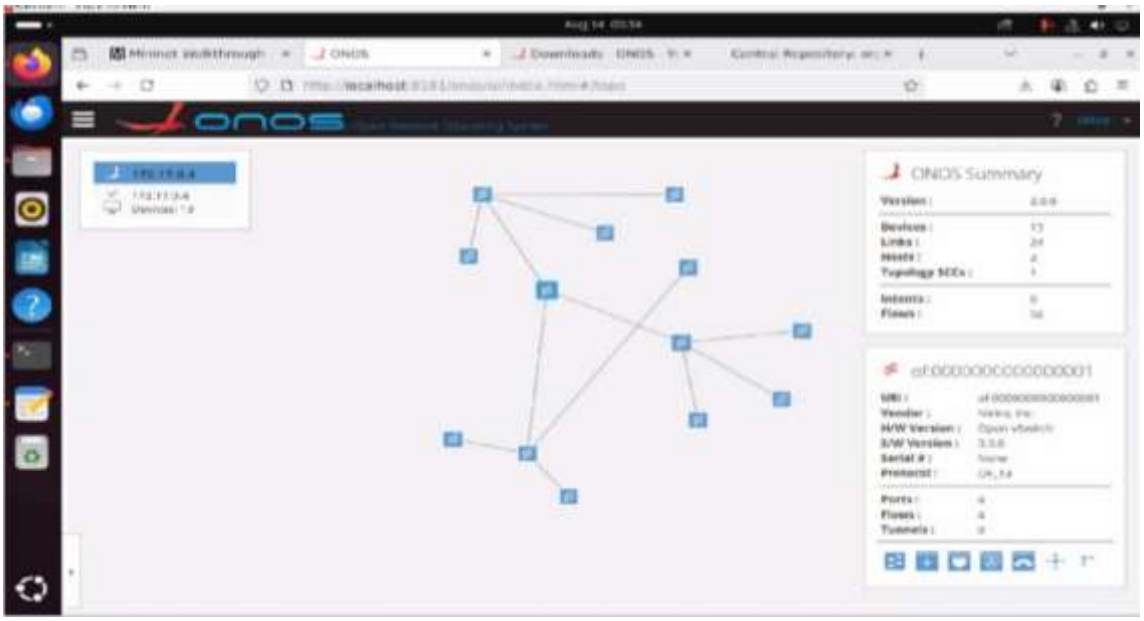
**Figura 1. Prueba de latencia ONOS desde el Mininet**




*Fuente*: Elaboración propia

La **Figura 1** muestra él envió de paquetes desde el host 1 al host 2 usando el comando h1 ping h2 para la prueba de latencia de la topología árbol en el controlador ONOS.

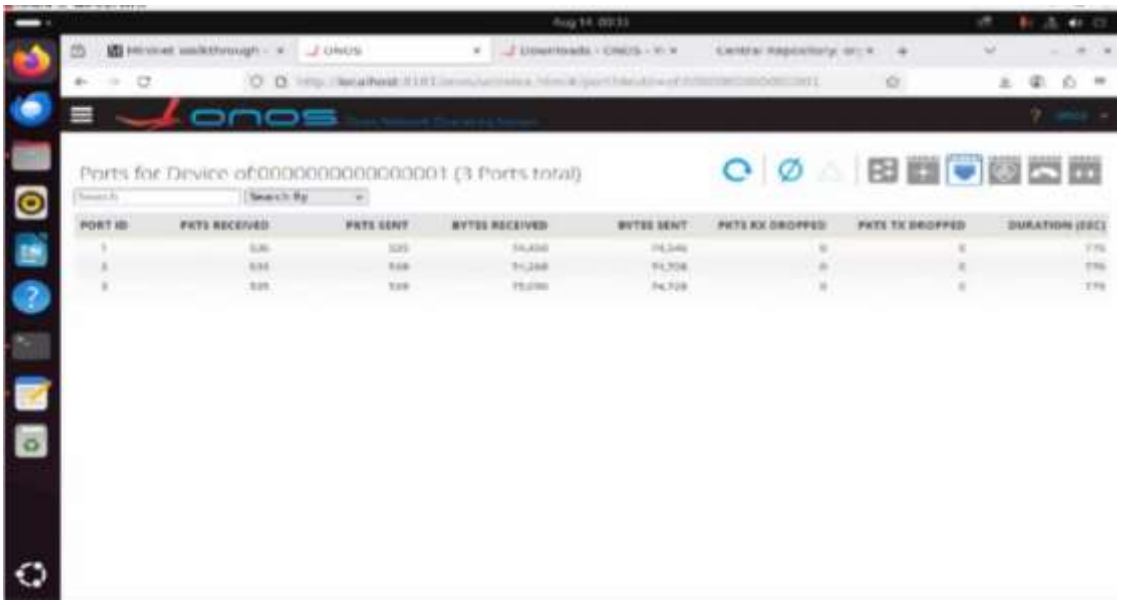
**Figura 2. ONOS topología en árbol**




*Fuente*: Elaboración propia

En **Figura 2** se observa la topología árbol diseñada en controlador ONOS.

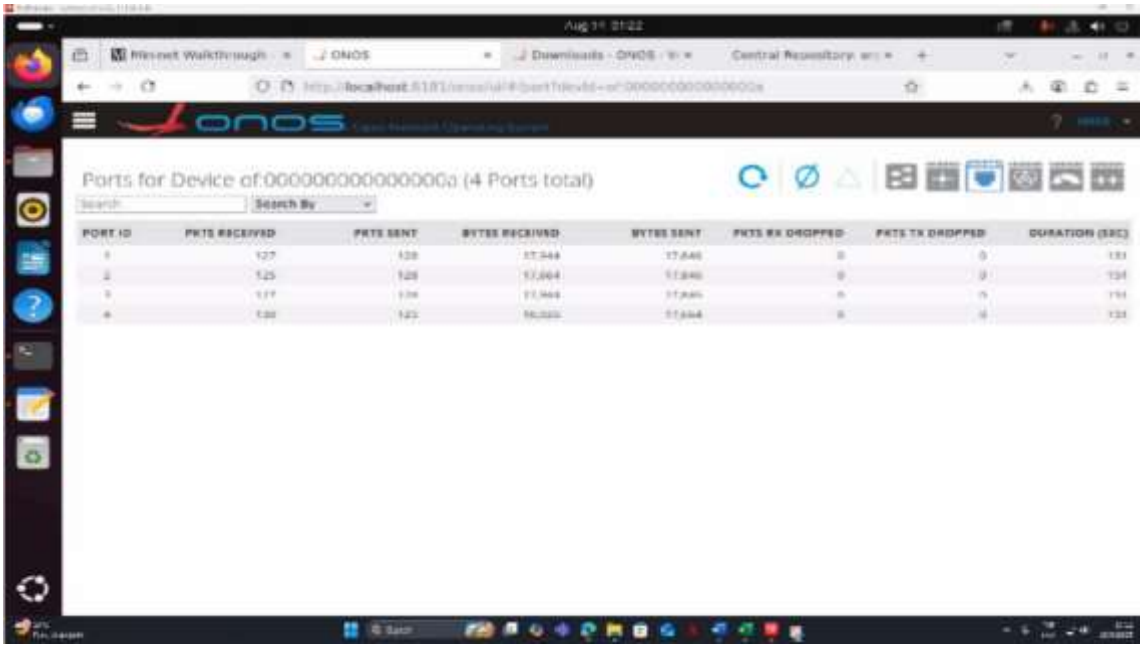
**Figura 3. Visualización de estadística en dispositivo h1 en árbol**



*Fuente*: Elaboración propia

Se visualiza las estadísticas del host 1 de la topología árbol.

