UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

INGENIERÍA EN ELECTRÓNICA Y REDES DE COMUNICACIÓN

IMPLEMENTACIÓN DE UNA RED HÍBRIDA SDN/MPLS PARA LA GESTIÓN DE SERVICIOS DIFERENCIADOS MEDIANTE SIMULACIÓN DE TIEMPO REAL GNS3

PRESENTADO POR:

JHONNY ALEXANDER LOZADA BENALCÁZAR

PARA OPTAR AL GRADO DE:

INGENIERO EN ELECTRÓNICA Y REDES DE COMUNICACIÓN

DOCENTE DIRECTOR:

ING. MAURICIO DOMÍNGUEZ

IBARRA, ECUADOR

2022

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1804880837		
APELLIDOS Y NOMBRES:	Jhonny Alexander Lozada Benalcázar		
DIRECCIÓN:	Av. Salinas 16-34 y Atahualpa - Atuntaqui		
EMAIL:	jalozadab1@utn.edu.ec		
TELÉFONO FIJO:	062910182	**TELÉFONO MÓVIL:**	0939192136

DATOS DE LA OBRA	
TÍTULO:	Implementación de una red híbrida SDN/MPLS para la gestión de servicios diferenciados mediante simulación de tiempo real GNS3
AUTOR (ES):	Jhonny Alexander Lozada Benalcázar
FECHA: DD/MM/AAAA	26/10/2022
SOLO PARA TRABAJOS DE GRADO	
PROGRAMA:	[X] **PREGRADO** [ ] **POSGRADO**
TITULO POR EL QUE OPTA:	Ingeniería en Electrónica y Redes de Comunicación
ASESOR /DIRECTOR:	Ing. Hernán Mauricio Domínguez, MSc.

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

CONSTANCIAS

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 26 días del mes de octubre de 2022

EL AUTOR:

Jhonny Alexander Lozada Benalcázar
C.C. 1804880837

UNIVERSIDAD TÉCNICA DEL NORTE

BIBLIOTECA UNIVERSITARIA

CERTIFICACIÓN

Ing. Hernán Mauricio Domínguez, MSc., director del presente Trabajo de Titulación certifica:

Que, el presente trabajo de titulación "Implementación de una red híbrida SDN/MPLS para la gestión de servicios diferenciados mediante simulación de tiempo real GNS3", fue realizado en su totalidad por el Sr. Jhonny Alexander Lozada Benalcázar, bajo mi supervisión.

Es todo en cuanto puedo certificar en honor a la verdad.

Ing. Hernán Mauricio Domínguez Limaico
DIRECTOR

DEDICATORIA

Dedico esta tesis,

A Dios que me ha dado la salud, fuerza y me ha guiado en este camino para culminar mis estudios.

A mis padres, quienes han sido los que me han apoyado con su amor, trabajo y sacrificio durante estos años de estudio, además que han sido los que han soportado mi carácter en los días más difíciles, confiando en mis capacidades para culminar esta etapa.

A mi abuelita, que se ha preocupado por mí y me ha entregado su cariño incondicional.

A mis doctores, Vannesa Sarzosa y Luis Rojas, que me han tratado con todo su cariño, amor y profesionalidad, siendo fundamentales para controlar mi salud y mi bienestar emocional y psicológico.

A mis amigos, que me han acompañado en las noches largas y me han sabido escuchar cuando más lo necesitaba, siendo un apoyo incondicional.

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

AGRADECIMIENTO

Agradezco,

A mis padres, por ser los que me han apoyado para cumplir todas mis metas. Para ellos mi amor y gratitud infinita.

A la familia Torres Olalla, por tratarme como un hijo más desde muy pequeño y por haberme dado la motivación para cursar esta carrera. De igual manera a la familia Cruz Vivas, por haber confiado en mí, dándome su fuerza y apoyo sincero cuando me sentía agobiado con los estudios.

A mi tutor el Ing. Mauricio Domínguez, que ha sido mi guía y me ha dado los conocimientos necesarios para realizar este trabajo investigativo; de igual manera a mi asesor el Ing. Edgar Maya que desde un principio me ha impartido sus mejores consejos y valores.

Y para finalizar, un agradecimiento especial a todos los docentes que me ayudaron a formarme como profesional y a todos con quien he compartido tiempo en las aulas de clase.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO I
ANTECEDENTES

Tema

IMPLEMENTACIÓN DE UNA RED HÍBRIDA SDN/MPLS PARA LA GESTIÓN DE SERVICIOS DIFERENCIADOS MEDIANTE SIMULACIÓN DE TIEMPO REAL GNS3

Problema

Con el aumento de nuevos dispositivos y aplicaciones, la última década ha sido testigo de un crecimiento exponencial del volumen de tráfico en las redes de comunicación (Experimen, 2017), por otra parte, dadas las situaciones de pandemia del COVID19, el uso de servicios de streaming, VoIP y WEB (Bahnasse et al., 2018) demanda de grandes recursos donde la calidad de servicio (QoS) se vuelve un tema muy crítico. Según (Servicio, 2016) la QoS se define como un conjunto de tecnologías que permiten a los administradores de red manejar los efectos de la congestión del tráfico usando óptimamente los diferentes recursos de la red, en lugar de ir aumentando continuamente capacidad, sino distribuirlo de acuerdo a las necesidades de la empresa.

La gran mayoría de empresas en el Ecuador utilizan redes con el protocolo MPLS (Sonny Eli Zaluchu, 2021), pero tal y como lo menciona en su caso de estudio muchos de los routers con los que se operan ya han cumplido su vida útil o se encuentran obsoletos, también uno de los problemas que se llegan a presentar es el limitado crecimiento de capacidades de transporte dando como resultado que este tipo de redes ya no pueden soportan la demanda del tráfico actual. Dada su infraestructura y administración cerrada las redes MPLS no permiten los cambios dinámicos (Bahnasse et al., 2018) ocasionando que no se puedan alinear a las necesidades de negocio por su retardo en el despliegue, mala gestión de los recursos e interrupciones en su servicio (Sinha et al., 2017).

El paradigma denominado SDN junto con el protocolo OpenFlow proporciona muchas funciones nuevas para gestión del tráfico (Xie et al., 2019), entre muchos de los beneficios que proporciona OpenFlow está su capacidad de enrutar y redireccionar el tráfico según el patrón de la red, por lo tanto nos abre un área muy amplia para la investigación sobre ingeniería de tráfico (Wiley, 2020), pero no todo lo que puede aportar SDN se encuentra en OpenFlow, sino

que al permitirnos separar el plano de control del plano de datos nos permite mayor flexibilidad para la expansión de las redes. En este sentido, uno de los puntos importantes de las redes SDN se encuentra en plano de control, al cual se le puede definir como el cerebro de la arquitectura (Wu et al., 2020), esta es la plataforma de código abierto para SDN OpenDayLight, la cual se encarga de proporcionar el control programático centralizado, así como la supervisión de los dispositivos de red (Badotra & Singh, 2017) y su compatibilidad con OpenFlow , implementando funcionalidades como el descubrimiento de la topología, descubrimiento de rutas diseñadas, asignación de recursos de manera eficiente, entre otras.

Por otra parte, la arquitectura horizontal que nos brinda las redes SDN fomenta la interoperabilidad y potencia la innovación, permitiendo que tan pronto la red comience a operar el controlador que tiene una vista general de la topología y una mejor capacidad de programación es capaz de superar las formas de descubrimiento de rutas de MPLS (Vissicchio et al., 2014). Con esto es muy claro que si se realiza una red híbrida SDN/MPLS utilizando las ventajas que nos llegan a ofrecer como; la reutilización de infraestructura MPLS existente, menor cantidad de interrupciones en los servicios, mejor asignación de los recursos y reducción de costes en equipos. En Ecuador la investigación en esta área es prácticamente una incógnita (Sinha et al., 2017), tomando en cuenta de que según las últimas estadísticas únicamente la transmisión de video por IP ocupa el 82% del tráfico IP total (Servicio, 2016), si a esto le sumamos el tráfico que se genera con VoIP y WEB sin duda alguna que el escenario establecido a analizar es muy prometedor.

Alcance

En el anteproyecto de grado se pretende analizar en un ambiente de simulación (GNS3) una red SDN usando el protocolo MPLS para Ingeniería de Tráfico Inteligente y la gestión de servicios diferenciados, con el fin de llegar a demostrar la flexibilidad y el comportamiento de la red de manera dinámica, obteniendo así una red segura y confiable.

Con el fin de cumplir el objetivo planteado previamente se establecerá el estado del arte acorde a la tecnología SDN y el protocolo MPLS, donde se contemplará temáticas como: arquitectura, protocolo OpenFlow, controlador OpenDayLight, manejo de etiquetas MPLS, marcaje de paquetes y modos de operación, con el fin establecer el estado de la situación actual de la tecnología.

Para la parte del diseño se trabajará en una plataforma abierta sobre la cual se pueda probar la arquitectura y el despliegue a escala de la tecnología SDN, se incluirá los requerimientos de software para poder trabajar en un ambiente controlado haciendo uso de herramientas OpenSource como OpenFlow y OpenDayLight, las cuales se encargarán de la configuración, operación y funcionamiento de la red, permitiendo explotar todas las funcionalidades que llegan a ofrecer en el manejo y gestión de las etiquetas MPLS. En cuanto a simulación se hará uso de la herramienta de GNS3 encargado de la interoperabilidad entre SDN y MPLS para gestionar los servicios de streaming, VoIP y WEB.

Además, para lograr la implementación de la red híbrida SDN/MPLS, se empezará con el planteamiento de una topología que pueda soportar tantos los servidores que se encargarán de los servicios de la red, como los diferentes protocolos y tecnologías que serán parte fundamental para el desarrollo del proyecto. Las pruebas que serán desarrolladas con la red planteada son: separación del plano de control y el plano de datos, planteamiento de la mejor opción de configuración del controlador OpenDayLight para Ingeniería de Tráfico Inteligente y DiffServ.

Tomando en cuenta que se desea realizar un análisis comparativo entre una red híbrida SDN/MPLS y una red MPLS tradicional, una vez establecida la topología, la convergencia y la conectividad entre las diferentes plataformas, ya sería posible proponer una infraestructura a escala para orquestar los servicios streaming, VoIP y WEB, la cual será establecida en GNS3 y que se va a definir a lo largo de la investigación.

El trabajo de grado finalizará con el análisis de las pruebas de funcionamiento, donde se plantea el establecimiento de la mejor configuración y diseño de red junto con plataformas de código abierto OpenFlow y OpenDayLight para la parte de SDN junto con GNS3 encargado de la parte de simulación y operación de la red, esto permitirá establecer la Ingeniería de Tráfico Inteligente y el manejo óptimo de los servicios diferenciados, dando a conocer las virtudes que proporciona una red híbrida SDN/MPLS como su flexibilidad, administración inteligente, impacto en el jitter en comunicaciones de VoIP y mínima latencia en la transmisión y recepción de datos.

La metodología a desarrollar el proyecto será en una primera parte analítica para investigar acerca de la tecnología y así poner en práctica lo planteado. Como segunda parte se utilizará una metodología deductiva por lo que se ha planteado su realización en cuatro etapas.

Primera etapa: Recopilación y análisis del estado del arte de la tecnología SDN, los protocolos OpenFlow, MPLS y el controlador OpenDayLight y para cumplir con los objetivos planteados.

Segunda etapa: Analizar el funcionamiento del protocolo OpenFlow y el controlador OpenDayLight para lograr implementarlos en una topología mediante GNS3 y así crear un entorno de red controlado.

Tercera etapa: Con la topología establecida, proceder a establecer la convergencia y conectividad de toda la red que permita analizar el funcionamiento de una red híbrida SDN/MPLS junto con los servicios de streaming, VoIP y WEB.

Cuarta etapa: Verificar el correcto funcionamiento de los servicios de la red mediante pruebas que permitan demostrar el aporte que puede llegar a brindar la tecnología SDN.

Justificación

Para el funcionamiento óptimo de una empresa u organización se busca realizar una red híbrida SDN - MPLS para el manejo óptimo de servicios de VoIP, streaming y WEB (Bermúdez, Cristian; Rodríguez, Jesús; Dussan, 2020), ya que se tendrá beneficios como un manejo eficiente de los recursos de la red, despliegue centralizado de la red, reducción de costes, manejo dinámico y la mejor QoS para el usuario.

Debido a que los proveedores de servicios de todo el mundo tienen grandes inversiones en infraestructuras de red MPLS altamente sofisticadas y ricas en funciones para brindar servicios a sus clientes (Sinha et al., 2017), estas se construyen sobre equipos de red tradicionales (Andersson & Brohne, 2017), es por esto que se deben sortear estos desafíos y se propone una arquitectura de ingeniería de tráfico para adoptar la integración de SDN y MPLS para que llegase a ser implementada en una red existente (Tajiki et al., 2019).

A nivel internacional existen diversos papers como (Sinha et al., 2017), (Pryslupskyi et al., 2019; Vissicchio et al., 2014), que discuten acerca de las ventajas que aportan la integración de SDN y MPLS, por lo que su estudio, el análisis de las ventajas que se pueden llegar a obtener y el impacto que se llegue a dar con su implementación en la ingeniería de tráfico inteligente y la diferenciación de los servicios. Por otra parte, a diferencia de lo que se establece no hay una investigación que abarque los servicios de streaming, VoIP y WEB en un mismo escenario, por lo que con el desarrollo de esta integración en una infraestructura a escala por medio de GNS3 se realizará diferentes pruebas de funcionamiento y así determinar el

impacto que se puede llegar a obtener en cuanto a manejo y gestión eficiente de los recursos de la red.

Adicionalmente en Ecuador existen algunas implementaciones de las redes SDN en diferentes ámbitos, pero en cuanto a una integración con MPLS es aún un ámbito que no se ha llegado a explotar en su totalidad, brindándonos de esta manera un campo donde los resultados que se puedan llegar a obtener pueden ser muy prometedores para dar a entender a los proveedores de servicios todos los beneficios que pueden poseer con una gestión dinámica de la red.

Objetivos

General

Desarrollar una red híbrida SDN/MPLS que permita el análisis de su comportamiento en comparación a una red MPLS tradicional, generando una topología que evidencie el impacto sobre los servicios diferenciados.

Específicos

Constituir el estado del arte de la tecnología SDN y el protocolo MPLS para diseñar y gestionar servicios diferenciados con ingeniería de tráfico inteligente.

Diseñar una topología de red en el software de simulación de GNS3 empleando equipos que soporten el protocolo OpenFlow y OpenDayLight para el despliegue de los servicios de streaming, VoIP y WEB.

Implementar SDN y MPLS para demostrar la interoperabilidad con la red de servicios mediante la topología planteada para demostrar la gestión inteligente de los servicios diferenciados.

Plantear las pruebas de funcionamiento y analizar los resultados obtenidos de calidad de servicio antes y después de implementar la red híbrida.

CAPÍTULO II
FUNDAMENTACIÓN TEÓRICA

En el presente capítulo se plantea el estudio bibliográfico acerca de los antecedentes investigativos que se tiene sobre la temática a tratar en este proyecto en el Ecuador, además se describirá el protocolo MPLS (MultiProtocol Label Switching), tipos de routers y manejo de etiquetas dentro de una red. Para la arquitectura de una Red Definida por Software (SDN) con el fin de realizar una red híbrida se analizará cada uno de sus planos, protocolos de operación y tipos de paquetes para finalmente dar a entender las ventajas de SDN junto a MPLS para servicios diferenciados.

2.1. MPLS

De acuerdo con (*rfc3031*, 2001) la conmutación de etiquetas multiprotocolo (MPLS) es un protocolo implementado principalmente por proveedores de servicios de Internet para mejorar el rendimiento de su red IP, en donde (Bahnasse et al., 2018), menciona que la principal motivación para utilizar el protocolo MPLS fue la velocidad del proceso de enrutamiento, con el principio de conmutación de etiquetas, MPLS proporciona un tratamiento rápido y simple sobre la retransmisión de paquetes IP.

De manera tradicional, como lo menciona (Bahnasse et al., 2018), en una red IP cada enrutador consulta el encabezado IP del paquete para determinar el destino, luego busca en la Base de información de enrutamiento (RIB) para determinar la dirección del siguiente salto o la interfaz saliente, a partir de entonces, se genera una solicitud de Protocolo de resolución de direcciones (ARP) para determinar la dirección física del siguiente salto si el medio es Ethernet.

Este procesamiento si se lo analiza probablemente sea muy costoso en términos de demora y dadas las correspondientes actualizaciones en el “RFC 3031”, para superar este límite, los fabricantes han incluido nuevos métodos de enrutamiento como la Base de información de reenvío (FIB) que consiste en crear tablas de enrutamiento y adyacencia.

La primera tabla se utiliza para almacenar, de manera ordenada, las resoluciones realizadas por la tabla RIB, y la segunda tabla se utiliza para insertar la información relacionada con el encapsulado vinculado a la capa de enlace de datos (*rfc3031*, 2001).

Los dispositivos que participan en los mecanismos del protocolo MPLS según (Alarcon Aquino, 2008), pueden ser clasificados en “ruteadores de etiqueta de borde o label edge routers (LERs)”, y en “ruteadores de conmutación de etiquetas o label switching routers (LSRs)”.

a. Label Edge Router (LER)

Un LER es un dispositivo que opera en el borde de una red de acceso hacia una red MPLS. Un LER soporta múltiples puertos conectados a diferentes tipos de redes (frame relay, ATM, y Ethernet); y se encarga, en el ingreso de establecer una LSP para el tráfico en uso y de enviar este tráfico hacia la red MPLS, usando el protocolo de señalización de etiquetas, y en el egreso de distribuir de nuevo el tráfico hacia la red de acceso que corresponda. El LER juega un papel muy importante en la asignación y remoción de etiquetas que se aplica al tráfico que entra y sale de una red MPLS.

b. Label Switching Routers (LSRs).

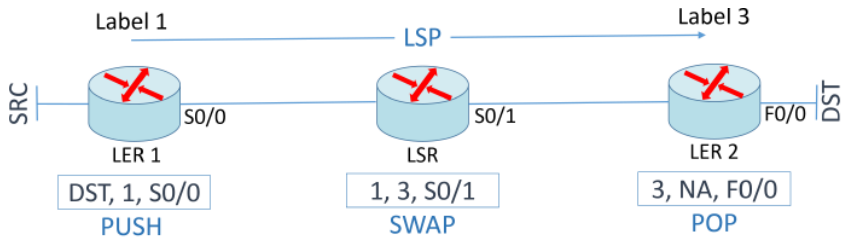
Un LSR es un dispositivo ruteador de alta velocidad, que dentro del núcleo de una red MPLS, participa en el establecimiento de las LSPs, usando el protocolo de señalización apropiado y una conmutación de alta velocidad aplicado al tráfico de datos, que se basa en las trayectorias establecidas.

c. Funcionamiento

Para comprender la manera en la que opera el protocolo MPLS, el funcionamiento en cómo opera MPLS se muestra a detalle en la Figura 1 y que será descrita a continuación: El dispositivo **LER 1** se encuentra en la frontera; realiza una operación PUSH a todos los paquetes IP que ingresan al dominio MPLS y es encargado de insertar la etiqueta 1 para llegar al destino (DST). Por otra parte, el **LSR** es un dispositivo intermedio que realiza una operación SWAP para consultar la tabla de etiquetas FIB, realizando el intercambio y envío de paquetes IP hacia el siguiente salto a través de la interfaz Serial 0/1. Finalmente, los paquetes IP llegan al dispositivo **LER 2,** el cual, es el encargado de realizar una operación POP para extraer las etiquetas que salen del dominio MPLS y enviar los paquetes IP hacia el destino.

Figura 1

Funcionamiento del protocolo MPLS



Fuente: (Bahnasse et al., 2018)

2.1.1. Tipos de LSP

(Alarcon Aquino, 2008) manifiesta que una LSP (Label Switched Path), es la secuencia de etiquetas en cada nodo a lo largo de su trayectoria, desde la fuente hasta el destino, las cuales, pueden ser establecidas previamente a la transmisión de datos (control-driven), o al momento en que se detecta un cierto flujo de datos (data-driven); para esto MPLS provee de dos opciones para establecer una LSP:

a) Ruteo salto a salto (hop-by-hop)

Cada LSR selecciona independientemente el siguiente salto (hop) para una FEC (Forwarding Equivalence Class) dada, esta metodología es similar a la que se usa en redes IP, en donde, el LSR usa cualquiera de los protocolos de ruteo disponibles.

b) Ruteo explícito

El LSR de ingreso especifica la lista de nodos por la cual viaja la trayectoria explícita, sin embargo, la ruta especificada puede ser no óptima. A lo largo de su trayectoria, es importante reservar recursos para asegurar calidad de servicio (QoS) al tráfico de datos, y así facilitar la ingeniería de tráfico.

2.1.2. Etiquetas

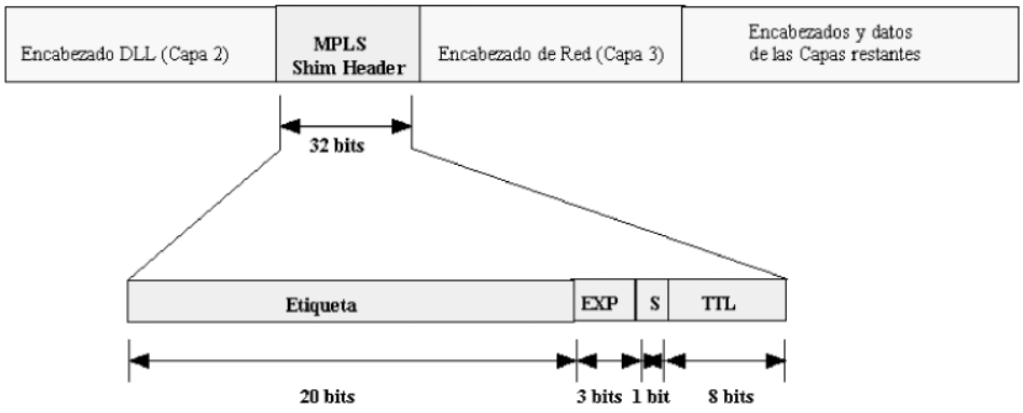
Una de las partes más importantes de MPLS son las etiquetas, estas en su forma más simple identifica la trayectoria que un paquete debe seguir, una etiqueta es acarreada o encapsulada dentro de un encabezado de Capa 2 junto con el paquete, donde el ruteador que recibe el paquete, examina el contenido de la etiqueta para determinar el siguiente hop (Alarcon Aquino, 2008).

Una vez que un paquete ha sido etiquetado, el resto del viaje del paquete a través de la red se basa en conmutación de etiquetas, el valor de una etiqueta es estrictamente de significancia local, es decir, que pertenecen únicamente a saltos entre LSRs (Alarcon Aquino, 2008).

La Figura 2 muestra el formato genérico de un encabezado MPLS o también llamado *shim header*, sus campos y como se interpone a los encabezados de las demás capas del modelo OSI. El campo de **etiqueta** se compone de 20 bits, estos son encargados de contener el valor de la etiqueta MPLS. El campo **EXP** se considera un campo experimental, contiene 3 bits y se toma en cuenta para consideraciones de QoS. El campo **S** que contiene 1 bit, nos sirve para indicar si está presente una pila de etiquetas (label stack) tomando el valor de 1, pero si la etiqueta es la única presente en la pila, entonces tomará el valor de 0. Finalmente, el campo **TTL** (Time to Live) de 8 bits, se usa para indicar el número de nodos MPLS por los que el paquete ha viajado hasta alcanzar su destino, su valor es copiado del encabezado del paquete cuando se ingresa a la LSP, y copiado nuevamente al encabezado del paquete IP cuando sale de la misma.

Figura 2

Formato genérico encabezado MPLS o shim header



Fuente: (Alarcon Aquino, 2008)

2.2. Redes Definidas por Software (SDN)

De acuerdo con (Wu et al., 2020) una SDN es una arquitectura de red emergente que consta de tres capas, los planos de aplicación, control y datos, la cual, es programable a través de una gestión lógicamente centralizada para simplificar las tareas de red complejas, como la optimización de rutas, la ingeniería de tráfico, etc., para una implementación de red cada vez más diversificada. Sin embargo (Open Networking Foundation (ONF), 2018) dice que una

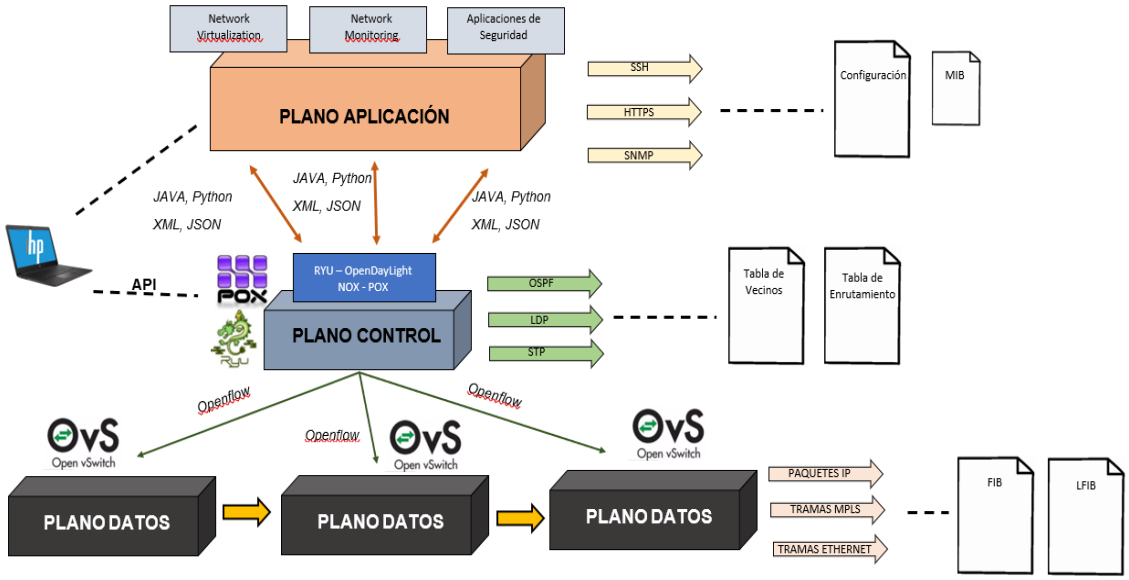
SDN es una arquitectura dinámica, administrable, rentable y adaptable, lo que la hace ideal para la naturaleza dinámica de gran ancho de banda de las aplicaciones actuales.

2.2.1. Arquitectura SDN

Como ya se definió en la sección 2.2, la arquitectura SDN consta de tres capas: Aplicación, control y aplicación, las cuales, se detallan en la Figura 3.

Figura 3

Arquitectura SDN



Según (Wiley, 2020), el plano de control SDN es necesario para descubrir una topología de red de toda la infraestructura SDN, principalmente para configurar rutas de transmisión de datos entre cualquier par de fuente a destino en el plano de datos, sin embargo, descubrir una topología de red es un desafío debido a la migración frecuente de las máquinas virtuales en el plano de datos, la falta de estándares de autenticación, etc.

La interfaz de programación de aplicaciones que reside entre los planos de control y aplicación se denomina interfaz en dirección norte (Sinha et al., 2017), donde se puede implementar un conjunto de servicios de red, como calidad de servicio (QoS), detección de intrusiones y funciones de monitoreo, la interfaz de comunicación entre los planos de control y de datos se denomina interfaz en dirección sur, donde el protocolo OpenFlow se utiliza comúnmente para intercambiar mensajes de control con dispositivos de reenvío, denominados switches OpenFlow (Bahnasse et al., 2018).

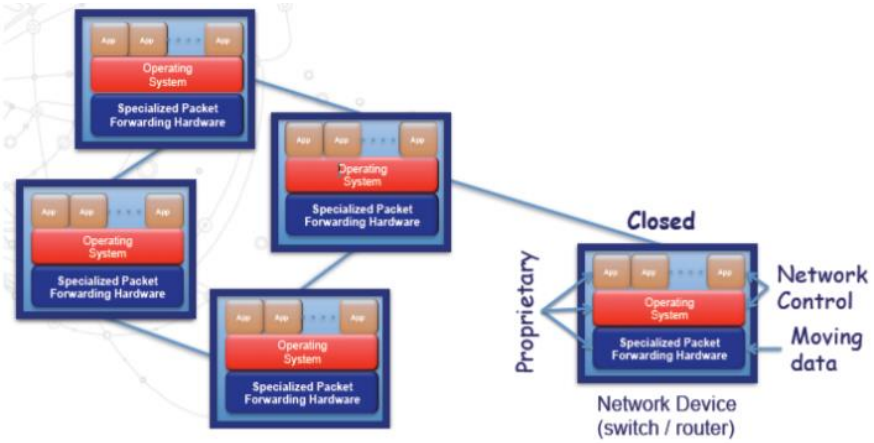
2.2.2. Comparación entre la red tradicional y la red SDN

(Lester et al., 2017) menciona, que las redes tradicionales, se encuentran basadas en IP, donde el gran problema que tienen es su difícil administración, esto se debe a que se encuentran integradas verticalmente, es decir, que cada uno de los dispositivos que están integrados en la red se controlan a sí mismos, con su propio firmware instalado en su espacio de memoria.

Por lo tanto, (Sinha et al., 2017) explica que para hacer cualquier tipo de cambio, por ejemplo, en su topología, reglas, protocolos, etc; el administrador de la red debe configurar manualmente cada uno de los dispositivos, esto aumenta la complejidad y el posible error, debido a todas estas causas, las arquitecturas de red están evolucionado hacia topologías dinámicas y programables, esto se lo representa en la Figura 4.

Figura 4

Arquitectura de red tradicional



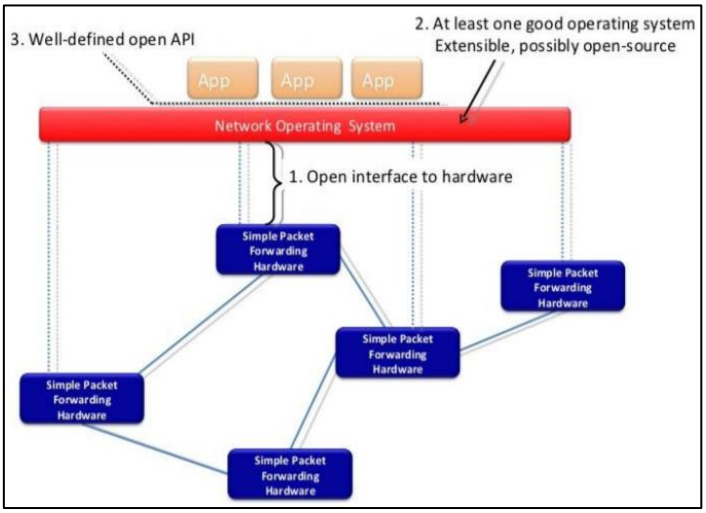
Fuente: (Sociedad et al., 2021)

La red SDN, es un tipo de arquitectura que tiene el objetivo de controlar el tráfico de datos de una manera centralizada, lo más característico es la eliminación del modelo de integración vertical, es decir, que existe una separación del plano de control y el plano de datos del core de la red (Sinha et al., 2017).

El administrador puede cambiar cualquier regla de los conmutadores de red cuando sea necesario dando o quitando prioridad, o hasta bloqueando tipos específicos de paquetes con un nivel de control muy detallado algo que no es posible en el entorno de red actual (Sinha et al., 2017), en la Figura 5 se puede visualizar este proceso.

Figura 5

Arquitectura red SDN



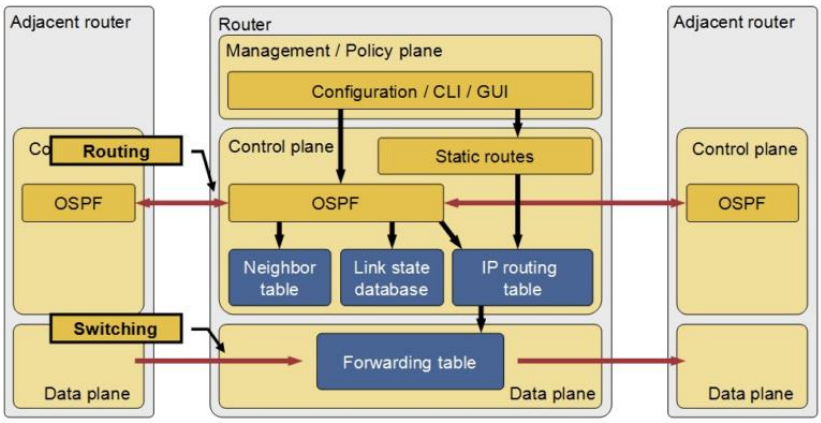
Fuente: (Sociedad et al., 2021)

a) Funcionamiento router tradicional

En su artículo (Ivan Pepelnjak, 2013), comenta que el plano de control es el cerebro de los equipos de red, dado que es el encargado de decidir cómo y dónde se reenvían o procesan los paquetes; además, es el responsable de conocer la topología de red y completar las tablas de enrutamiento. El plano de datos es la parte del hardware y software del dispositivo, donde al recibir un paquete de un puerto de entrada, se realiza una búsqueda en la tabla de enrutamiento para encontrar el puerto de salida del paquete. En la Figura 6, se puede apreciar que en un equipo tradicional estos planos residen en el mismo hardware, donde el plano de control realiza la función de routing utilizando los diferentes protocolos de enrutamiento OSPF, RIP, BGP, etc; ya sea por tabla de vecinos, tablas de enrutamiento o base de datos de estado de enlace. Mientras que el plano de datos realiza la función de switching para realizar la conmutación de puertos de ingreso y salida de paquetes.