**Figura 4. Visualización de estadística en dispositivo h2 en árbol**

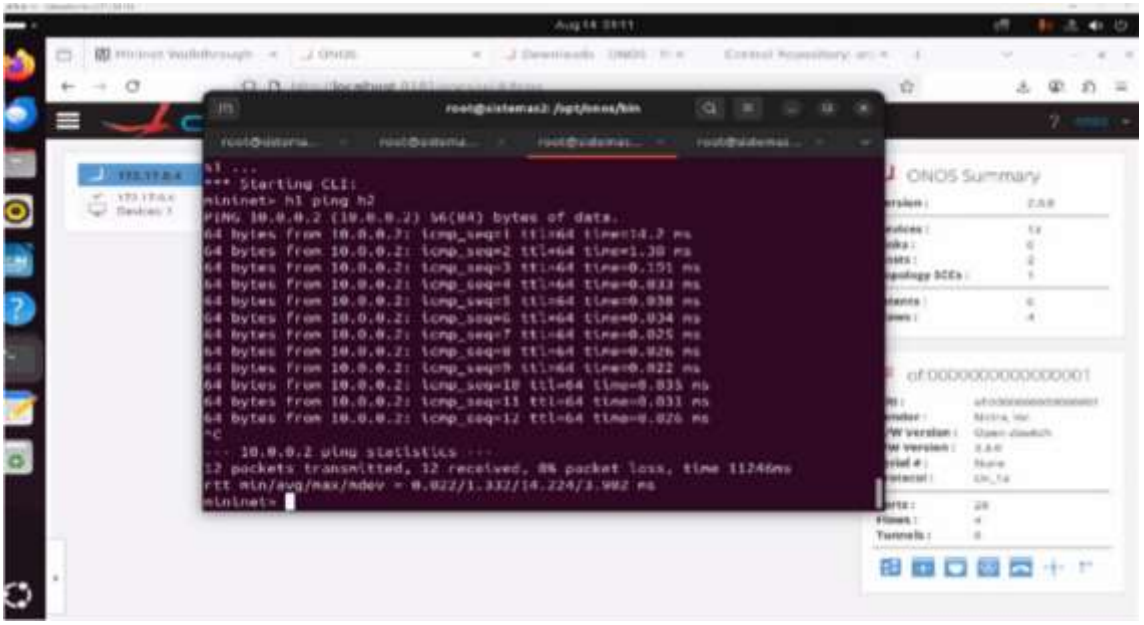



*Fuente*: Elaboración propia

Se visualiza las estadísticas del host 2 de la topología árbol.

**Estrella**

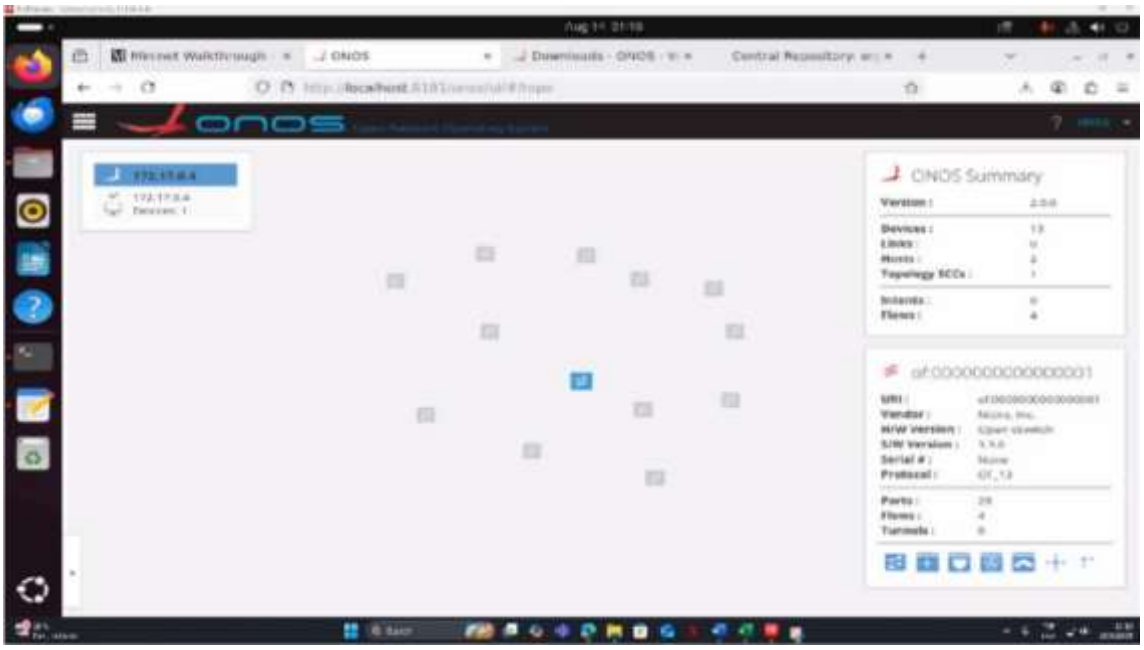
**Figura 5. Prueba de latencia ONOS desde el Mininet**




*Fuente*: Elaboración propia

En la **Figura 5** se muestra él envió de paquetes desde el host 1 al host 2 usando el comando h1 ping h2 para la prueba de latencia de la topología estrella en el controlador ONOS.

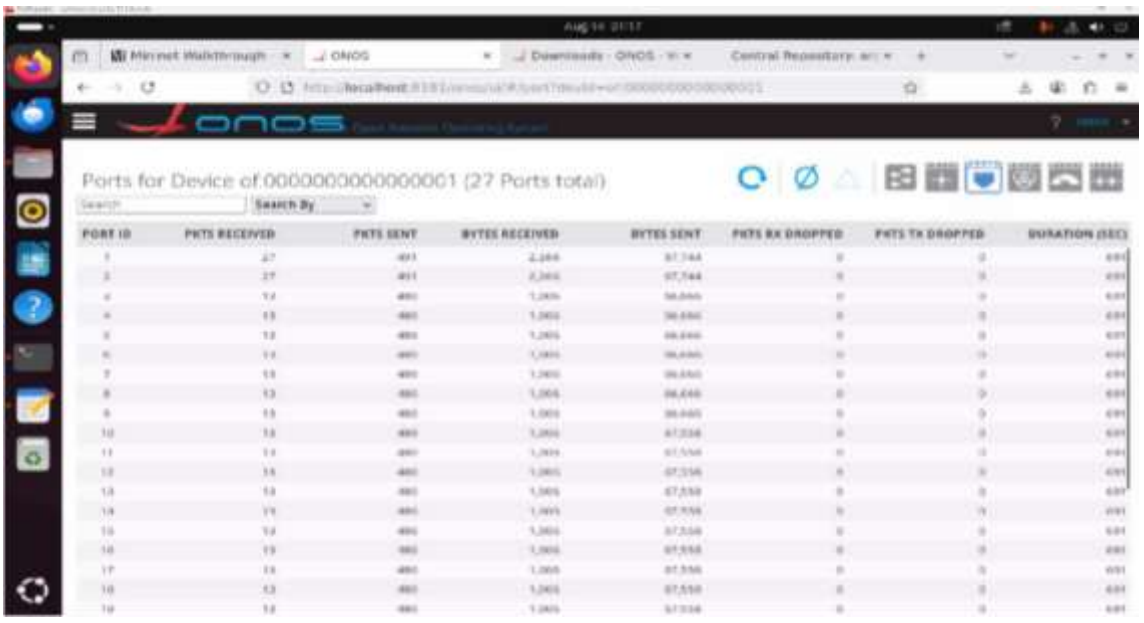
**Figura 6. ONOS topología estrella**




*Fuente*: Elaboración propia

En la **Figura 6** se puede visualizar la topología estrella diseñada en controlador ONOS

**Figura 7. Visualización de estadística en dispositivo h1 en estrella**




*Fuente*: Elaboración propia

Las estadísticas representadas en la **Figura 7** corresponden al host 2 en la topología estrella.

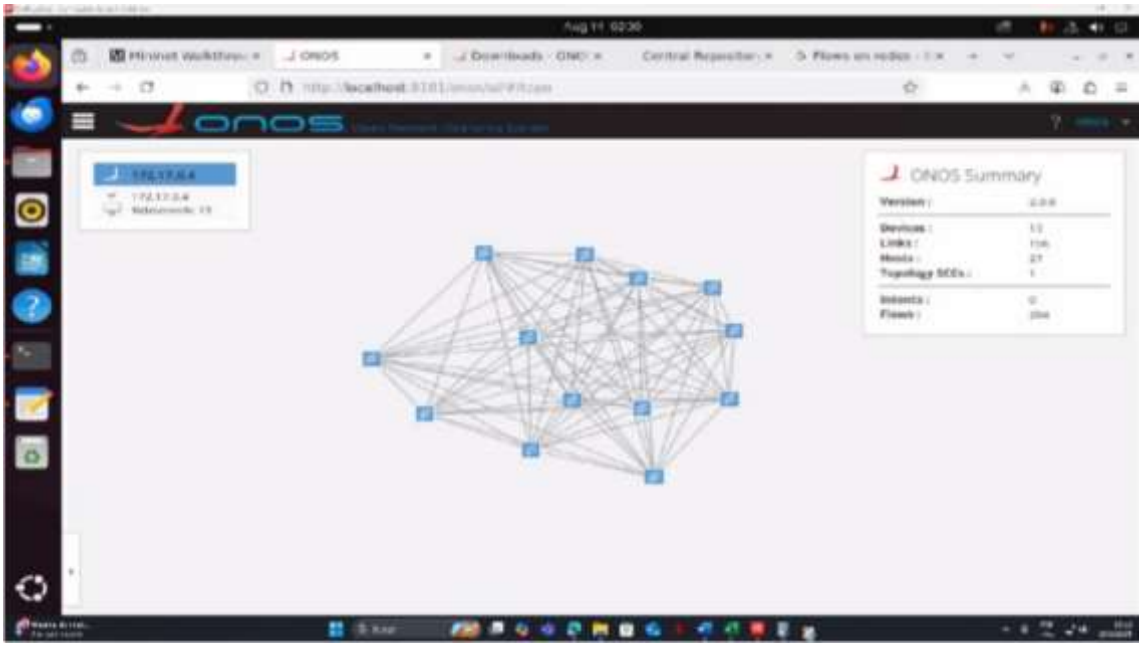
**Figura 8. Visualización de estadística en dispositivo h2 en estrella**


*Fuente*: Elaboración propia

Las estadísticas representadas en la **Figura 8** corresponden al host 2 en la topología estrella.

**Malla**

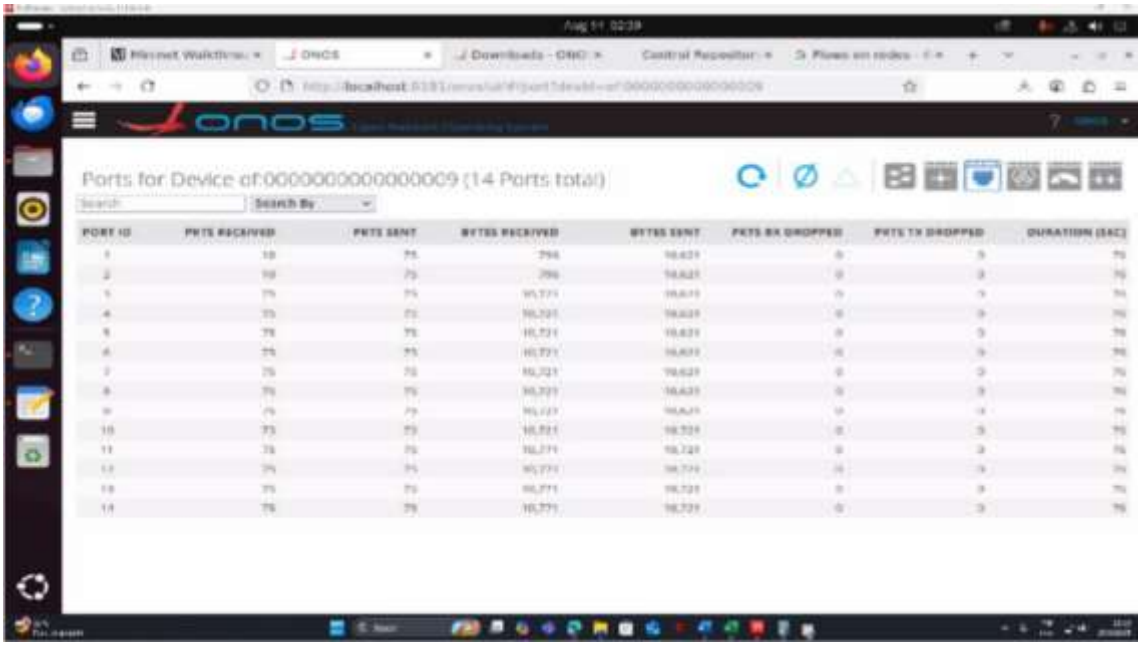
**Figura 9. ONOS topología malla**




*Fuente*: Elaboración propia

En **Figura 9** se visualiza la topología malla diseñada en controlador ONOS.

**Figura 10. Visualización de estadística en malla**



*Fuente*: Elaboración propia

**Figura 10** representa las estadísticas del switch en la topología malla.

**Tabla 2. Prueba de rendimiento en Herramienta ONOS**

| ONOS | | | | | | |
|---|---|---|---|---|---|---|
| Infraestructura | Topología | Trafico | Enviados | Recibidos | Throughput | Perdida de paquetes |
| 1 | Árbol | 3022 | | | 702/702 | 0% |
| 2 | malla | 2231 | | | 702/702 | 0% |
| 3 | estrella | 706 | | | 702/702 | 0% |

Nota: Elaboración propia

**Tabla 2** muestra la comparación de rendimiento de las 3 topologías en el controlador ONOS.

**Tabla 3. Latencia en envió de paquetes ONOS**

| h1-h2 | | |
|---|---|---|
| Latencia | Numero de paquete enviados | Topología |
| 11254 | 12 | Árbol |
| 11236 | 12 | Malla |
| 11263 | 12 | Estrella |

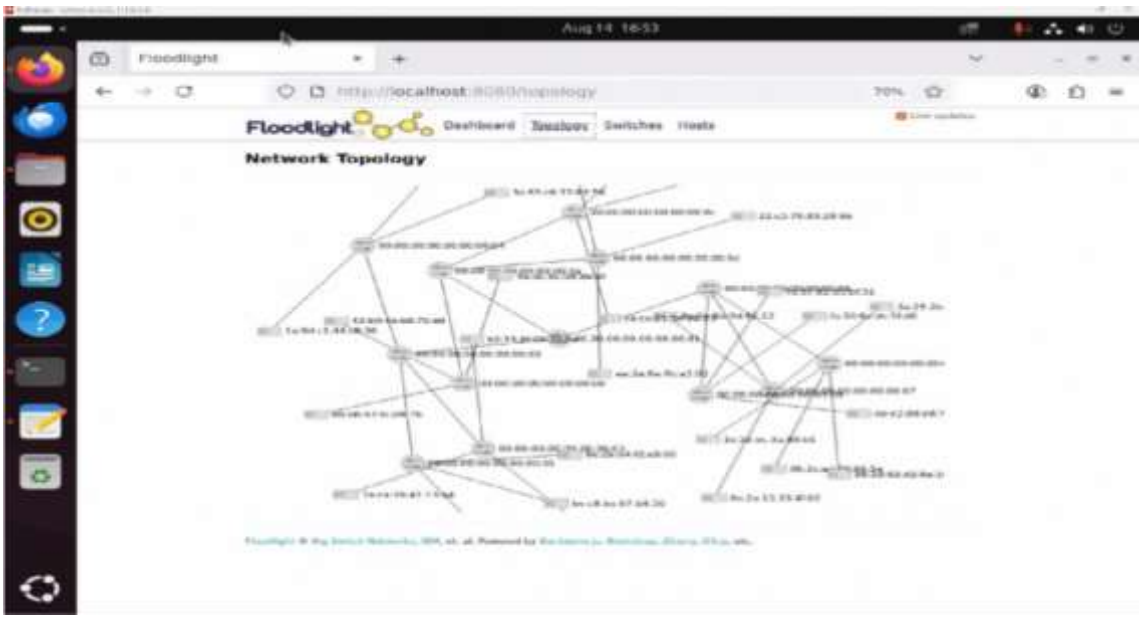
Nota: Elaboración propia

En la **Tabla 3** se compara la latencia entre h1-h2 siendo que la topología malla presenta una menor latencia en comparación con las topologías estrella y árbol.