Figura 6

Estructura de un Router Tradicional



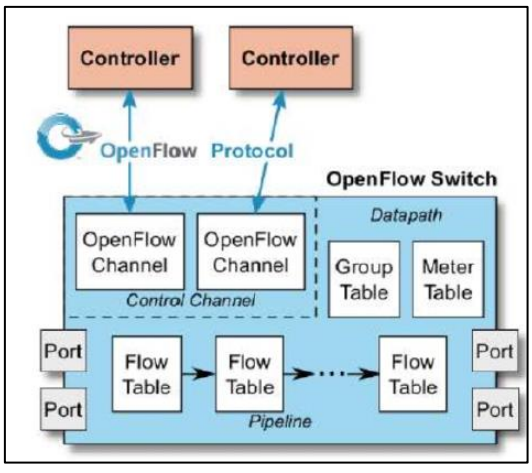
Fuente: (Ivan Pepelnjak, 2013)

b) Funcionamiento equipos SDN

(Wiley, 2020) comenta que el plano de datos SDN es una plataforma que permite que los dispositivos de red anuncien sus capacidades de reenvío y procesamiento de datos al controlador, la ruta de datos reenviará el tráfico entrante de acuerdo con su motor de reenvío. Cada uno de los apartados que componen el plano de datos de la Figura 7, se explican a detalle en la sección 2.3. En la Figura 8 se detalla que, todas las tablas de grupo se agregan a las tablas de flujo, se encuentran ubicadas entre dos tablas de flujo elementales, gracias a esto se puede añadir información adicional en la tabla de flujo para, por ejemplo, permitir un multipunto o destruir algunos paquetes.

Figura 7

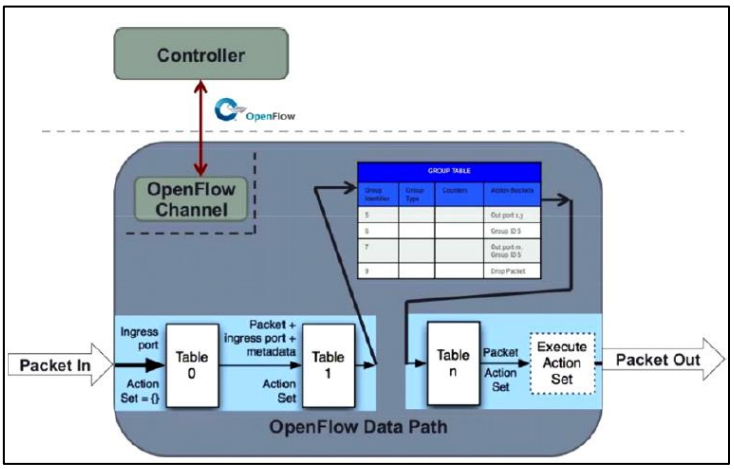
Gestión de tablas de flujo OpenFlow



Fuente: (Wiley, 2020)

Figura 8

Tabla de grupo integrada entre dos tablas de flujo elementales



Fuente: (Wiley, 2020)

2.2.3. Ventajas de SDN frente a MPLS

Para (Wu et al., 2020) una de las ventajas es que los proveedores de servicios de todo el mundo tienen grandes inversiones en infraestructuras de red MPLS altamente sofisticadas y ricas en funciones para brindar servicios a sus clientes, estas infraestructuras se basan en equipos de red tradicionales (plano de datos y plano de control combinados), que son costosos de escalar, complejos de administrar y que requieren mucho tiempo de reconfiguración.

En (Bahnasse et al., 2018; Tajiki et al., 2019) menciona que la virtualización de funciones de red (NFV), la computación en la nube y la proliferación de dispositivos conectados están provocando un aumento exponencial del tráfico y fluctuaciones significativas en los patrones de uso, estas razones hacen que los operadores de red se muevan hacia arquitecturas ágiles que admitan la reconfiguración dinámica tanto de los servicios como de las infraestructuras de red.

(Experimen, 2017) y (Servicio, 2016) coinciden que estas capacidades proporcionan nuevos ingresos, reducen el tiempo de comercialización, aumentan la aceptación de nuevos servicios y mejoran su capacidad para diferenciar significativamente sus ofertas.

En (Campista et al., 2016) menciona que el objetivo de SDN es reducir los costos mediante la virtualización, la automatización y la simplificación, para este propósito, SDN facilita la personalización de las redes, un tiempo de configuración muy corto y un despliegue de red con la calidad de servicio adecuada en lugar de una calidad de servicio general..

2.3. Protocolo OpenFlow

El protocolo OpenFlow es un estándar definido por la ONF (Open Networking Foundation) como la primera interfaz estándar para la comunicación de la capa de control y la capa de infraestructura de la arquitectura, y fue especialmente diseñada para la arquitectura SDN (David Bermúdez Escobar et al., s/f).

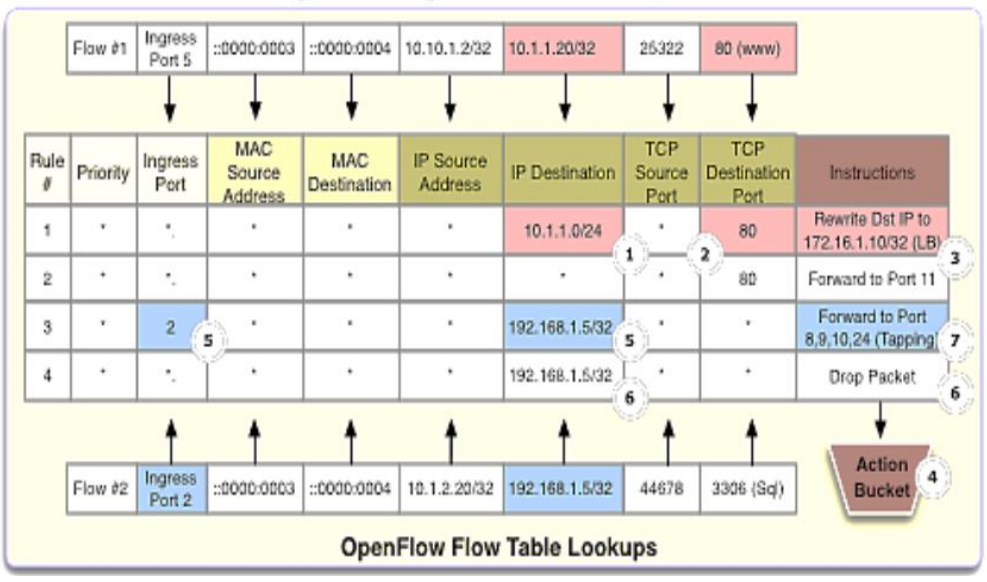
En (Wu et al., 2020) se detalla que OpenFlow proporciona un protocolo abierto para programar el flujo en diferentes dispositivos de red como conmutadores y enrutadores, permite acceder directamente y manipular el plano de direccionamiento de dispositivos de red ya sean físicos o virtuales.

2.3.1. Funcionamiento

OpenFlow aprovecha el hecho que la mayorías de los switches Ethernet contienen tablas de flujo (Flow-Tables), aunque cada una de estas son propias de los fabricantes, se han identificado varias características en común, las cuales son utilizadas por OpenFlow para programar dichas tablas (ONF, 2015); un ejemplo calificado se indica en la siguiente Figura 9.

Figura 9

Tabla de flujo OpenFlow



Fuente: (ONF, 2015)

(Wu et al., 2020) detalla que el primer paquete de una tabla de flujo arriba al switch y este comprueba las tablas de flujo propias con el objetivo de hallar similitudes, si ninguna coincidencia fue encontrada cada paquete se vuelve a enviar al controlador y este se

encargará de añadir la entrada de flujo en la tabla del conmutador switch, luego de insertar la entrada cada paquete que coincida con la entrada añadida nueva se envían directamente al destino sin la necesidad de enviarlo al controlador.

2.3.2. Switch OpenFlow

De acuerdo con (Wiley, 2020), el switch OpenFlow tiene dos partes: la parte que contiene las colas, los transmisores de tramas y los receptores de tramas con las tablas de flujo asociadas a los mismos, y la segunda parte que gobierna la comunicación con el controlador mediante el protocolo de señalización OpenFlow.

La primera parte contiene todos los elementos necesarios para el transporte físico de las tramas del nodo y también contiene las tablas utilizadas para dirigir los flujos a la cola de salida derecha, puede haber una tabla para cada flujo, como una tabla para un conjunto de flujos multiplexados en la misma ruta (Sinha et al., 2017).

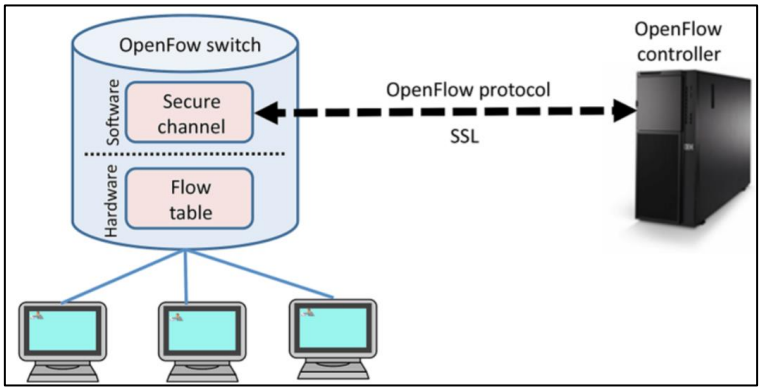
OpenFlow proporciona los elementos necesarios del controlador para crear, administrar y destruir líneas de la tabla de flujo, que también podría denominarse tabla de conmutación, la segunda parte se refiere a la comunicación entre el nodo y el controlador, y la información que debe enviarse entre ellos (Sinha et al., 2017) .

(Wu et al., 2020) menciona que OpenFlow utiliza un canal SSL / TLS seguro para autenticar ambos extremos de la comunicación, lo que reduce en gran medida el riesgo de ataque a la comunicación en curso y exige una autenticación mutua. OpenFlow ofrece los medios para identificar los flujos de paquetes utilizando información de nivel 1, 2, 3 y 4.

Finalmente (Wu et al., 2020) detalla que se envía estadísticas precisas al controlador para que el algoritmo de determinación de ruta pueda hacer su trabajo con un conocimiento casi perfecto del estado de la red, como el controlador está centralizado, hay un ligero retraso en la transmisión de los datos, se detalla este proceso en la Figura 10.

Figura 10

Switch OpenFlow



Fuente: (Wiley, 2020)

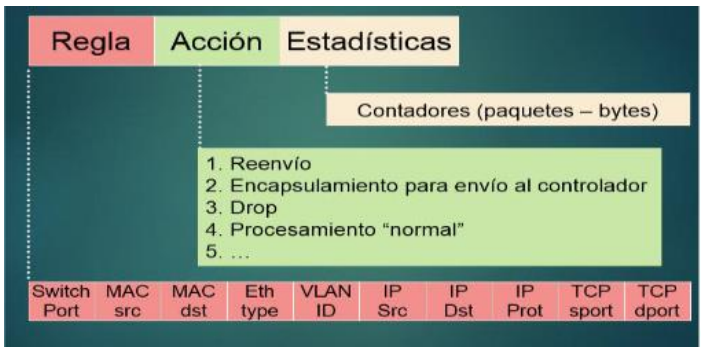
2.3.3. Tabla de flujos OpenFlow

Dado lo mencionado en la sección 2.3.2, en OpenFlow existen dos tipos de tablas: las tablas de flujo y las tablas de grupo, las primeras se encargan de buscar coincidencias entre los paquetes en las FTEs (Flow Table entries) que contienen, además si ocurre una coincidencia aplicar las instrucciones guardadas en la FTE, en cambio, las tablas de grupo únicamente contienen su identificador e instrucciones para aplicar a los paquetes que son enviados a los grupos (Wu et al., 2020).

Las tablas pueden procesar los paquetes en un pipeline, su flujo está determinado por las acciones que se ejecutan en las FTEs coincidentes con el paquete, estas pueden no redirigir el paquete a ninguna otra tabla de flujo o grupo por lo cual terminaría su procesamiento (Wu et al., 2020), en la Figura 11 se representa la estructura de una FTE.

Figura 11

Estructura de una tabla de flujos OpenFlow



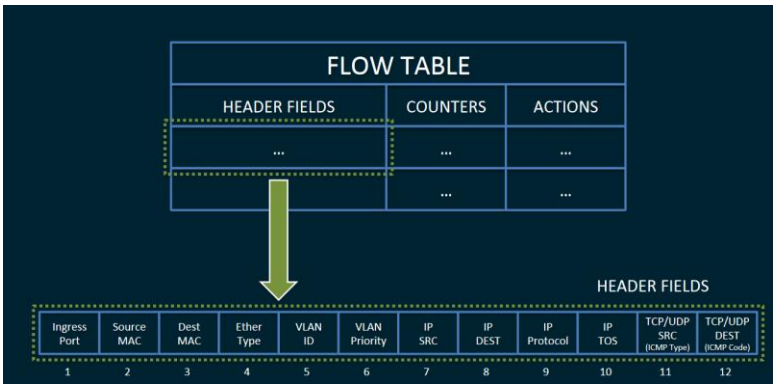
Fuente: (ONF, 2013)

a) Campos de Cabecera

Los campos de cabecera contienen información que se encuentra en la cabecera del paquete y es utilizada para comparar o buscar coincidencia con los paquetes entrantes, formados por el puerto de ingreso, los encabezados de paquetes y opcionalmente metadatos especificados en una tabla previa (Leitón, 2015), estos campos se representan en la Figura 12.

Figura 12

Campos de cabecera



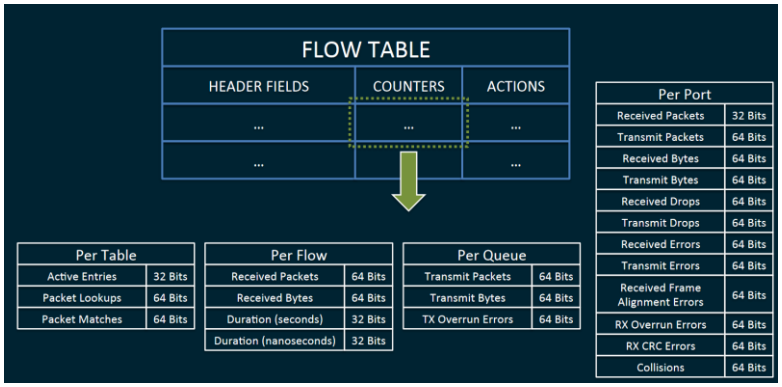
Fuente: (ONF, 2013)

b) Contadores

Estos son utilizados para acumular estadísticas del flujo particular, así como el número de paquetes recibidos, número de bytes, y el tiempo de vida del flujo, de igual manera son actualizados cuando existen coincidencias con los paquetes (Leitón, 2015), en la Figura 13 se especifican cada uno a detalle.

Figura 13

Contadores



Fuente: (ONF, 2013)

c) Acciones

Según (OPEN NETWORKING FOUNDATION, 2012) cada entrada de flujo se asocia con cero o más acciones que dictan cómo el switch trata a los paquetes coincidentes, en el caso de no haber acciones de envió presentes, el paquete es descartado, las listas de acciones para las entradas de flujo insertadas deben ser procesadas en el orden especificado, aunque, no hay un orden de paquetes de salida garantizado dentro de un puerto.

De igual forma (Vissicchio et al., 2014) comenta que un switch puede rechazar una entrada de flujo si no se puede procesar la lista de acciones en el orden especificado, en cuyo caso se debería devolver inmediatamente un error de flujo no soportado. A continuación, se detalla la lista de acciones soportadas por el protocolo OpenFlow:

1. **Acciones requeridas:** Los switches OpenFlow deben soportar el reenvío el paquete a puertos físicos y a los siguientes puertos virtuales:

   ALL: Envía el paquete a todas las interfaces, sin incluir la interface de entrada.

   CONTROLLER: Encapsular y enviar el paquete al controlador.

   TABLE: Realizar acciones en la tabla de flujo. Solamente para mensajes de tipo packet-out.