## Floodlight

### Árbol

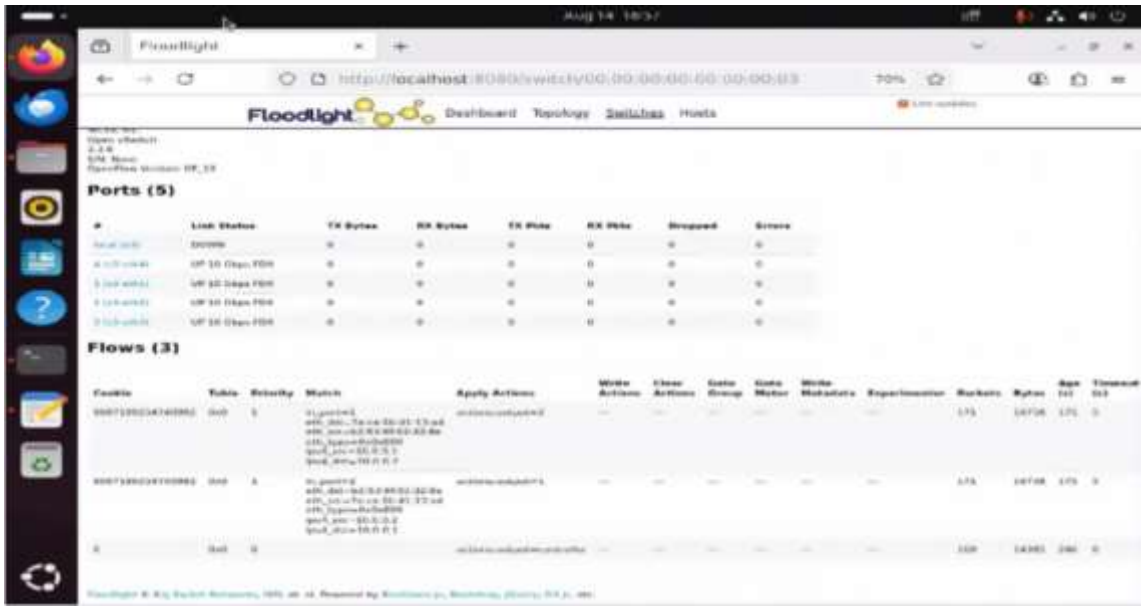
**Figura 11. Floodlight topología árbol**




*Fuente*: Elaboración propia

En la **Figura 11** se puede visualizar la topología árbol diseñada en controlador Floodlight, como se evidencia en la imagen se visualiza los hosts a diferencia de ONOS y OpenDaylight.

**Figura 12. Visualización de estadística en árbol**




*Fuente*: *Elaboración propia*

Se observa en **Figura 12** las estadísticas del switch al realizar el comando ping en la topología árbol.

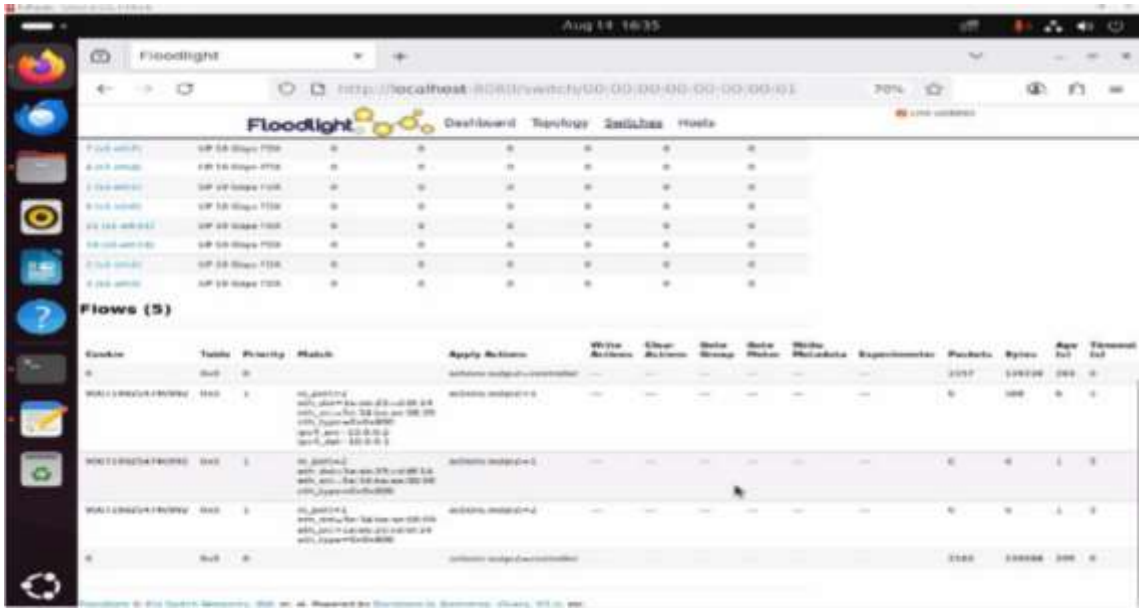
**Estrella**

**Figura 13. Floodlight topología estrella**


*Fuente*: Elaboración propia

En la **Figura 13** se puede visualizar la topología árbol diseñada en controlador Floodlight, incluyendo hosts.

**Figura 14. Visualización de estadística en estrella**




*Fuente*: Elaboración propia

Se observa las estadísticas del switch al realizar el comando ping en la topología estrella.

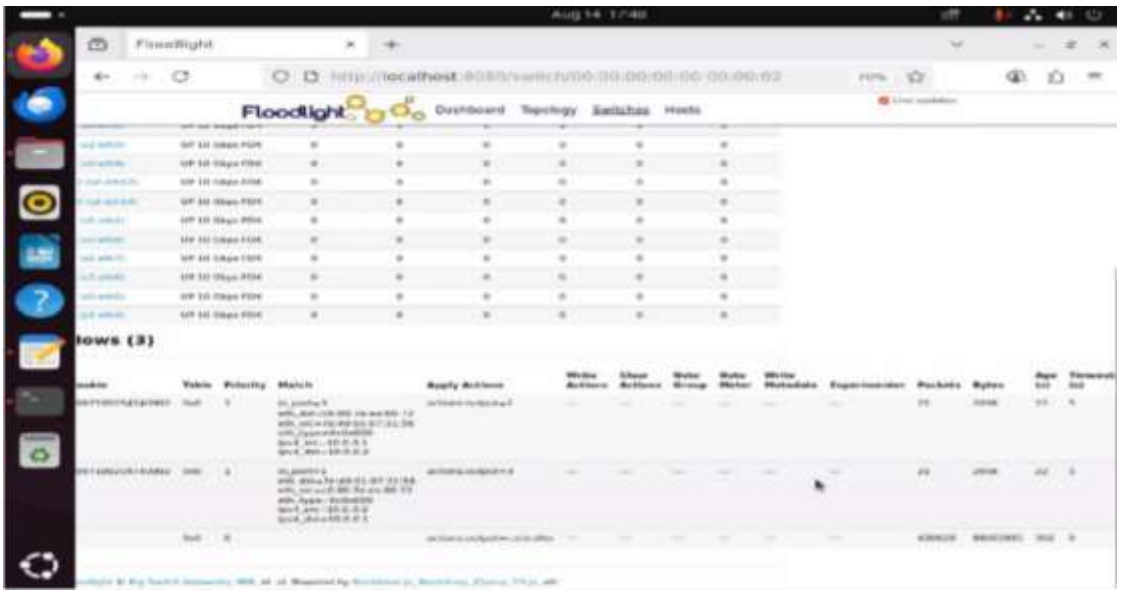
**Malla**

**Figura 15. Floodlight topología malla**


*Fuente*: Elaboración propia

En la **Figura 15** se visualiza la topología árbol diseñada en el controlador Floodlight, incluyendo hosts.

**Figura 16. Visualización de estadística en malla**



*Fuente*: Elaboración propia

Se visualiza las estadísticas del switch al realizar el comando ping en la topología malla.

**Tabla 4. Prueba de rendimiento en Herramienta Floodlight**

| Floodlight | | | | | |
|---|---|---|---|---|---|
| Infraestructura | Topología | Enviados | Recibidos | Throughput | Perdida de paquetes |
| 1 | Árbol | 171 | 169 | 169/171 | 0,011695906 |
| 2 | Malla | 21 | 21 | 21/21 | 0 |
| 3 | Estrella | 2163 | 2162 | 2162/2163 | 0,000462321 |

Nota: Elaboración propia

La **Tabla 4** muestra la comparación de rendimiento de las topologías en el controlador Floodlight, donde se evidencia que malla no genera perdida y su rendimiento es estable en comparación a las topologías árbol y estrella siendo árbol la menos destacable.