   IN_PORT: Enviar el paquete al puerto de entrada.

   ANY: Valor especial utilizado en algunos comandos de OpenFlow cuando no se especifica ningún puerto

2. **Acciones opcionales:** El switch puede soportar opcionalmente los siguientes puertos virtuales:

   LOCAL: Representa la pila de red local del conmutador y su pila de administración. Puede usarse como puerto de entrada o como puerto de salida.

   NORMAL: procesar el paquete utilizando la ruta de reenvío tradicionales soportada por el switch (es decir, el procesamiento tradicional de la capa 2, VLAN, y La capa 3). El switch se puede comprobar el campo VLAN para determinar si debe o no reenviar el paquete a lo largo de la ruta de procesamiento normal. Si el switch no puede enviar entradas por el VLAN OpenFlow

especificada regresa a la ruta de procesamiento normal, esto debe indicar que no admite esta acción.

FLOOD: Inundación del paquete a todas las interfaces del switch excepto la interfaz de entrada.

3. **Acción opcional: Set-Queue.** La acción set-queue envía un paquete a través de una cola conectada a un puerto, el comportamiento de reenvío está dictaminado por la configuración de la cola y se utiliza para proporcionar soporte básico de calidad de servicio (QoS).
4. **Acción opcional: Drop.** Una entrada de flujo con ninguna acción especificada indica que todos los paquetes que concuerden deben descartarse.
5. **Acción opcional: Group**. Procesa el paquete a través del grupo especificado. La interpretación exacta depende del tipo de grupo.

d) Canal OpenFlow

El canal OpenFlow es la interfaz que conecta cada conmutador OpenFlow a un controlador, a través de esta interfaz, el controlador configura y administra el conmutador, recibe eventos del conmutador y envía paquetes fuera del conmutador (OPEN NETWORKING FOUNDATION, 2012).

2.3.4. Tipos de Switches OpenFlow

Según (Vissicchio et al., 2014) existen dos tipos de switches OpenFlow: OpenFlow-only y los OpenFlow-híbridos que se describen a continuación.

a) OpenFlow-only

Los switches solo admiten la operación OpenFlow, en esos switches todos los paquetes son procesados por OpenFlow pipeline y no se pueden procesar de otra manera (Vissicchio et al., 2014).

b) OpenFlow-hybrid

Los conmutadores admiten tanto la operación OpenFlow como la operación de conmutación normal de Ethernet, es decir, conmutación Ethernet L2 tradicional, aislamiento de VLAN, enrutamiento L3 (enrutamiento IPv4, enrutamiento IPv6 ...), procesamiento ACL y QoS (Vissicchio et al., 2014).

2.4. Plano de Control o Controlador SDN

Respecto al plano de control o el controlador SDN, (Xie et al., 2019) menciona que es el cerebro de los sistemas SDN, que puede programar recursos de red, actualizar reglas de reenvío dinámicamente y hacer que la administración de la red sea flexible y ágil, el componente principal de CP es el controlador lógicamente centralizado, que controla la comunicación entre los dispositivos de reenvío y las aplicaciones.

(Bermúdez, Cristian; Rodríguez, Jesús; Dussan, 2020) detallan que por un lado, el controlador expone y abstrae la información del estado de la red del plano de datos al plano de la aplicación y, por otro lado, el controlador traduce los requisitos de las aplicaciones en políticas personalizadas y las distribuye a los dispositivos de reenvío.

Además, el controlador proporciona funcionalidades esenciales que la mayoría de las aplicaciones de red necesitan, como enrutamiento de ruta más corta, almacenamiento de topología de red, configuración de dispositivos y notificaciones de información de estado, etc. Hay muchas arquitecturas de controlador, como NOX, POX, Floodlight, Ryu , OpenDayLight y Beacon (Bermúdez, Cristian; Rodríguez, Jesús; Dussan, 2020); algunos de los controladores se exponen en la Tabla 1, en donde se detalla la versión de OpenFlow que soportan y el tipo de lenguaje que manejan.

Tabla 1

Controladores SDN

Controlador	Lenguaje	Creado por	Versión OpenFlow
NOX	Python, C++	Nicira	1.0, 1.3
POX	Python (2.7)	Nicira	1.0
Beacon	Java	Stanford university	1.0.1
Maestro	Java	Rice university	1.0
Floodlight	Java	Big Switch Networks	1.0
Floodlight-plus	Java	Big Switch Networks	1.3
Ryu	Python	NTT Labs	1.0 - 1.4
(ODL) OpenDayLight	Java	Linux Foundation	1.0, 1.3

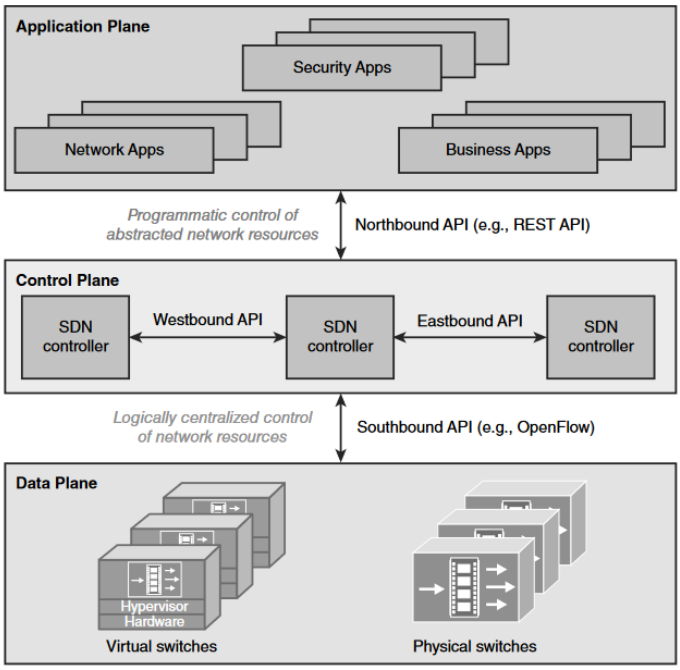
Fuente: *Adaptado de (Ungerman, s/f)*

Tres interfaces de comunicación permiten que los controladores interactúen: southbound, northbound y eastbound/westbound interfaces (Bermúdez, Cristian; Rodríguez,

Jesús; Dussan, 2020), estas se representan en la Figura 14, donde de igual manera se puede entender cómo se comunican con el plano aplicación y el plano de datos.

Figura 14

Interfaces del controlador SDN



Fuente: (Stallings et al., 2016)

2.5. Servicios Diferenciados (DiffServ)

Según (Andersson & Brohne, 2017) DiffServ es una arquitectura de redes informáticas que utiliza valores de puntos de código de servicios diferenciados (DSCP) por sus siglas en inglés, para clasificar los paquetes que pasan por su dominio, este es una colección de enrutadores que utilizan las mismas políticas Diffserv definidas. Un dominio DiffServ se compone de un grupo de nodos interconectados que utilizan la misma política de servicio y PHB (Per Hop Behavior), cada enrutador dentro del dominio está configurado para diferenciar la prioridad del tráfico según la clase que tengan los paquetes entrantes.

Sin embargo (Stephen Richardson, 2021) menciona, que el marcaje QoS permite asignar una aplicación a una clase de tráfico según su prioridad y tipo, en la Tabla 2 muestra el conjunto de clases de tráfico asignadas según el tipo de aplicación. Se recuerda que el campo *Class Selector* (CS) está codificado en 3 bits; 8 valores posibles que van de 0 a 7, donde 0 = Mejor esfuerzo y 7 es el tráfico de red.

Tabla 2

Clasificación de servicios

Aplicación	DSCP	AF	CS	RFC
Best Effort	0	0	0	2474
Scavenger	8	CS1	1	3662
Bulk-Data	10, 12, 14	AF11, AF12, AF13	1	2597
Network Management	16	CS2	2	2474
Transactional Data	18, 20, 22	AF21, AF22, AF23	2	2597
Call Signaling	24	CS3	3	2474
Mission Critical	26, 28, 30	AF31, AF32, AF33	3	2597
Real Time Interactive	32	CS4	4	2474
Interactive Video	34, 36, 38	AF41, AF42, AF43	4	2597
Voice	46	EF	5	3247
Routing	48	CS6	6	2474
Network	56	CS7	7	N/A

Fuente: Adaptado de (Quality of Service QoS - My IT notes, s/f)

CAPÍTULO III

DISEÑO DE RED DEFINIDA POR SOFTWARE SDN Y MPLS

Este apartado va dirigido a la forma en la que se va a plantear el escenario en el simulador de tiempo real GNS3, incluyendo los servicios, servidores y la topología que lo conforman, además se plantea el ambiente de la Red Híbrida SDN/MPLS que será diseñado con el fin de realizar las pruebas de funcionamiento correspondientes de QoS en Servicios Diferenciados (DiffServ).

3.1. GNS3

Es un software utilizado por cientos de miles de ingenieros de redes en todo el mundo para emular, configurar, probar y solucionar problemas de redes virtuales y reales en un ambiente controlado. Permite la ejecución de topologías que constan de unos pocos dispositivos, hasta varios alojados en múltiples servidores o incluso alojados en la nube (GNS3 Staff, 2020). Por último, resaltar que es un software gratuito de código abierto cuyo sitio web es **http://gns3.com**; en él se puede encontrar los enlaces de descarga y la documentación respectiva.

Para la implementación de GNS3 se necesitan algunos requisitos de hardware (RQ_H#), los mismos se plantean en base a que serán implementados en una máquina portátil, en la Tabla 3 de acuerdo con su prioridad se detalla cada uno.

Tabla 3

Requerimientos de hardware para GNS3

Nro.	Requerimiento	Prioridad	Descripción
RQ_H1	Sistema Operativo	Baja	Ser multiplataforma: Windows, Linux, Mac
RQ_H2	Procesador	Alta	4 o más núcleos lógicos: serie AMD-V/RVI o Intel VT-X/EPT
RQ_H3	Virtualización	Media	Se requieren extensiones de virtualización
RQ_H4	Memoria RAM	Alta	16GB RAM
RQ_H5	Almacenamiento	Alta	Unidad de estado sólido (SDD) con 35 GB de espacio disponible

De la misma manera, dentro del entorno de GNS3 se tiene requerimientos a nivel de software para cada uno de los equipos como: routers, servidores, ODL y Open vSwitch. En la

Tabla 4 se plantean los requerimientos de software (RQ_S#) para cada uno de los equipos, se hace énfasis en apartados críticos como memoria RAM y CPU para que puedan ser ejecutados en máquinas virtuales de acuerdo con la capacidad de la máquina anfitrión.

Tabla 4

Requerimientos de software en el entorno de GNS3

Nro.	Equipo	Memoria RAM	Núcleos	Descripción
RQ_S1	Router	512MB	1	Se basa en los IOS que Cisco provee
RQ_S2	Open vSwitch	256MB	1	Basado en Kernel Linux 3.2
RQ_S3	Hosting y Streaming	4GB	4	Kernel Linux 4.4 o posteriores para Ubuntu 16
RQ_S4	ODL	4GB	2	Basado en Kernel Linux 3.19.X para Ubuntu 14
RQ_S5	VoIP	2GB	2	Kernel Linux 3.10 para CentOS 7

En la Tabla 5 se plantean las especificaciones de la máquina anfitrión Acer Aspire VX15, en ella se toman aspectos importantes como procesador, memoria RAM, memoria VRAM y sistema operativo.

Tabla 5

Especificaciones máquina anfitrión

Características	Descripción
Procesador	Intel Core i7-7700HQ Quad-core a 2,80 GHz con capacidad de 8 hilos por núcleo
RAM	24GB DDR4 a 2700Mz
Gráfica	NVIDIA GeForce GTX 1050 con 4 GB de VRAM y HD Graphics 630 de Intel
Disco Duro	512GB SSD NVMe 3.0 y 1TB HDD

Sistema Operativo	Windows 10 Home Single Language

Para finalizar, se ha elaborado en la Tabla 6 un resumen donde se validan cada uno de los requerimientos de hardware y software que requiere GNS3, de acuerdo con las especificaciones de la máquina anfitrión, la cual será la encargada de soportar los servicios necesarios para el desarrollo del proyecto.

Tabla 6

Resumen de requerimientos para la máquina anfitrión

Nro. Servicio	Servicio	Cumplimiento
RQ_H1	GNS3	✓
RQ_H2		✓
RQ_H3		✓
RQ_H4		✓
RQ_H5		✓
RQ_S1	Equipos y Servidores en GNS3	✓
RQ_S2		✓
RQ_S3		✓
RQ_S4		✓
RQ_S5		✓

3.2. Topología de Red

La topología de red, como su nombre lo indica, es la va a permitir la realización del esquema de distribución de la red, misma que se conforma con la parte SDN, controlador y plano de datos, así como también, la parte MPLS con la distribución de los diferentes servicios y los clientes que accederán a los mismos, cabe recalcar que al estar simulando un entorno real se ha tomado en cuenta la distribución de las facultades de la Universidad, las cuales servirán para dar a entender los enlaces WAN (Wide Area Network) y las redes LAN (Local Access Network), con sus respectivos hosts que son los que van a acceder a los servicios de la DMZ (Demilitarized Zone).

3.2.1. Topología Lógica

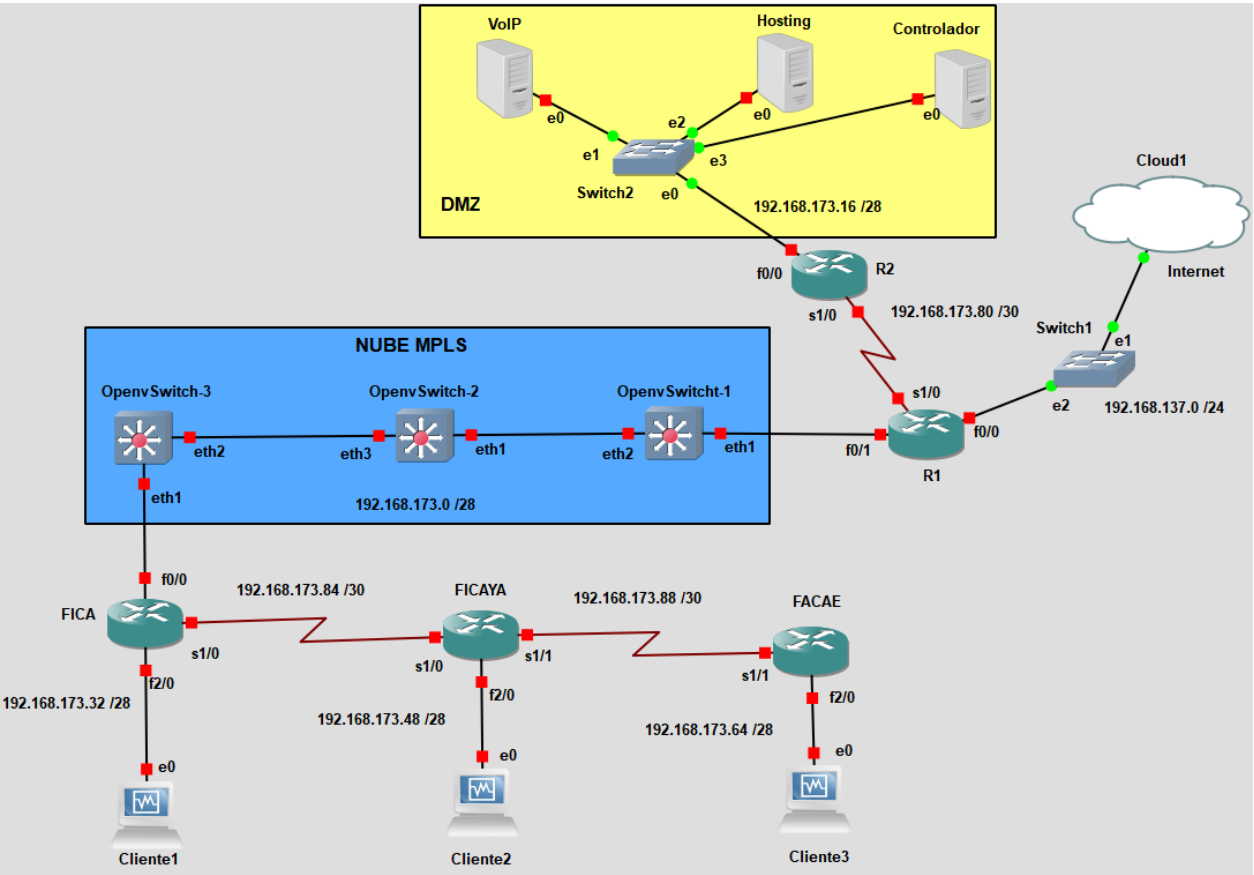
La topología lógica, como bien dice su nombre, es la que indica la distribución lógica de la red, misma que se conforma por la red híbrida SDN/MPLS, donde se realizará la interoperabilidad entre los equipos SDN y routers tradicionales, para establecer la conexión entre hosts, redes y servidores. La Figura 15 detalla la topología propuesta, incluyendo cada uno de los enlaces (WAN y LAN), el direccionamiento IP de estos y cada uno de los equipos que serán los encargados de realizar la simulación correspondiente, como si de un entorno real se tratase.