**Tabla 5. Perdida de paquetes en bytes**

| Perdidas de bytes | Enviados | Recibidos | Topología |
|---|---|---|---|
| 0,14160401 | 16758 | 14385 | Árbol |
| 0 | 2058 | 2058 | Malla |
| 0,001273557 | 139766 | 139588 | Estrella |

Nota: Elaboración propia

**Tabla 5** refleja la perdida de paquete en bytes de las topolgias.

**Tabla 6. Latencia en envió de paquetes**

| | h1-h2 | |
|---|---|---|
| Latencia | Numero de paquete enviados | Topología |
| 11272 | 12 | Árbol |
| 11257 | 12 | Malla |
| 11273 | 12 | Estrella |

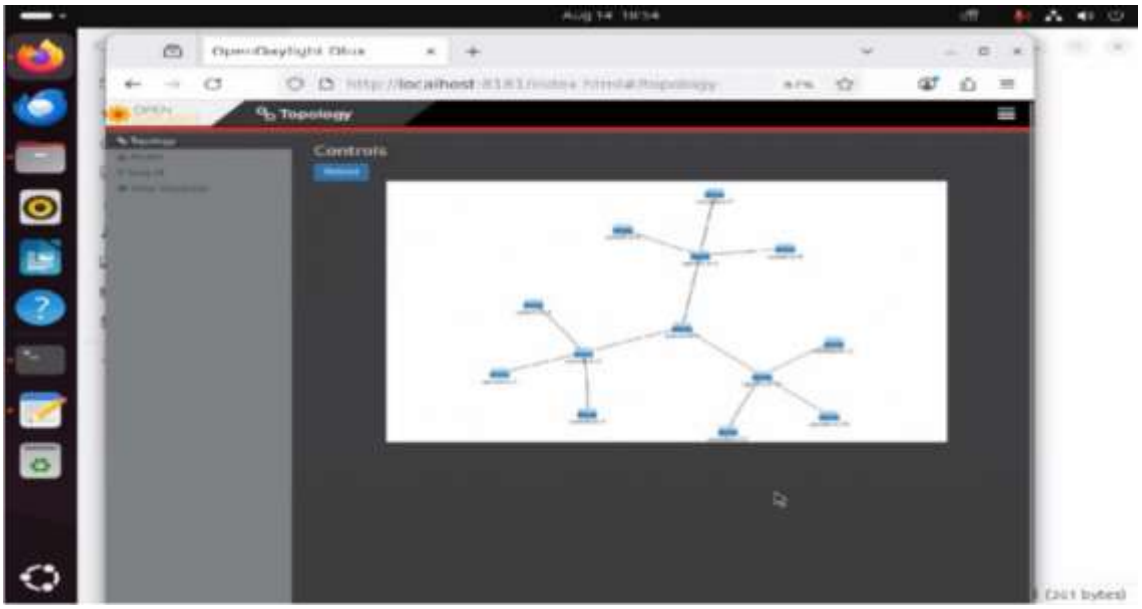
Nota: Elaboración propia

**Tabla 6** compara la latencia de las topologías en el controlador Floodlight, una vez más malla se destaca con su menor latencia con respecto a las otras, aunque, las diferencias entre árbol y estrella es mínima.

## OpenDaylight

### Árbol

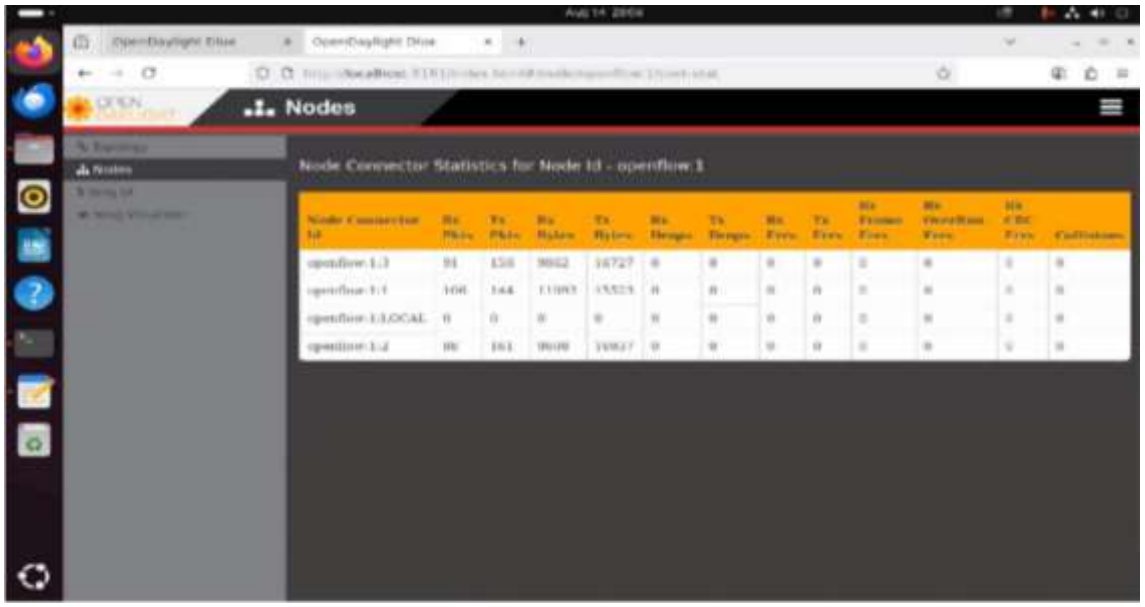
**Figura 17. OpenDaylight topología árbol**




*Fuente*: Elaboración propia

**Figura 17** representa la topología árbol en el controlador OpenDaylight, como es el caso de ONOS no muestra los hosts hasta que se realiza el comando ping o pingall como se presenta en figuras posteriores.

**Figura 18. Visualización de estadística en árbol**

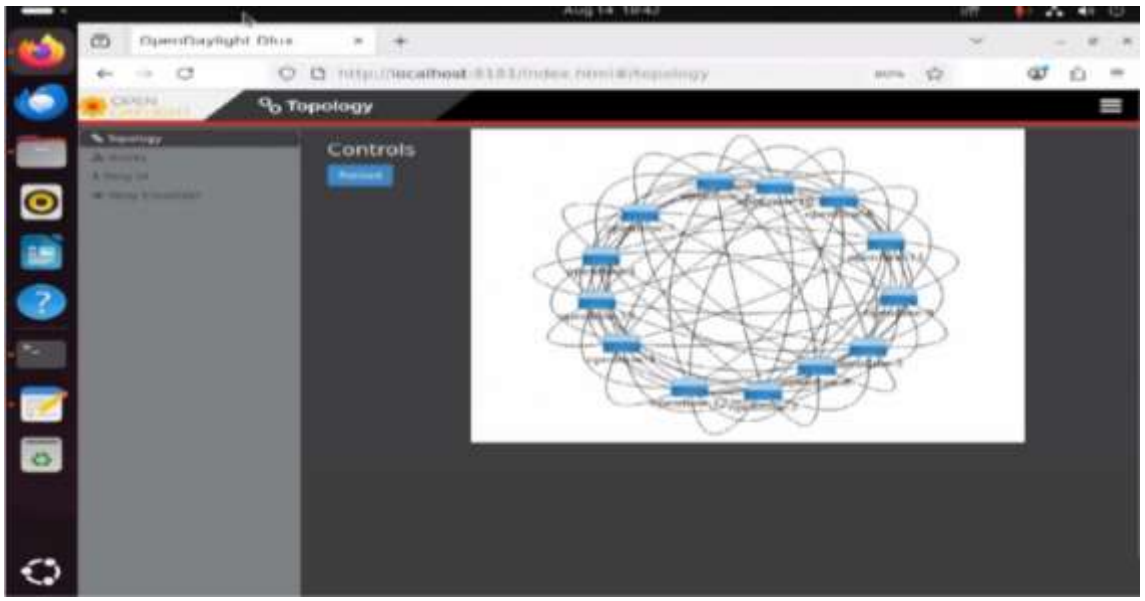



*Fuente*: Elaboración propia

Se visualiza en **Figura 18** las estadísticas del nodo que conecta los hosts 1 y 2 en la topología árbol.