El diseño propuesto se basa en cumplir las necesidades de los usuarios, además de tener una red ordenada y fácil de administrar para los encargados de TI (Tecnología de la Información), la parte de la DMZ se plantea con el objetivo de tener mayor seguridad al tratarse de servicios que pueden accederse desde Internet, de esta manera se puede tener un mejor control del tráfico de la red aplicando teoría de colas y también permitir diferentes aplicaciones de cortafuegos para brindar servicios confiables y seguros. Otro aspecto importante es el router de frontera (R1), este es encargado de dividir la red interna y la red pública, es decir, es el último router que se controla antes de internet, así como también la primera y última línea de defensa.

Para la nube MPLS y las redes LAN se hace el diseño en base a una red de backbone distribuida, esto debido a que es fácilmente escalable si la red requiere expandirse, permite una administración simple de la red, y además al ser una red SDN con el controlador ODL se puede tener control total sobre los equipos que funcionan y son parte de la red. Lo que se busca, es que la red se pueda expandir de acuerdo con las necesidades que se vayan generando con el pasar del tiempo.

Figura 15

Topología Lógica Red Híbrida SDN/MPLS



Una vez establecida la topología, esta debe responder a ciertas interrogantes para determinar su factibilidad al momento de ser implementada, en la Tabla 7 se tabulan los requerimientos que responden a todas las preguntas acerca del diseño de red.

Tabla 7

Requerimientos para la topología lógica

Requerimiento	Descripción
Rentabilidad	Una de las ventajas de una red SDN es el ahorro económico que brinda, esto es porque los equipos SDN se encuentran basados en software y los convencionales en hardware.
Escalabilidad	Se cumple con este requerimiento, dado que el diseño propuesto es flexible y escalable conforme las necesidades de la red lo requieran, ya sean más servidores en la DMZ o mayor número de usuarios en las redes LAN.

Facilidad para detectar fallos	La topología muestra un mapa sencillo de interpretar para los Ingenieros de TI, dado que las interfaces están identificadas y etiquetadas, por lo tanto, cumple este requerimiento.
Tecnología implementada	El diseño es propio y partido desde cero, además los protocolos de comunicación como OSPF para el enrutamiento o MPLS poseen una gran documentación, tomando en cuenta que permiten una administración eficiente de la red.
Rutas alternativas	Este apartado no cumple la topología debido a las limitaciones en hardware de la máquina anfitrión, aunque sí está pensado para admitir rutas alternativas para permitir la expansión de la red.

La Tabla 8 detalla los dispositivos que se encuentran en la topología, describiendo el sistema operativo en el que operan, la función que cumple cada uno de ellos y el nombre por el que son identificados. Exceptuando los equipos SDN que se explicarán y detallarán más adelante.

El servidor Hosting (Web, Streaming), se establece en el SO Ubuntu al ser ideal para correr con tecnologías como C, Perl, MySQL o PHP con servidores web Nginx o Apache debido a que están optimizadas para este entorno, ya que son lenguajes creados específicamente bajo estas plataformas basadas en Linux y Unix, permitiendo sacar el máximo rendimiento.

El servidor VoIP se establece en el SO CentOS al tratarse de un sistema mucho más pequeño que otras distros, por lo que cada administrador puede configurarlo según sus necesidades, evitando tener paquetes innecesarios instalados de serie. Sus dos puntos fuertes son la estabilidad y el soporte extendido que permite usar la distro sin tener que cambiar de versión durante muchos años.

Tabla 8

Dispositivos de la Red

Dispositivo	Sistema Operativo	Función/Servicio
Cliente 1	Ubuntu 16.01	Host LAN FICA
Cliente 2	Kali Linux	Host LAN FICAYA
Cliente 3	Ubuntu 18.01	Host LAN FACAE

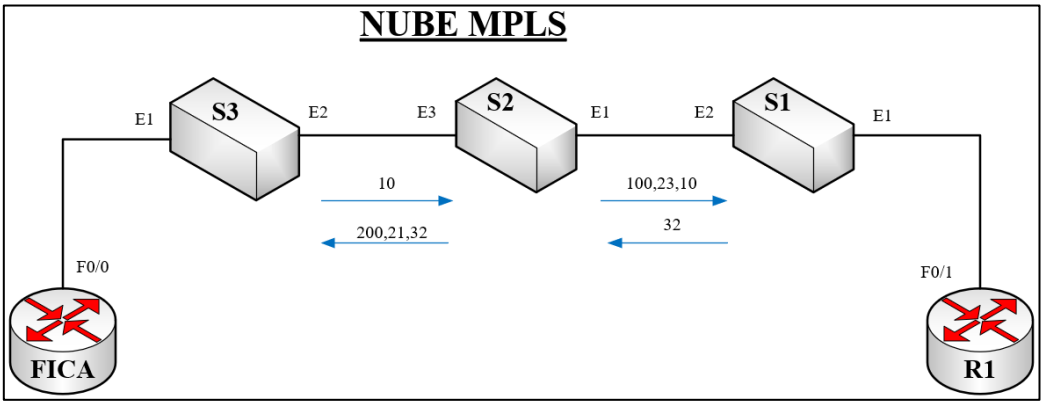
VoIP	Issabel	Servidor de VoIP
Hosting	Ubuntu 16.01	Servidor WEB, FTP y Streaming
Controlador	Ubuntu 14.01	Plano de control

El segmento de red de la nube MPLS se plantea en la Figura 16, consta de tres Open vSwitch (S1, S2 y S3), los cuales son los encargados del plano de datos SDN, junto con los routers CISCO C7200 (R1 y FICA) que son los encargados del enrutamiento de la red, en conjunto forman la red híbrida SDN/MPLS.

Este escenario propuesto se basa en la Figura 1, donde se hace uso de routers tradicionales; pero ahora estos han sido reemplazados con los Open vSwitch. Estos equipos de igual manera cumplen con la conmutación de etiquetas MPLS, realizando operaciones PUSH, SWAP y POP.

Figura 16

Nube MPLS



Una vez planteado el escenario, lo siguiente es definir la designación de puertos y las respectivas etiquetas; en S3 se establecen dos puertos, el puerto E1 se establece el enlace con el router de la red interna y el puerto E2 con el puerto E3 de S2 con la etiqueta 10, realizando operaciones de PUSH y POP. Para S2, algo importante a tomar en cuenta, es que hace operación SWAP, es decir, todo lo que viene con etiqueta 10 por el puerto E1 hacia E2 de S1, se transforma en las etiquetas 100, 23 y 10, y viceversa, todo lo que viene con etiqueta 10 de parte de S1, pasa a tomar el valor de 200, 21 y 32. Por último S1, realiza el mismo trabajo que S3, sin embargo este Open vSwitch se conecta con el router de frontera R1, el cual, es el encargado de redirigir los paquetes hacia la red pública y también hacia la DMZ. En la Tabla 9, para tener una mejor noción de las configuraciones de cada equipo en la nube MPLS, se las describe a detalle.

Tabla *9*

Enlaces y Etiquetas en la nube MPLS

Enlace	Etiqueta	Operación
S1 E1-F0/1	N/D	POP
S1 E2-E1	32	PUSH
S2 E1-E2	100, 23, 10	SWAP
S2 E3-E2	200, 21, 32	SWAP
S3 E2-E3	10	PUSH
S3 E1-F0/0	N/D	POP

Así como hay una distribución de etiquetas y puertos, también es importante el direccionamiento IP correspondiente para cada enlace, mismo que se encuentra configurado dentro de los routers, Open vSwitch, servidores y host correspondientes a cada LAN. En la Tabla 10 se describe a detalle el dispositivo identificado en la topología, su interfaz, dirección IP, máscara de red, gateway y la dirección de la subred. Para aclarar, se ha tomado como referencia la IP 192.168.173.0 /24, la cual por medio de VLSM se ha determinado las siguientes máscaras: /28 para las redes LAN, DMZ y la nube MPLS, por último /30 para los enlaces WAN.

Tabla 10

Direccionamiento IPv4

Disp.	Interfaz	Dirección IP	Máscara de Subred	Gateway	Dirección de Subred
R1	S1/0	192.168.173.81	255.255.255.252	N/D	192.168.173.80 /30
	F0/0	192.168.137.2	255.255.255.0	N/D	192.168.137.0 /24
	F0/1	192.168.173.1	255.255.255.240	N/D	192.168.173.0 /28
R2	S1/0	192.168.173.82	255.255.255.252	N/D	192.168.173.80 /30
	F0/0	192.168.173.17	255.255.255.240	N/D	192.168.173.16 /28
FICA	S1/0	192.168.173.85	255.255.255.252	N/D	192.168.173.84 /30
	F0/0	192.168.173.8	255.255.255.240	N/D	192.168.173.0 /28
	F2/0	192.168.173.33	255.255.255.240	N/D	192.168.173.32 /28
FICAYA	S1/0	192.168.173.86	255.255.255.252	N/D	192.168.173.84 /30
	S1/1	192.168.173.89	255.255.255.252	N/D	192.168.173.88 /30

	F2/0	192.168.173.49	255.255.255.240	N/D	192.168.173.48 /28
FACAE	S1/1	192.168.173.90	255.255.255.252	N/D	192.168.173.88 /30
	F2/0	192.168.173.65	255.255.255.240	N/D	192.168.173.64 /28
S1	E1	192.168.173.2	255.255.255.240	192.168.173.1	192.168.173.0 /28
	E2	192.168.173.3	255.255.255.240	192.168.173.1	192.168.173.0 /28
S2	E1	192.168.173.4	255.255.255.240	192.168.173.1	192.168.173.0 /28
	E3	192.168.173.5	255.255.255.240	192.168.173.1	192.168.173.0 /28
S3	E1	192.168.173.6	255.255.255.240	192.168.173.1	192.168.173.0 /28
	E2	192.168.173.7	255.255.255.240	192.168.173.1	192.168.173.0 /28
VoIP	E0	192.168.173.18	255.255.255.240	192.168.173.17	192.168.173.16 /28
Hosting	E0	192.168.173.19	255.255.255.240	192.168.173.17	192.168.173.16 /28
ODL	E0	192.168.173.20	255.255.255.240	192.168.173.17	192.168.173.16 /28
Cliente1	E0	192.168.173.34	255.255.255.240	192.168.173.33	192.168.173.32 /28
Cliente2	E0	192.168.173.50	255.255.255.240	192.168.173.49	192.168.173.48 /28
Cliente3	E0	192.168.173.66	255.255.255.240	192.168.173.68	192.168.173.64 /28
Cloud	E0	192.168.137.1	255.255.255.240	N/D	192.168.137.0 /24

3.2.2. Enrutamiento OSPF

Existen diversos protocolos de enrutamiento para permitir la interconexión de la red, en este caso se hace uso de OSPF y como lo menciona (Mendez, 2020) éste se encuentra diseñado como un protocolo perfecto; además permite conexiones punto a multipunto a diferencia del protocolo IS-IS que no es compatible. Otro aspecto para tomar en cuenta es que, un enrutador OSPF es capaz de pertenecer a múltiples áreas, mientras que un enrutador IS-IS solo puede pertenecer a una única área.

El protocolo OSPF es de tipo enlace-estado y basado en el algoritmo de vía más corta (SPF); los routers OSPF necesitan establecer una relación de vecindad antes de intercambiar actualizaciones de enrutamiento. Para establecer el enrutamiento OSPF existen tres consideraciones a tomar en cuenta: primero declarar la dirección IP de subred por cada interfaz involucrada, segundo hay que declarar la máscara de wildcard, en el caso del presente proyecto

se tiene redes /28 con máscara de wildcard 0.0.0.15 y para /30 se utiliza 0.0.0.3, finalmente se establece el área, en este caso todos los routers pertenecen al área 0.

Al momento de identificar un router en la topología, una de las herramientas de OSPF es el Router ID, que se establece con el comando **`router-id rid,`** donde *rid* contiene 32 bits y se representa como si fuese una dirección IPv4; permite definir el *Router Designado* (DR) y el *Router Designado de Respaldo (BDR),* donde DR es el router con el ID más alto y BDR es el segundo con el ID más alto. En caso de que el router ID no se configure, se elige la dirección IPv4 más alta de cualquiera de las interfaces loopback configuradas, y si no estuviese configurado este segundo método, finalmente el router elige la dirección IPv4 activa más alta de cualquiera de sus interfaces físicas, aunque esto no es recomendable para los administradores de la red porque es más difícil identificar un router en la red.

En la Tabla 11, se presenta los respectivos router ID configurados para permitir identificar cada router en la topología, hay que aclarar que el costo de los enlaces no se establece, ni tampoco direcciones de loopback.

Tabla 11

Router ID para cada router de la red

Equipo	Router ID
R1	1.1.1.1
R2	2.2.2.2
FICA	3.3.3.3
FICAYA	4.4.4.4
FACAE	5.5.5.5

3.3. Servidores DMZ

En la DMZ de la topología, se encuentran implementados los servidores basados en las necesidades de los usuarios, así como también el controlador SDN, el cual, se explicará más adelante. Los servidores que se encuentran son: VoIP (Issabel), Hosting (Apache) y Streaming (Streama), mismos que son descritos a continuación en los siguientes incisos.

- **Servidor VoIP**

  Issabel es un software que nació en 2016 por parte de la comunidad Asterisk, surgió con la intención de evitar perder todos los avances que se realizaron con Elastix, el cual, es su predecesor; se trata de un software de código abierto de telefonía IP y

comunicaciones unificadas basadas en Asterisk, sus numerosas aplicaciones hacen que sea una de las más completas herramientas de comunicaciones (*About Us » Issabel.org*, 2016). En este caso, los docentes y las áreas administrativas de todas las facultades son los que se benefician de tener un servicio confiable de VoIP y con la mejor calidad.

- **Servidor Hosting**

  El proyecto de servidor Apache HTTP (The Apache Software Foundation, 2011), es un software de servidor web gratuito y de código abierto para plataformas UNIX, este servidor se ha vuelto popular debido a su modularidad y actualización constante por parte de la comunidad. Se lo puede encontrar en la mayoría de hosting a nivel mundial dado que utiliza una estructura cliente-servidor, es decir, el navegador (Chrome, Mozilla, Safari, etc.) envía una solicitud al servidor y Apache devuelve la respuesta con todos los archivos solicitados. El servidor y el cliente se conectan a través del protocolo HTTP y el servidor es el encargado de tener una comunicación fluida y segura entre las dos máquinas.

- **Servidor Streaming**

Streama es un servidor de streaming de medios gratuito (ESGEEKS, s/f), se ejecuta en Java y que se puede usar en cualquier distribución de Linux. Su soporte de visualización sincronizada en vivo permite que se pueda visualizar videos remotamente, dado que tiene soporte multiusuario, es decir, se puede crear cuentas individuales para cada uno de los integrantes de la universidad.

En la Tabla 12 se indica los requerimientos en hardware y software que tiene cada VM, estos se basan según lo que mencionan (GNOME, 2016) y (DanielLOP, 2021) en cuanto a requerimientos mínimos para el funcionamiento de cada SO, los mismos tendrán una variación de acuerdo con la necesidad que requiera cada servidor, evitando de esta manera cualquier problema a futuro si hay una expansión en la red.

Tabla 12

Especificaciones VM para servidores

Servicio	Versión de Software	Hardware Recomendado	Hardware Implementado
Web, FTP, Streaming	16.04 LTS	7GB HDD 1,5GB RAM	30GB HDD 4GB RAM

VoIP	CentOS 7	20GB HDD 500Mb RAM	30GB HDD 3GB RAM

3.4. Diseño de la Red Definida por Software

Para la implementación de la Red Definida por Software (SDN), es necesario una plataforma o medio que sirva como infraestructura de hardware base para que apile las capas que componen una SDN (Capa Infraestructura, Capa de Control y Capa de Aplicación). En este trabajo no se utiliza una infraestructura hiperconvergente, dado que cada uno está montado en su propia máquina virtual.

Una vez que el escenario virtual ha sido establecido, ya es posible la implementación de cada una de las capas que componen una SDN, empezando por la capa de control, la capa de aplicación y por último la capa de infraestructura. A continuación, se detalla el proceso que se sigue en cada una de ellas.

a) Capa de Control

Como ya se hizo mención en la sección 2.4 hay diferentes tipos en lo que a controladores SDN se refiere, por esto en la Tabla 13 se presenta una comparación entre los diferentes controladores que han sido citados anteriormente; a estos se les da una calificación con puntos que van entre 0 y 1, donde 0 significa que no cumple el parámetro y 1 que si lo cumple. Al final el que más puntos suma será el controlador elegido para el presente proyecto.

Los parámetros que se tomarán en cuenta son: soporte OpenFlow (principalmente), que sean de código abierto, soporte multiplataforma, API REST, interfaz web, documentación y multi-thread. Todos estos parámetros mencionados se describen a continuación.

- *Soporte OpenFlow*: Hace referencia a que el controlador tenga soporte para el protocolo OpenFlow garantizando la comunicación con la capa de infraestructura.
- *Código abierto*: El software cuyo código fuente se ha puesto a disposición de todo el mundo de manera gratuita y otorgado con licencias para su adaptación en contextos diferentes.
- *Soporte multiplataforma*: Quiere decir si el software funciona en varios sistemas operativos, estos pueden ser libres o comerciales.

- *API REST*: Describe un tipo de API, donde permite que las aplicaciones se asienten en diferentes hosts, utilizando mensajes HTTP para transferir datos a través de la API.
- *Interfaz web*: Permite que los encargados de la SDN no deban usar constantemente comandos para su administración.
- *Documentación*: Para que se pueda manejar de manera más sencilla el controlador debe tener una buena documentación, esto tanto para configuraciones que se deseen hacer, como para solucionar posibles problemas.
- *Multi-thread*: Soporte en hardware para ejecutar eficientemente múltiples hilos de ejecución, con el objetivo de aumentar el rendimiento optimizando la utilización de la CPU.

Tabla 13

Selección de controlador SDN

Controlador / Criterios	NOX	POX	Beacon	Maestro	RYU	ODL
OpenFlow	1	1	1	1	1	1
Código abierto	1	1	1	1	1	1
Multiplataforma	0	1	1	1	0	1
API REST	1	1	1	1	0	1
Interfaz web	1	1	1	1	1	1
Documentación	0	0	0	0	0	1
Multi-thread	0	0	1	1	1	1
Total	4	5	6	6	4	7

Con una puntuación total de siete puntos, el controlador ODL es el seleccionado dado que cumple con todos los parámetros que se buscan para el diseño de la SDN, es por esto por lo que será implementado para la topología planteada.

El controlador ODL se instala en una máquina virtual (VM) conforme lo establece el Anexo 1; los requerimientos de hardware y software de esta máquina se especifican en la Tabla 14, donde hay que tomar en cuenta la versión de la distribución de Ubuntu con la cual la distribución ODL pueda operar, con el propósito de que el administrador lo pueda ejecutar sin ningún inconveniente y con la libertad de poder agregar nuevas aplicaciones extras en el servidor.

Tabla 14

Especificaciones de Máquina Virtual

Software	Versión	Versión Java	Hardware Recomendado	Hardware Implementado
Ubuntu	14.04 LTS	1.7	16GB HDD 2GB RAM	30GB HDD 4GB RAM
OpenDayLight	Lithium			

Cabe resaltar que posterior a la instalación de ODL en la VM, se debe añadir las características o módulos adicionales para establecer la comunicación con las demás capas de la SDN, los cuales se detallan a continuación en la Tabla 15.

Tabla 15

Características o Módulos ODL

Módulo	Descripción	Comando
Model-Driven (MD-SAL)	Permite acceder mediante interfaz web a una lista de API´s de modelado de datos, además proporciona funcionalidad de mensajería y almacenamiento de datos con modelos definidos por el usuario	odl-mdsal-all
Soporte RESTCONF API	Habilita el acceso de la API REST a MD-SAL, incluido el almacén de datos.	odl-restconf
L2 Switch	Proporciona reenvío L2 (Ethernet) a través de conmutadores OpenFlow conectados y soporte para host	odl-l2switch-switch
OpenFlow Plugin	Estándar para permitir la interacción y comunicación entre las capas de la arquitectura SDN, permite utilizar el puerto 6633 del protocolo OpenFlow.	odl-openflowplugin-all
DLUX	Proporciona una interfaz gráfica de usuario intuitiva para OpenDayLight	odl-dlux-all

Fuente: Adaptado de (Beryllium & Project, 2017)

b) Capa de Aplicación

Para la capa aplicación no es necesario una configuración adicional, solamente en la misma máquina virtual del controlador se deben instalar los módulos adicionales, los más importantes es el módulo web y el API REST del cual dispone, permitiendo de esta manera la administración dependiendo de las necesidades que los usuarios presenten en la SDN, como el descubrimiento de nuevas redes, implementación de servidores, bloqueo de interfaces, entre otras que llegasen a suscitar.

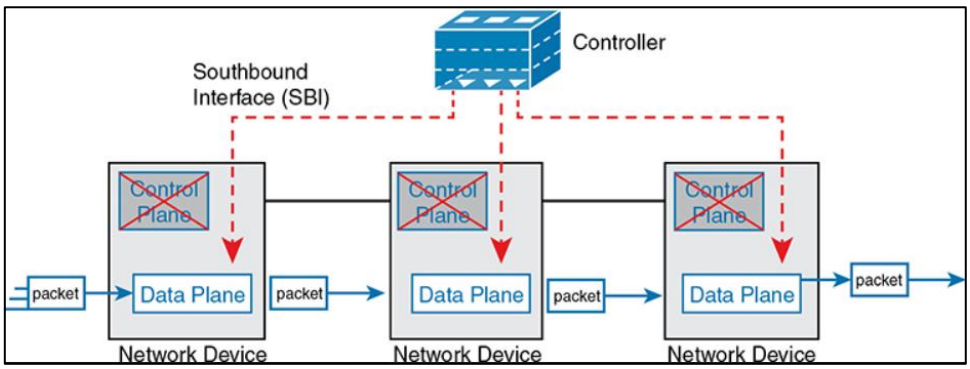
c) Capa de Infraestructura

La capa de infraestructura asegura la interacción entre los usuarios finales y los servidores implementados en la SDN, para que esto sea posible se necesitan de dispositivos que tengan compatibilidad con OpenFlow, dado que este protocolo es el encargado de brindar la comunicación con el controlador ODL. Para ello es necesario configurar los Open vSwitch, gracias a GNS3 y el stack de aplicaciones que contiene, permite hacer uso de la imagen ISO que brinda para conformar el escenario de la red híbrida SDN/MPLS.

Para comprender mejor el escenario que se ha planteado, en la Figura 17 se muestra el modelo utilizado en la topología, donde un controlador SDN centraliza todas las funciones del plano de control y que puede estar ubicado en cualquier lugar de la red que tenga accesibilidad IP a los dispositivos de la red. Este modelo muestra el modelo utilizado por las implementaciones originales de SDN basadas en el estándar de la industria llamado OpenFlow.

Figura 17

Plano de control centralizado y plano de datos distribuido



Fuente: (Wendell, 2020)

CAPÍTULO IV
IMPLEMENTACIÓN DE LA RED HÍBRIDA SDN/MPLS

Este capítulo se enfoca en el proceso de implementación de la red híbrida SDN/MPLS, estableciendo la comunicación entre la capa de infraestructura, la capa de control y la capa aplicación, para el acceso a los servicios de la DMZ por parte de los usuarios de las respectivas redes LAN, tomando en cuenta la ingeniería de tráfico y la diferenciación de servicios, en donde se priorice la mejor disponibilidad, QoS y escalabilidad.

4.1 Implementación de la Red Definida por Software

4.1.1 OpenDayLight

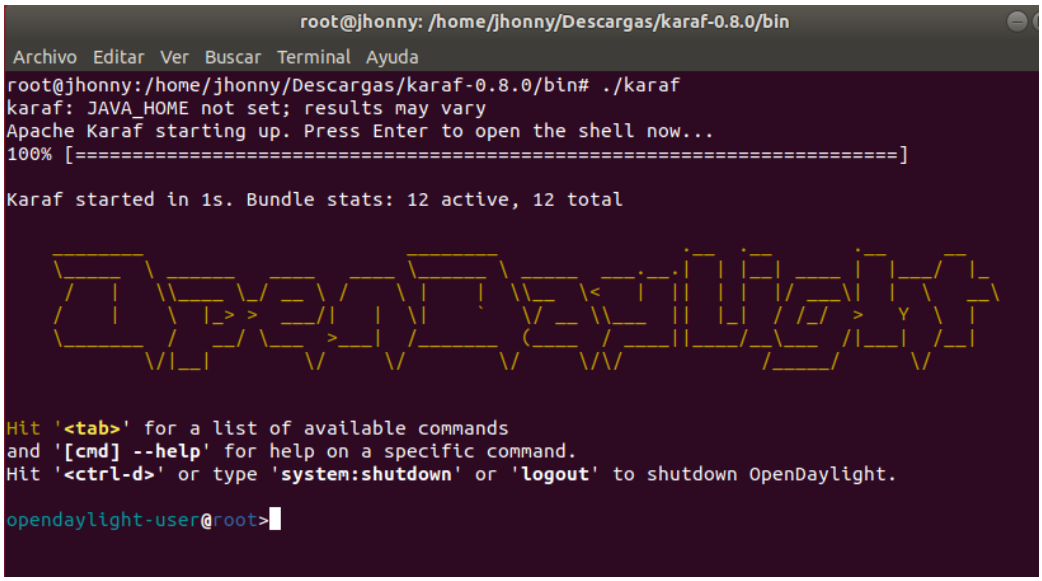
La parte más esencial en la implementación de una SDN empieza por la capa de control, puesto que esta es la parte esencial en este tipo de red, el cual se encuentra basado en el diseño planteado en la sección 3.3, donde el software escogido es OpenDayLight (ODL).

Cabe señalar que ODL es una plataforma abierta modular para personalizar y automatizar redes de cualquier tamaño, dado esto es que los beneficios son varios, entre ellos están: interoperabilidad de diferentes dispositivos físicos y virtuales, visibilidad continua de los flujos desde el origen hasta el destino, programabilidad para dar forma al comportamiento de la red de acuerdo con las necesidades del usuario, entre otros más (OpenDaylight Foundation, 2018).

Al completarse la instalación del controlador, proceso que se describe en detalle en el Anexo1, se dirige a la carpeta en donde se encuentra ubicado el programa ODL, para esto se hace uso del comando: **`cd /home/jhonny/Descargas/karaf-0.8.0/bin# ./karaf.`** De esta manera el controlador ODL se ejecuta normalmente y muestra una carátula con su nombre reflejado en la Figura 18, con esto ODL ya se encuentra operativo y listo para continuar con la instalación de los módulos.

Figura 18

Ejecución del controlador ODL

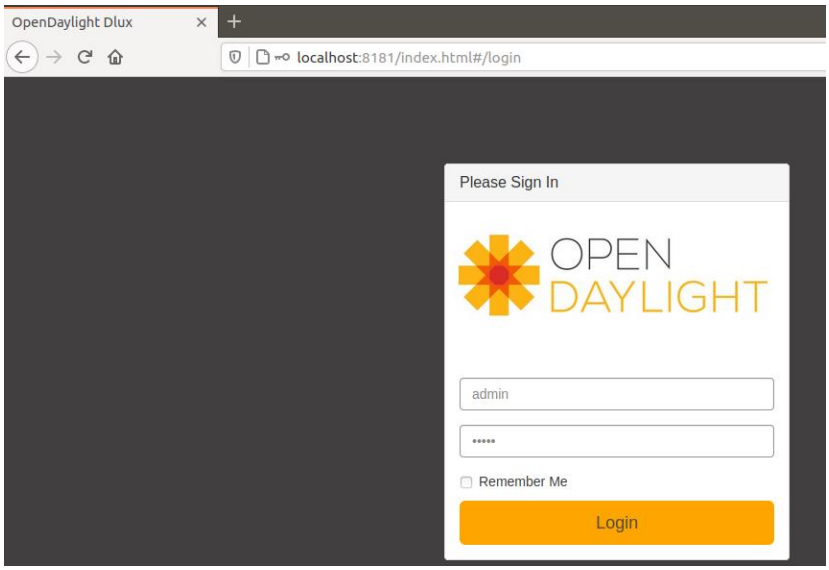


Ya en la consola que ODL ofrece, se instala de acuerdo con la Tabla 14 los respectivos módulos, para lo cual se ejecuta el siguiente comando: **`feature:install odl-restconf-all odl-openflowplugin-all odl-l2switch-all osl-mdsal-all odl-yangtools-common odl-dlux-all`**

A continuación, a través de un navegador web se ingresa a la URL **localhost:8181/index.html**, donde, tal como se muestra en la Figura 19, se tiene la página de inicio de ODL, en la cual para ingresar se debe digitar sus credenciales por defecto (*admin*).

Figura 19

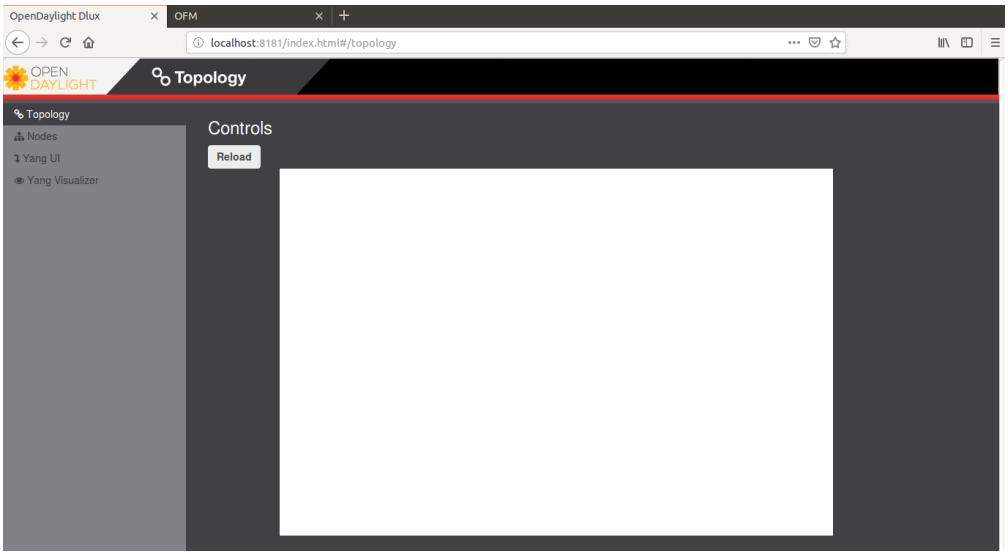
Página de inicio de ODL



La interfaz web tiene como objetivo visualizar la topología de los dispositivos de red que se conectan al controlador ODL, a partir de la información recibida por el puerto 6633. La Figura 20 muestra la interfaz web del controlador ODL una vez ingresado con las correspondientes credenciales, así el entorno se encontraría listo para enlazar los Open vSwitch y formar la red SDN.

Figura 20

Interfaz Web de ODL



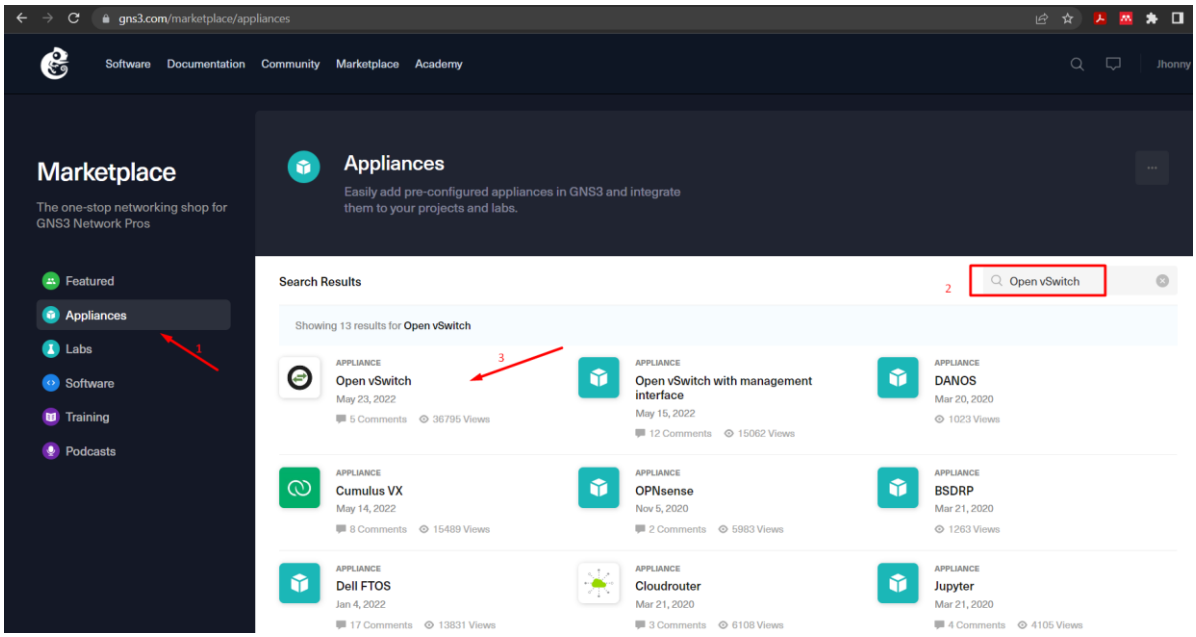
4.1.2 Open vSwitch

Para la capa de infraestructura de la SDN se hace uso de Open vSwitch como switch virtual multicapa para cumplir lo mencionado en el apartado c de la sección 3.3, este es implementado gracias al stack de aplicaciones que brinda GNS3 y que se explica paso a paso la integración de la aplicación en el Anexo 2.