**Malla**

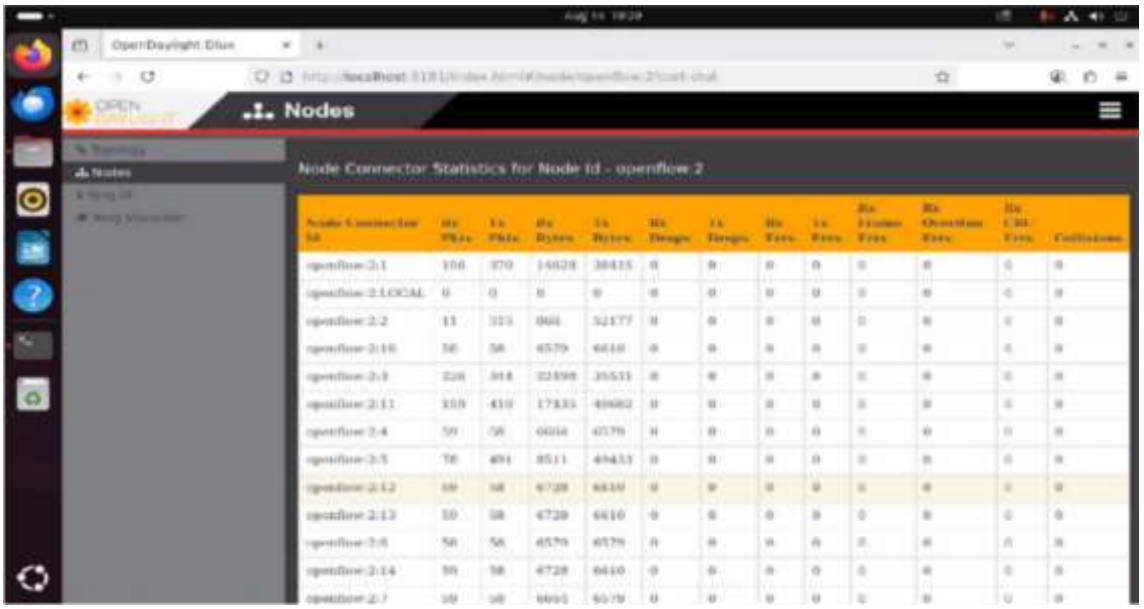
**Figura 19. OpenDaylight topología malla**




*Fuente*: Elaboración propia

**Figura 19** presenta la topología malla en el controlador OpenDaylight.

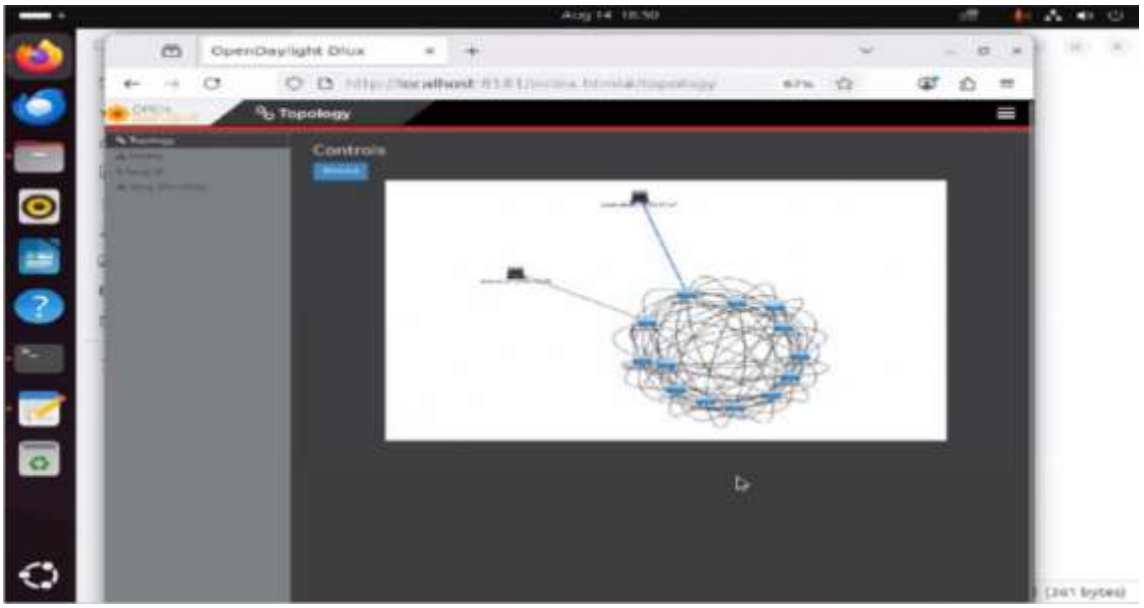
**Figura 20. Visualización de estadística en malla**




*Fuente*: Elaboración propia

Se observa en **Figura 20** las estadísticas del conector de nodo que conecta a los nodos de los cuales están conectados los hosts 1 y 2.

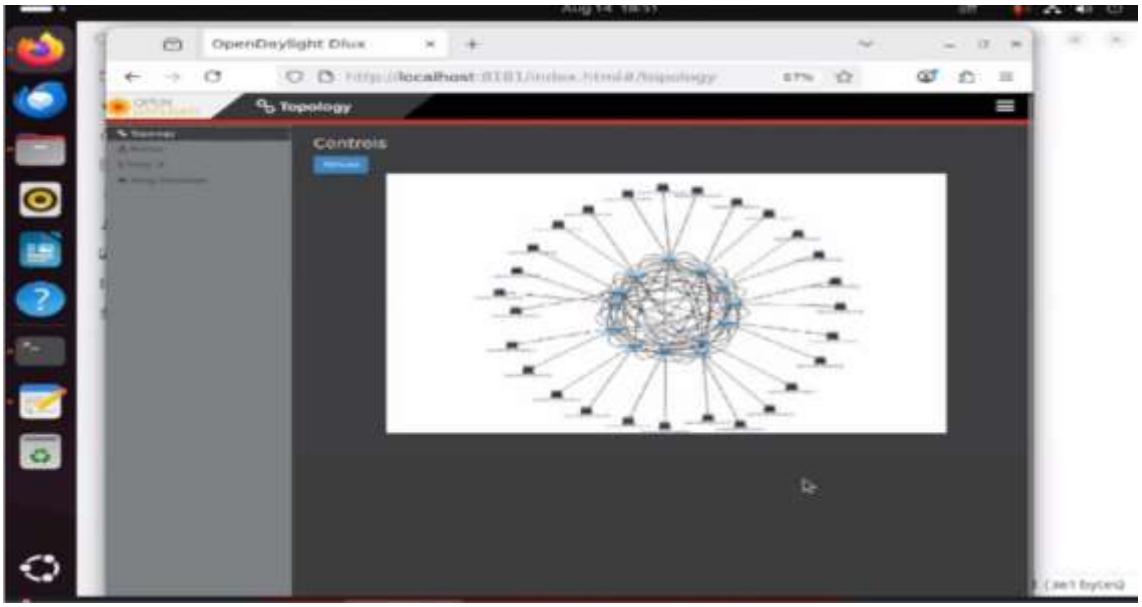
**Figura 21. Visualización al realizar un ping desde dispositivos h1 a h2**




*Fuente*: Elaboración propia

Como se presenta en esta **Figura 21** al realizar ping entre host 1 y host 2 en la visualización de la topología aparecen las ilustraciones que representas a dichos hosts.

**Figura 22. Visualización al realizar un pingall**

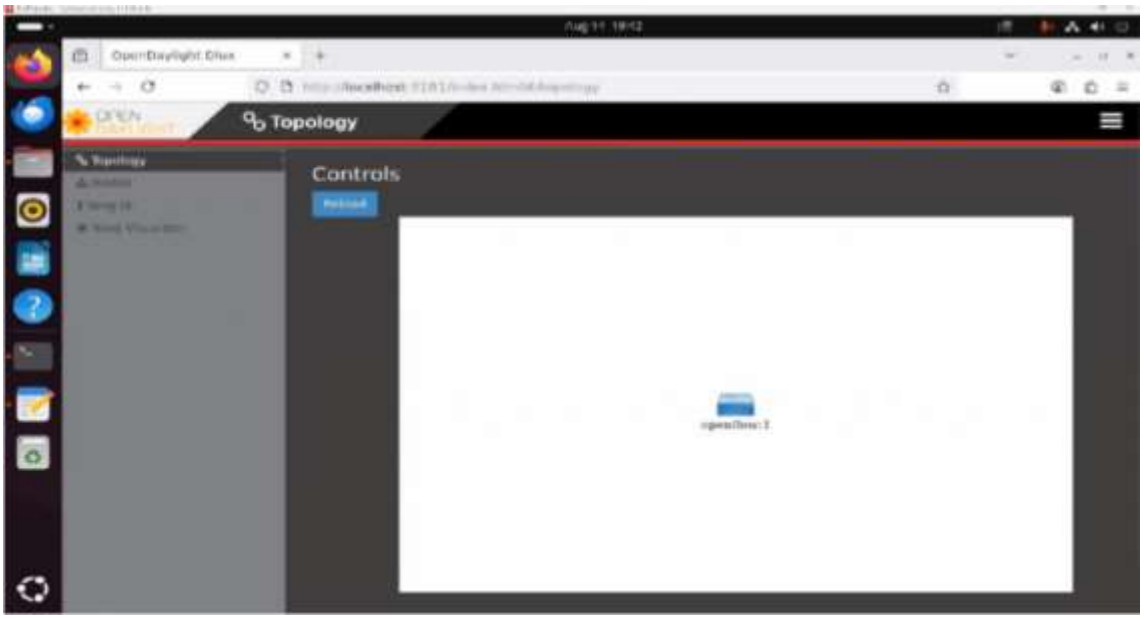



*Fuente*: Elaboración propia

**Figura 22** al realizar pingall, en la visualización de la topología aparecen las ilustraciones que representas a todos los hosts.