Para realizar la instalación de Open vSwitch, se debe ingresar a la página web de GNS3 (https://www.gns3.com/), luego en la sección de Marketplace hay que dirigirse a la tienda de aplicaciones de GNS3. La Figura 21 muestra el procedimiento a seguir, primero se ingresa en la sección de Appliances, segundo en la barra de búsqueda se escribe Open vSwitch y finalmente se escoge la aplicación a descargar.

Figura 21

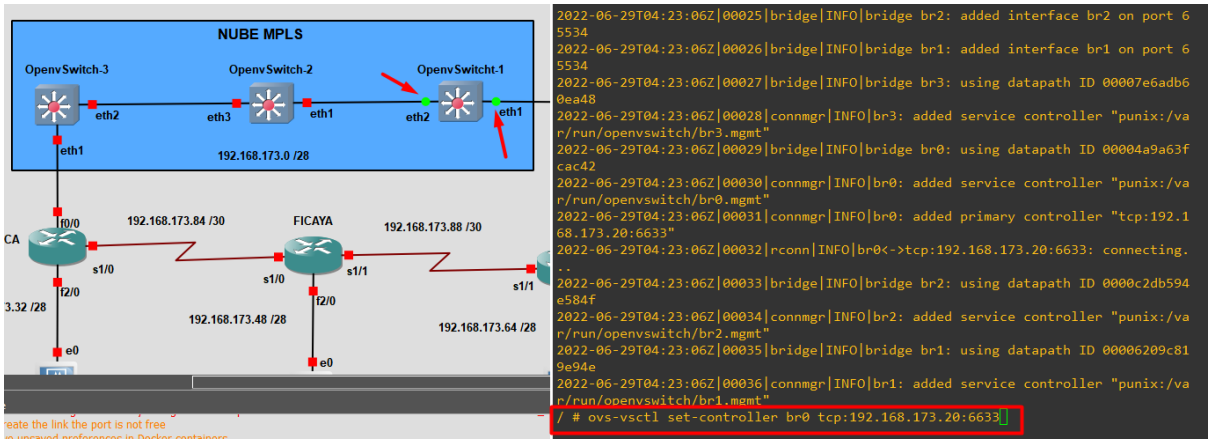
Descarga de Open vSwitch en la tienda de aplicaciones GNS3



Se aclara que este tipo de switch, en cuanto a recursos ya viene optimizado por GNS3 al momento de importar la aplicación. Lo siguiente es incluir el equipo en la interfaz de acuerdo con el diseño de la topología, una vez conectados los diferentes puertos se procede a enlazarlo con el controlador como se muestra en la Figura 22, para lo cual se hace uso del siguiente comando: **`ovs-vsctl set-controller br0 tcp:192.168.173.20:6633`**

Figura 22

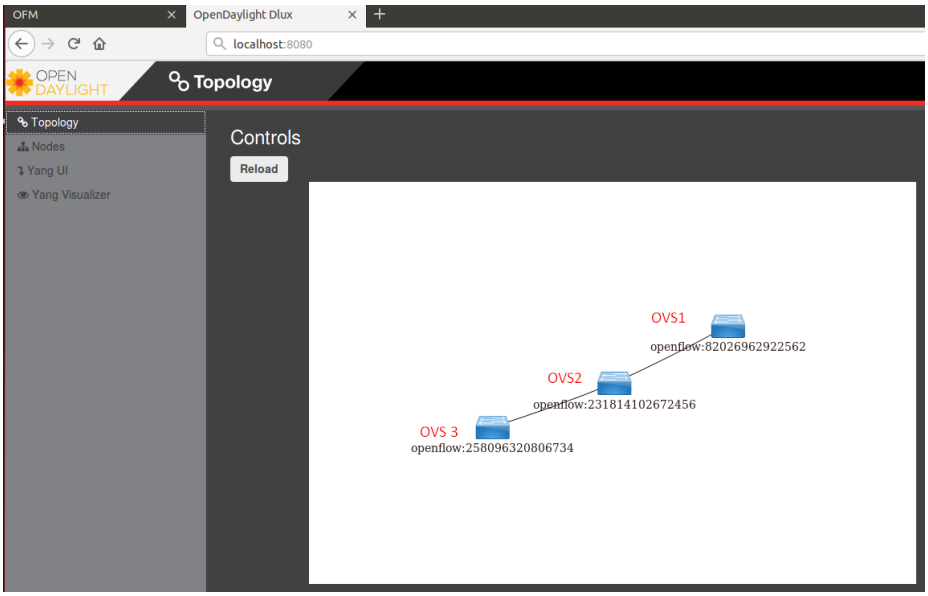
Establecimiento de Open vSwitch en el entorno de GNS3 y enlace con el controlador ODL



En la Figura 23 una vez aplicado el comando, en la interfaz web de ODL ya es posible visualizar a los equipos, y además cómo se encuentran establecidos sus enlaces dado que se crea una topología de la red, en este caso se trata de la nube MPLS.

Figura 23

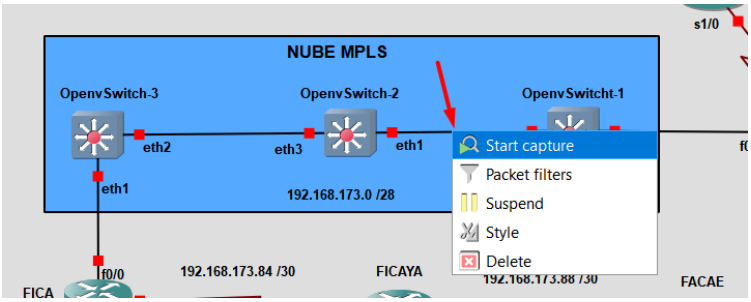
OVS enlazados con ODL



Con los OVS implementados y funcionando, se procede a la captura de paquetes OpenFlow, mediante el uso del sniffer Wireshark que es una de las herramientas que contiene GNS3, para lo cual primero se debe ubicar en el enlace que se desea capturar el tráfico en la red, luego con click derecho se despliega un pequeño menú y finalmente se selecciona *start capture* así como se muestra en la Figura 24.

Figura 24

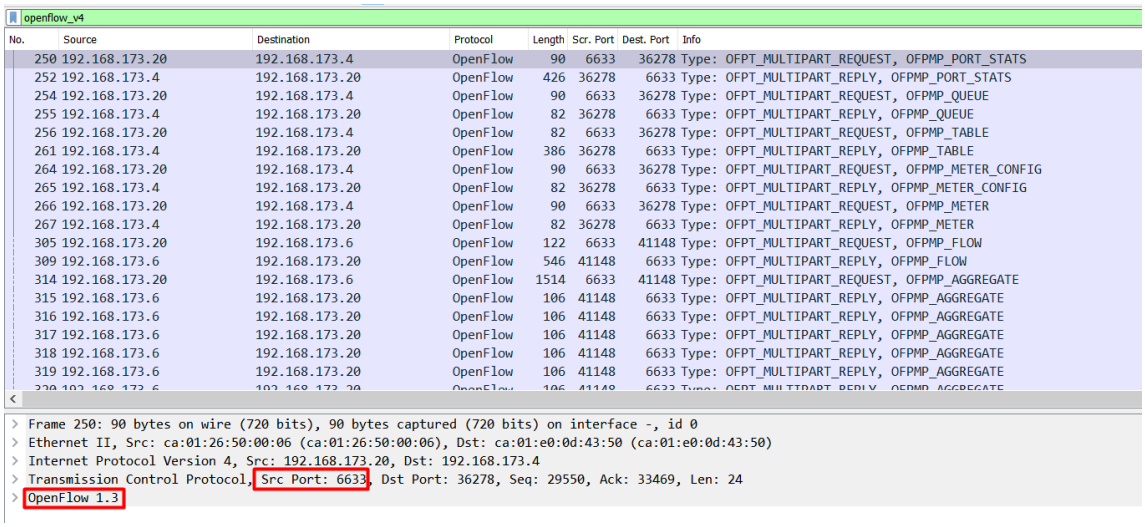
Selección del enlace para captura de tráfico con Wireshark



Al empezar la captura se puede visualizar todos los diferentes paquetes que viajan en la red, en este caso es necesario únicamente los paquetes OpenFlow, para lo cual se aplica el filtro **openflow_v4**. En la Figura 25 se muestra una captura de este tipo de paquetes con el filtro aplicado, remarcando que los OVS usan el puerto 6633 del controlador ODL y OpenFlow verión 1.3.

Figura 25

Captura de paquetes OpenFlow

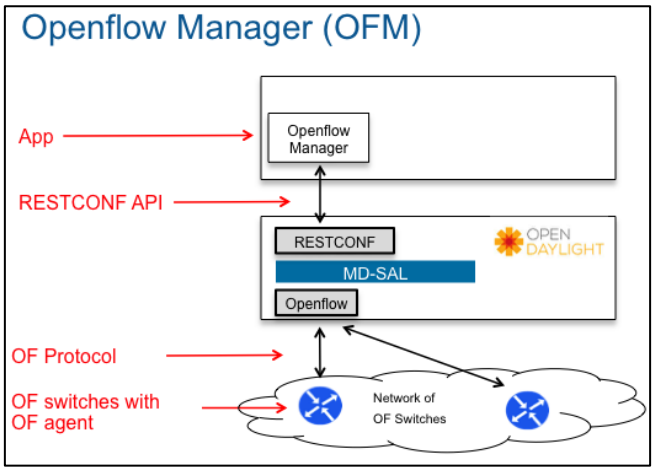


4.1.3 OpenFlow Manager

Adicional a las características ya descritas en la sección 4.1.1, ODL posee una aplicación llamada OpenFlow Manager (OFM) que permite la administración de la red OpenFlow. La Figura 26 muestra un diagrama de la interacción entre las diferentes capas SDN que posee esta aplicación, tomando en cuenta el controlador ODL como elemento central, en dirección sur (capa de control – capa infraestructura) utiliza diferentes protocolos (OpenFlow, ForCES, OPFlex) y en dirección norte (capa de control – capa aplicación) presenta una abstracción de la red utilizando en la práctica API REST comunes (CiscoDevNet, 2014).

Figura 26

Arquitectura OFM

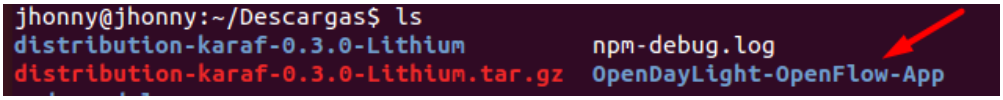


Fuente: (CiscoDevNet, 2014)

Para implementar esta aplicación, en la consola de línux con el comando: **`git clone`** **`https://github.com/CiscoDevNet/OpenDaylight-Openflow-App`** se importa el repositorio de GitHub en la carpeta que se desee, de preferencia en la misma que se encuentra el controlador ODL. Al terminar la importación, la carpeta con los archivos correspondientes estará disponible tal y como lo muestra la Figura 27.

Figura 27

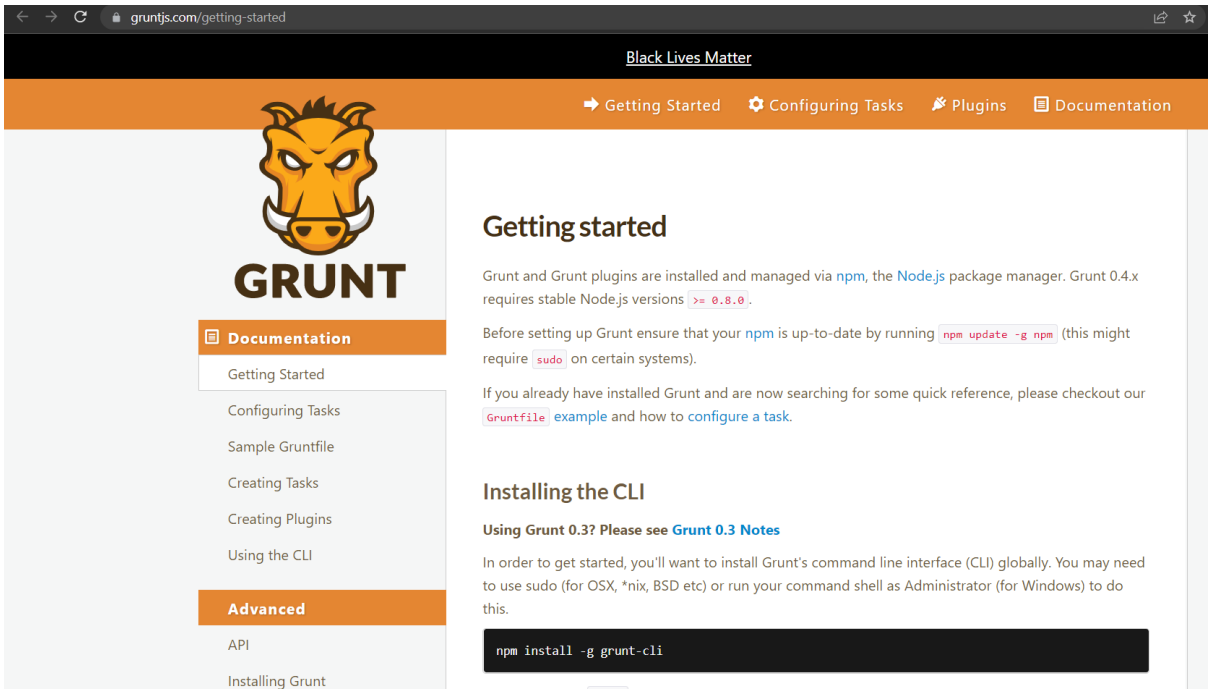
Repositorio de OFM importado



Prosiguiendo con la implementación de OFM, para la ejecución de la herramienta es necesario instalar el complemento **GRUNT** con el comando; **`npm install -g grunt-cli`**; el cual posee su propia página web https://gruntjs.com/getting-started que se muestra en la Figura 28, con el proceso paso a paso de cómo realizar una instalación correcta.

Figura 28

Página web GRUNT



Con el entorno preparado se ejecuta la aplicación OFM, primero se ingresa en la carpeta importada **`cd OpenDayLight-OpenFlow-APP`**, una vez dentro simplemente en la consola se escribe **`grunt`** y después de un corto período de tiempo OFM estaría operativo como lo muestra la Figura 29 a través del puerto 9000.

Figura 29

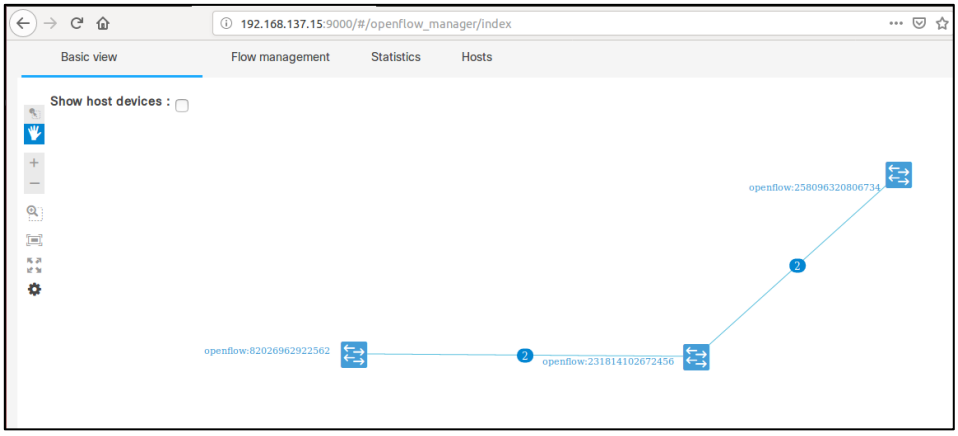
Ejecución de OFM

```
jhonny@jhonny:~/Descargas/OpenDayLight-OpenFlow-App$ grunt
Running "connect:def" (connect) task
Waiting forever...
Started connect web server on http://localhost:9000
```

Otro elemento para tomar en cuenta es que la interfaz de usuario se encuentra basada en HTML5/CSS7Javascript, lo cual significa que se trata de código abierto, permitiendo acceder a OFM desde cualquier navegador solo es necesario digitar la URL **192.168.137.15:9000** en la barra de navegación. La Figura 30 muestra la pantalla inicial una vez se enlazan con el controlador ODL los equipos OpenFlow, al igual que en la interfaz de ODL aquí se muestra la topología de la nube MPLS.