**Estrella**

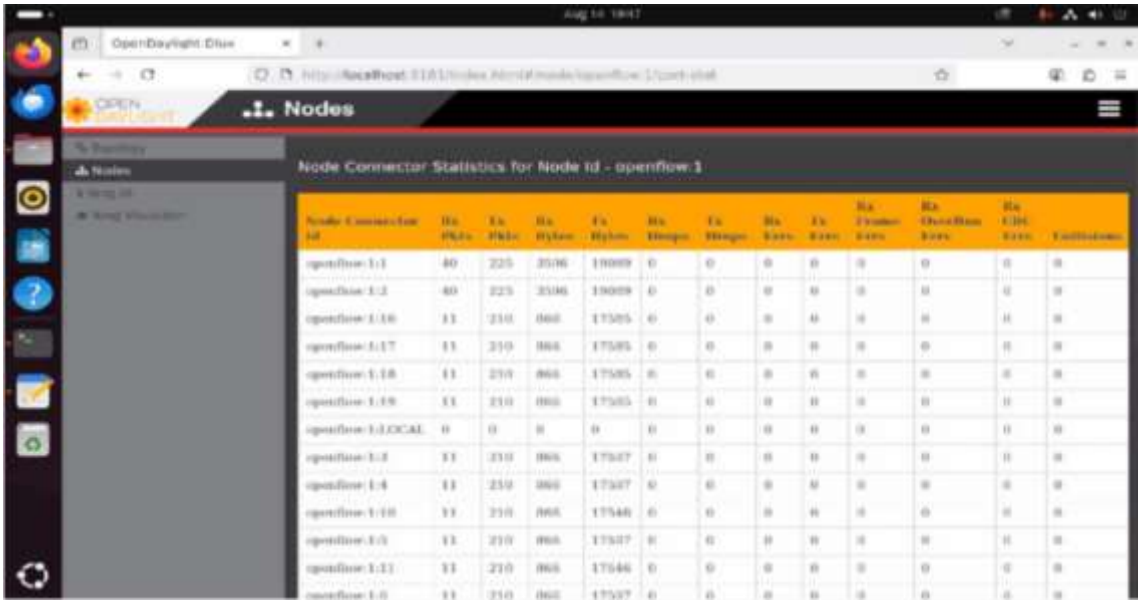
**Figura 23. OpenDaylight topología estrella**




*Fuente*: Elaboración propia

**Figura 23** refleja la topología estrella, aunque solo refleje un solo nodo, como se mencionó antes al realizar un comando pingall se visualizaran todos los hosts.

**Figura 24. Visualización de estadística en estrella**




*Fuente*: Elaboración propia

Se visualiza en **Figura 24** las estadísticas del conector de nodo que conecta a los nodos de los cuales están conectados los hosts 1 y 2.

**Tabla 7. Prueba de rendimiento en Herramienta OpenDaylight**

| OpenDaylight | | | | | |
|---|---|---|---|---|---|
| Infraestructura | Topología | Enviados | Recibidos | Throughput | Perdida de paquetes |
| 1 | Árbol | 91 | 91 | 91/91 | 0 |
| 2 | Malla | 151 | 148 | 21/21 | 0,01986755 |
| 3 | Estrella | 40 | 40 | 40/40 | 0 |

Nota: Elaboración propia

**Tabla 7** muestra la comparación de rendimiento de las topologías en el controlador OpenDaylight, demostrando que en este caso malla fue la única topología que presento perdidas en comparación a estrella y árbol.

**Tabla 8. perdida de paquetes en bytes**

| Perdidas de bytes | Enviados | Recibidos | Topología |
|---|---|---|---|
| 0 | 8454 | 8454 | Árbol |
| 0,020877716 | 14082 | 13788 | Malla |
| 0 | 3596 | 3596 | Estrella |

Nota: Elaboración propia

Se refleja en **Tabla 8** la perdida en bytes de las topologías.

**Tabla 9. Latencia en envió de paquetes**

| | h1-h2 | |
|---|---|---|
| Latencia | Numero de paquete enviados | Topología |
| 11273 | 12 | Árbol |
| 11273 | 12 | Malla |
| 11269 | 12 | Estrella |

Nota: Elaboración propia

Como se observa en **Tabla 9** quien presenta la menor latencia es la topología estrella en comparación con árbol y malla, aunque no es demasiada la diferencia, además árbol y estrella demuestran una latencia igual.

## 4.2 Discusión

El análisis de los resultados obtenidos en las simulaciones con Mininet permite identificar tendencias claras respecto al desempeño de los controladores SDN en diferentes topologías. En términos de rendimiento (throughput), se evidenció que ONOS alcanzó los valores más altos en las topologías árbol y malla, lo que concuerda con estudios previos como los de Guerrero y Cando (2023), quienes señalaron la robustez de ONOS en escenarios con mayor tráfico de datos. Sin embargo, a diferencia de lo reportado por Lasso Guamán y Puchaicela (2021), donde Floodlight se destacó en la transferencia masiva de datos, en esta investigación Floodlight presentó un desempeño menos favorable en comparación con ONOS y OpenDaylight, aunque se mantuvo como una alternativa confiable en la topología de malla.

Respecto a la latencia, los tres controladores obtuvieron valores similares, lo que indica que esta métrica no constituye un factor diferenciador significativo bajo las condiciones del experimento. Este hallazgo coincide con lo planteado por (Haggag et al., 2022), quienes destacan que la latencia en entornos simulados se mantiene estable independientemente del controlador, siendo más sensible a las condiciones del hardware físico.

El aporte más relevante de esta investigación radica en que, a diferencia de prácticas previas realizadas de manera improvisada en laboratorios universitarios, el presente trabajo propone un diseño metodológico estructurado y replicable para la evaluación de redes SDN. Este enfoque reduce la brecha entre teoría y práctica, ya que genera métricas claras y repetibles que pueden ser aplicadas por estudiantes e investigadores en diferentes entornos académicos. En este sentido, el estudio no solo valida la utilidad de Mininet como herramienta de simulación, sino que también deja

una base metodológica que fortalece la enseñanza de redes avanzadas en contextos donde no se dispone de infraestructura física especializada.

# CAPÍTULO V - CONCLUSIONES Y RECOMENDACIONES

## 5.1 Conclusiones

El diseño de una infraestructura SDN en un entorno simulado, utilizando OpenFlow y Mininet, permitió obtener mediciones precisas y repetibles de métricas de rendimiento, lo que confirma la viabilidad de este enfoque como solución académica ante la falta de equipos físicos especializados.

ONOS demostró un desempeño superior en términos de rendimiento en las topologías árbol y malla, mientras que OpenDaylight destacó por su confiabilidad y Floodlight se mantuvo como una opción viable en escenarios de baja complejidad. Esto evidencia que no existe un único controlador óptimo, sino que la selección debe responder al tipo de topología y necesidades específicas del entorno.

Los resultados alcanzados contribuyen a reducir la brecha entre el conocimiento teórico y la experiencia práctica en redes SDN, fortaleciendo el proceso formativo de estudiantes en instituciones con recursos limitados.

El principal aporte de esta investigación es la elaboración de un modelo metodológico replicable para evaluar el desempeño de redes SDN en entornos simulados, lo cual puede ser aplicado en otras universidades como guía de enseñanza y experimentación.

### 5.2 Recomendaciones

Implementar el modelo metodológico diseñado en este trabajo como una herramienta estandarizada de enseñanza en universidades y centros de formación que carezcan de equipos físicos para prácticas en SDN.

Ampliar el alcance de futuras investigaciones incorporando otros controladores como POX y Ryu, con el fin de enriquecer la comparación de resultados en diferentes escenarios de red.

Realizar pruebas con tráfico que represente aplicaciones reales, como servicios de streaming o VoIP, para validar la aplicabilidad del diseño en condiciones más cercanas al uso profesional.

Incluir en próximos estudios métricas adicionales de seguridad y escalabilidad, con el objetivo de evaluar la resiliencia de las redes SDN ante fallos, ataques o incremento progresivo de nodos.

Automatizar las pruebas mediante scripts en Python, lo cual permitiría ejecutar simulaciones masivas de forma más eficiente, reduciendo la intervención manual y mejorando la confiabilidad de los resultados.

## REFERENCIAS

Addad, R. A., Dutra, D. L. C., Bagaa, M., Taleb, T., Flinck, H., & Namane, M. (2022). *Benchmarking the ONOS Intent interfaces to ease 5G service management* (Versión 1). arXiv. https://doi.org/10.48550/ARXIV.2201.01407

Agnew, D., Boamah, S., Mathieu, R., Cooper, A., McNair, J., & Bretas, A. (2022). *Distributed Software-Defined Network Architecture for Smart Grid Resilience to Denial-of-Service Attacks* (Versión 1). arXiv. https://doi.org/10.48550/ARXIV.2212.09990

Ahmad, A. A., Boukari, S., Bello, A. M., Bichi, A. M., Gimba, S., & Muhammad, M. A. (2025). *A Review on Design and Simulation of Multiple- Controller SDN Based Model in Mininet with SNORT Intrusion Detection System*. *13*(1).

Aslan, F., & Al-Somaidai, M. (2022). Performance Analysis of Various SDN Controllers with Different Network Size in SDWN. *Proceedings of 2nd International Multi-Disciplinary Conference Theme: Integrated Sciences and Technologies, IMDC-IST 2021, 7-9 September 2021, Sakarya, Turkey*. Proceedings of 2nd International Multi-Disciplinary Conference Theme: Integrated Sciences and Technologies, IMDC-IST 2021, 7-9 September 2021, Sakarya, Turkey, Sakarya, Turkey. https://doi.org/10.4108/eai.7-9-2021.2314875

Canedo, E., Lopes De Mendonça, F., Nze, G., Praciano, B., Pinheiro, G., & Sousa Jr., R. (2020). Performance Evaluation of Software Defined Network Controllers: *Proceedings of the 10th International Conference on Cloud Computing and Services Science*, 363-370. https://doi.org/10.5220/0009414303630370

Haggag, A., Awad, S., & Gaballah, A. S. (2022). Controllers Performance Analysis in Software-Defined Networking. *International Journal of Scientific Research in Computer Science and Engineering*, *10*(3), 47-51.

Haro, Q., & Fernando, J. (2021). *Software defined wide area networking (SD-WAN) como mecanismo de seguridad en accesos wan*. https://repositorio.puce.edu.ec/handle/123456789/8535

Khudhair, T., & Athab, O. (2025). Recent Tools of Software-Defined Networking Traffic Generation and Data Collection. *Al-Khwarizmi Engineering Journal*, *21*(2), 93-105. https://doi.org/10.22153/kej.2025.06.002

Kumar, C. L., Betam, S., Pustokhin, D., Laxmi Lydia, E., Bala, K., Aluvalu, R., & Panigrahi, B. S. (2025). Metaparameter optimized hybrid deep learning model for next generation cybersecurity in software defined networking environment. *Scientific Reports*, *15*(1), 14166. https://doi.org/10.1038/s41598-025-96153-w

Lasso Guamán, D. J., & Puchaicela Condoy, J. R. (2021). *Evaluación del rendimiento de un prototipo SDN (Software Defined Networking) bajo el protocolo OpenFlow utilizando herramientas Open Source en un entorno virtualizado* [bachelorThesis]. http://dspace.ups.edu.ec/handle/123456789/19861

Li, A., Padhye, R., & Sekar, V. (2022). *SPIDER: Fuzzing for Stateful Performance Issues in the ONOS Software-Defined Network Controller* (Versión 2). arXiv. https://doi.org/10.48550/ARXIV.2209.04026

López Huera, M. E. (2021). *Metodología de transición de arquitecturas convencionales a redes definidas por software en plataformas de código abierto con base en el estándar ETSI GS NFV-PER 001* [bachelorThesis]. https://repositorio.utn.edu.ec/handle/123456789/11636

Lopez-Gomez, F., Marin-Lopez, R., Canovas, O., Lopez-Millan, G., & Pereniguez-Garcia, F. (2025). SDN-AAA: Towards the standard management of AAA infrastructures. *Journal of Network and Computer Applications*, *236*, 104114. https://doi.org/10.1016/j.jnca.2025.104114

Medina Magües, S. S., & Tenezaca Mawyin, J. F. (2023). *Prototipado Programable y Simulación Virtual de Redes SDN*. http://www.dspace.espol.edu.ec/handle/123456789/58223

Merayo, N., De Pintos, D., Aguado, J. C., De Miguel, I., Durán, R. J., Fernández, P., Lorenzo, R. M., & Abril, E. J. (2021). Experimental validation of an SDN residential network management proposal over a GPON testbed. *Optical*

*Switching and Networking*, *42*, 100631. https://doi.org/10.1016/j.osn.2021.100631

Montazerolghaem, A., & Imanpour, S. (2025). *Evaluation and Performance Analysis of the Ryu Controller in Various Network Scenarios* (Versión 1). arXiv. https://doi.org/10.48550/ARXIV.2505.19290

Raca, D., Salian, M., & Zahran, A. H. (2022). Enabling scalable emulation of differentiated services in mininet. *Proceedings of the 13th ACM Multimedia Systems Conference*, 240-245. https://doi.org/10.1145/3524273.3532893

Rahim, J. A., Nordin, R., & Amodu, O. A. (2024). Open-Source Software Defined Networking Controllers: State-of-the-Art, Challenges and Solutions for Future Network Providers. *Computers, Materials & Continua*, *80*(1), 747-800. https://doi.org/10.32604/cmc.2024.047009

Rakissaga, W. A. O., Omar, H. H., & Kouraogo, P. J. (2025). Software Defined Networks: Strengths, Weaknesses, and Resilience to Failures. *Engineering*, *17*(01), 19-29. https://doi.org/10.4236/eng.2025.171002

Ruchel, L. V., Turchetti, R. C., & De Camargo, E. T. (2022a). Evaluation of the robustness of SDN controllers ONOS and ODL. *Computer Networks*, *219*, 109403. https://doi.org/10.1016/j.comnet.2022.109403

Ruchel, L. V., Turchetti, R. C., & De Camargo, E. T. (2022b). Evaluation of the robustness of SDN controllers ONOS and ODL. *Computer Networks*, *219*, 109403. https://doi.org/10.1016/j.comnet.2022.109403

Shafiq, S., Rahman, Md. S., Shaon, S. A., Mahmud, I., & Hosen, A. S. M. S. (2024). A Review on Software-Defined Networking for Internet of Things Inclusive of Distributed Computing, Blockchain, and Mobile Network Technology: Basics, Trends, Challenges, and Future Research Potentials. *International Journal of*

*Distributed Sensor Networks*, *2024*(1), 9006405. https://doi.org/10.1155/2024/9006405

Shamsoddini, M., Ghaffari, A., Kargar, M., & Derakhshanfard, N. (2025). RCPFH: Reliable controller placement in software-defined networks using fuzzy systems and a modified walrus optimization algorithm. *Simulation Modelling Practice and Theory*, *144*, 103171. https://doi.org/10.1016/j.simpat.2025.103171

Singh, A., Kaur, N., & Kaur, H. (2022). Extensive performance analysis of OpenDayLight (ODL) and Open Network Operating System (ONOS) SDN controllers. *Microprocessors and Microsystems*, *95*, 104715. https://doi.org/10.1016/j.micpro.2022.104715

Zhu, L., Karim, M. M., Sharif, K., Li, F., Du, X., & Guizani, M. (2019). *SDN Controllers: Benchmarking & Performance Evaluation* (Versión 1). arXiv. https://doi.org/10.48550/ARXIV.1902.04491