Figura 30

Panel principal OFM



4.2 Implementación de Servidores

Para realizar la implementación de los servidores, es necesario hacer uso de máquinas virtuales (VM) que alojen cada uno de los servicios de los cuales se van a encargar, en la Tabla 8 se mencionó el sistema operativo (SO) sobre el que opera cada servidor y qué servicio es el que debe cumplir, los cuales son Web, FTP y Streaming que se los detalla a continuación.

4.2.1 Servidor Web

El servidor web es el encargado de entregar el contenido de un sitio web al usuario, en otras palabras, es el software que permite a los usuarios o clientes que quieran ver una página web desde cualquier navegador. La comunicación establece mediante el Protocolo de

Transferencia de Hipertexto (HTTP) que trabaja sobre el puerto 80, y también con el Protocolo Seguro de Transferencia de Hipertexto que trabaja sobre el puerto 443.

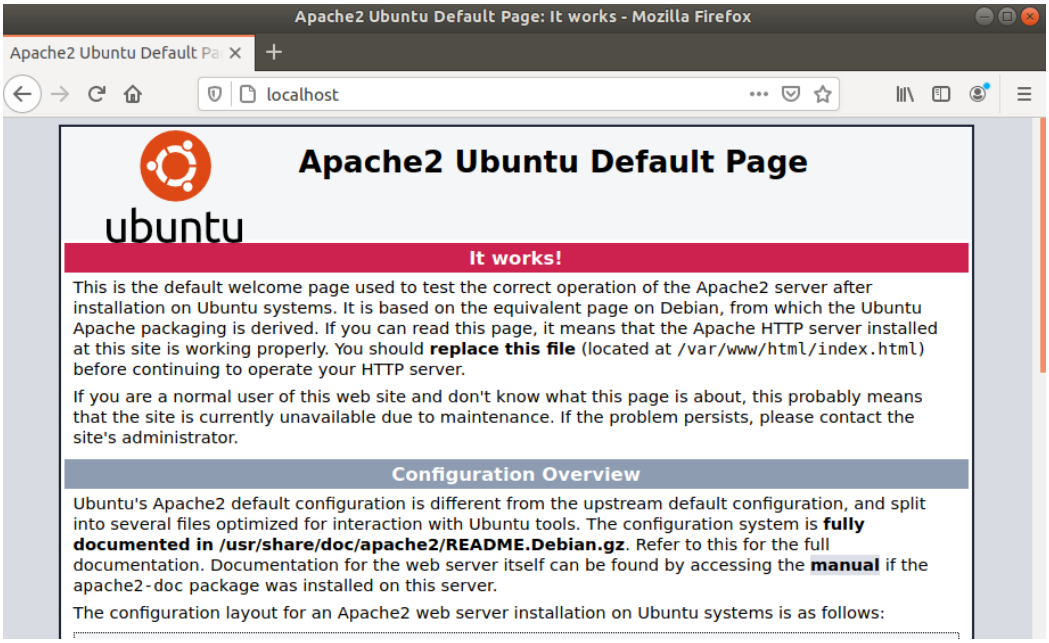
La diferencia entre ambos protocolos es básicamente que HTTPS incluye encriptación SSL (capa de sockets seguros) o TLS (seguridad de la capa de transporte) que es una versión más actualizada de SSL, esto con el fin de tener mayor confianza al momento de la transferencia de datos, mientras que HTTP lo realiza en texto plano sin ningún tipo de seguridad.

Existen diferentes tipos de servidores web, en este caso se hace uso del servidor Apache donde según (Rodríguez Ximena, 2019), es uno de los más populares utilizado para servidores web, teniendo como sus principales características el ser de código abierto y multiplataforma; este servidor se usa principalmente para servir páginas dinámicas o estáticas.

La instalación del servidor web se realiza mediante el comando: **`apt-get install apache2;`** una vez el proceso finaliza para comprobar su funcionamiento se ingresa como URL **localhost (IP del servidor)** para cargar la página web por defecto de Apache, así como lo muestra la Figura 31.

Figura 31

Página web por defecto



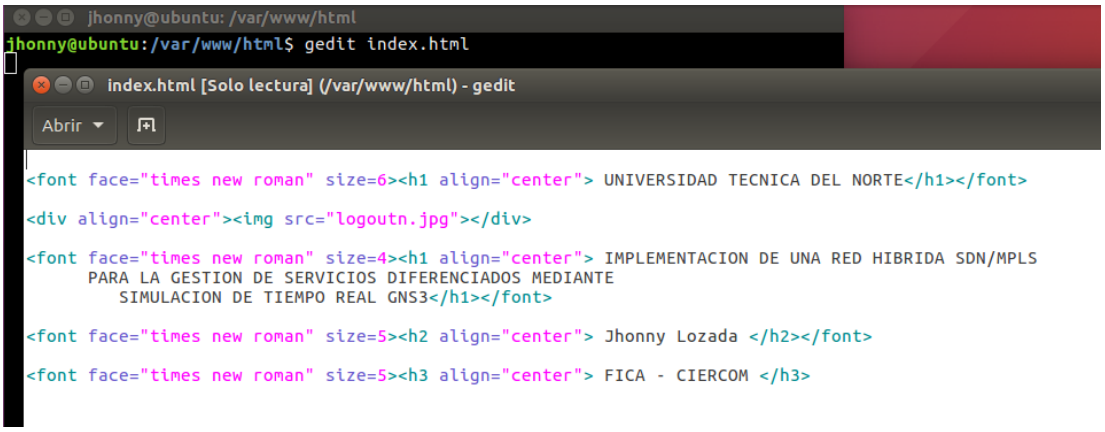
Continuando con el proceso, se procede a configurar la página web que se va a diseñar, en primer lugar se cambia el nombre del archivo **index.html** original para tenerlo de respaldo,

este proceso se realiza mediante el comando **`mv /var/www/html/index.htm /var/www/html/pindex.html`.**

Luego se procede a crear un nuevo archivo **index.html,** la Figura 32 muestra el proceso a seguir con los siguientes comandos: **`touch /var/www/html/index.html`** para crear el archivo y **`gedit /var/www/html/index.html`** para editar el archivo y configurar el sitio Web a través de lenguaje de programación HTML5.

Figura 32

Creación y edición del archivo index.html



Una finalizada con la programación de la página web, se vuelve a cargar la misma en el navegador, con todo esto completado, en la Figura 33 se muestra una prueba de funcionamiento si todo está según lo que se desea, el proceso estaría terminado. La instalación del servidor Web se detalla paso a paso en el Anexo 3.1

Figura 33

Servidor web en funcionamiento

4.2.2 Servidor FTP

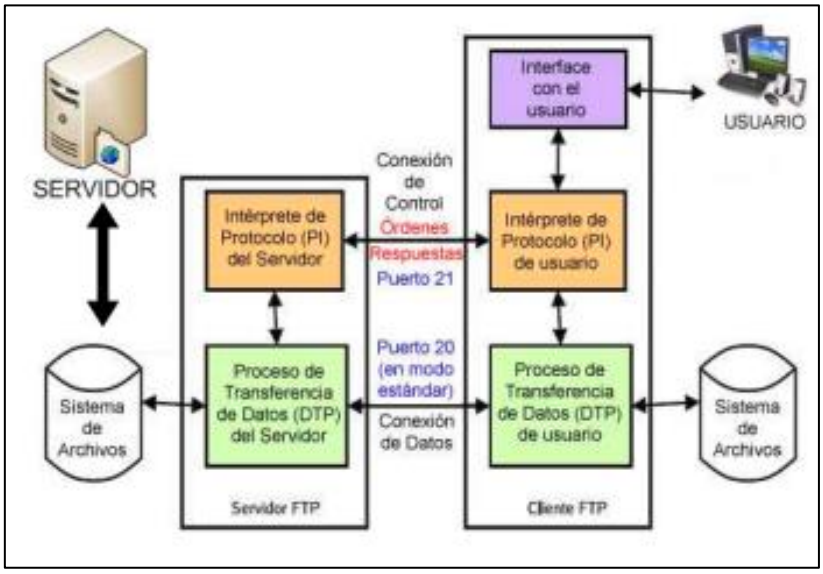
El servicio FTP (Protocolo de Transferencia de Archivos) se utiliza para el envío y obtención de archivos entre dos equipos conectados a la misma red de tipo TCP/IP, trabajando sobre un modelo cliente-servidor como se muestra en la Figura 34 y explicado a continuación.

Usa principalmente dos puertos para conectarse, el puerto 20 para la transferencia de datos y el puerto 21 para control, cabe señalar que todos los datos que se envían incluyendo el nombre de usuario y contraseña de la cuenta FTP, son enviados en texto plano, lo cual, vuelve vulnerable los datos si alguien logra interrumpir la conexión para hacerse con la información; una vez el cliente establece el enlace con el servidor ya puede recibir y transferir datos.

Si se necesita una conexión segura FTP, es necesario optar por la aplicación SFTP (Protocolo de Transferencia Segura de Archivos) que usa el puerto 22, a través de SSH (Secure Shell) para poder cifrar el contenido que se transfiere; si alguien se llega a hacer con la información transferida, ésta se encontrará cifrada.

Figura 34

Funcionamiento protocolo FTP

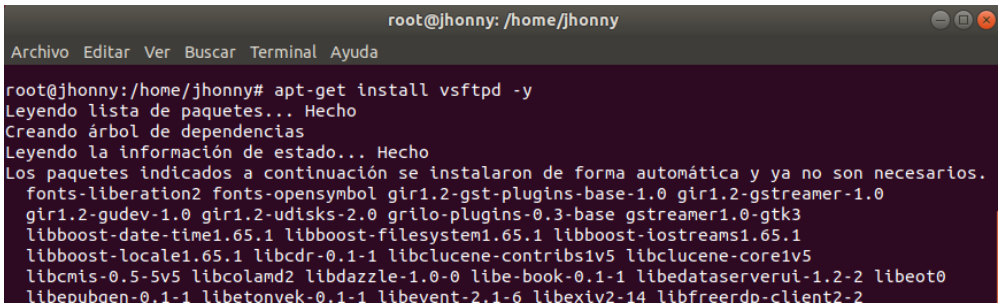


Fuente: (Quesada, s/f)

La instalación del servidor FTP se realiza mediante la herramienta **VSFTPD,** en la Figura 35 se muestra su instalación a través del comando: **`apt-get install vsftpd -y`**

Figura 35

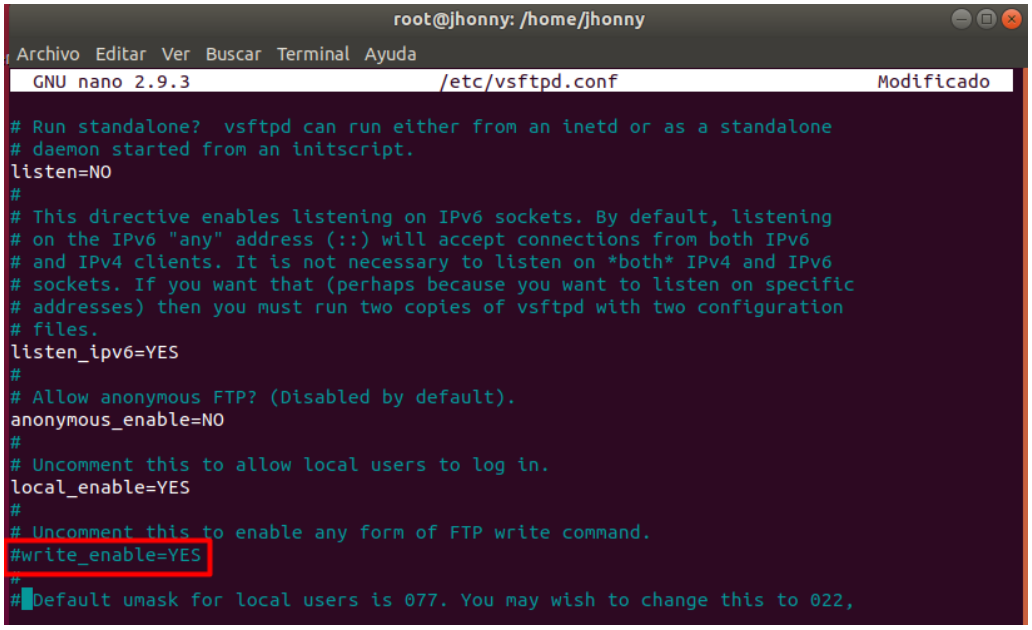
Instalación de Vsftpd para el servidor FTP



Finalizada la instalación se procede a editar el archivo de configuración Vsftpd, para lo cual se hace uso del comando **`nano /etc/vsftpd.conf`**, tal y como se muestra en la Figura 36 se desmarca la línea **`write_enable=YES`** y se guardan los cambios al momento de salir de la edición.

Figura 36

Escritura habilitada en Vsftpd



Se reinicia el servicio con el comando **`service vsftpd restart`** para que tomen efecto los cambios implementados, y luego a través del comando **`service vsftpd status`** se verifica que el servicio esté corriendo normalmente como se muestra en la Figura 37.

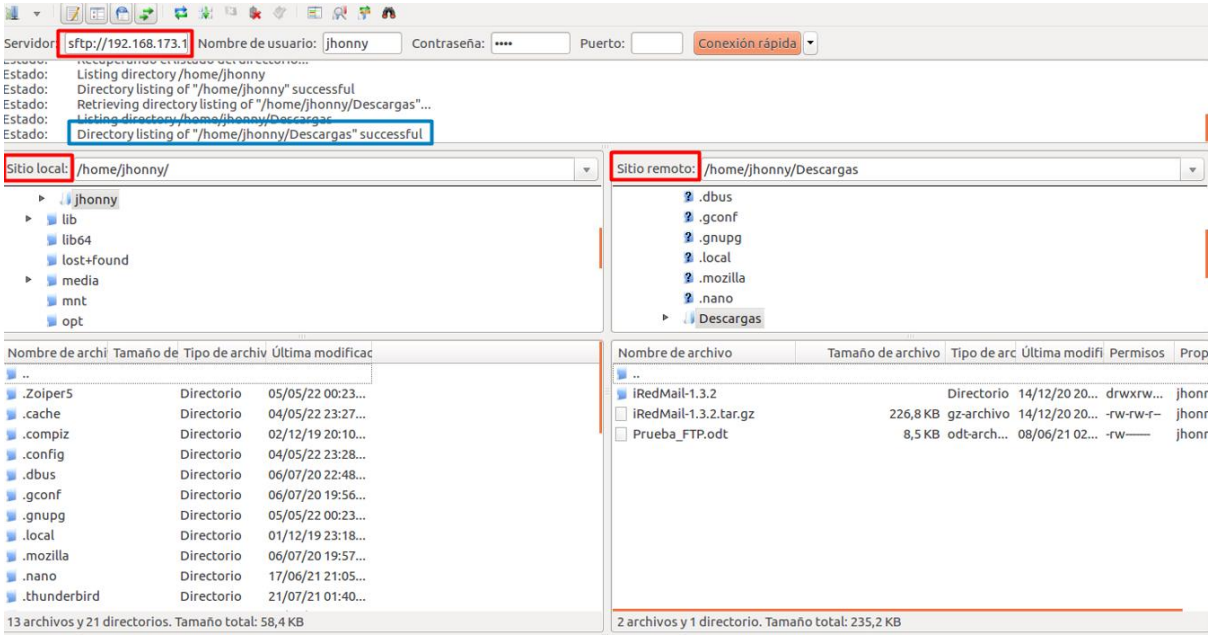
Figura 37

Vsftpd corriendo normalmente

Una vez configurado el servidor y los clientes de acuerdo con lo desarrollado en el Anexo 3.2; se procede con una prueba de funcionamiento. En la Figura 38 haciendo uso de la aplicación FileZilla se ingresa con las credenciales del servidor (IP, usuario, contraseña), luego se da al botón de conexión rápida, si las credenciales son aceptadas aparece un mensaje *succesfull* (recuadro azul) y finalmente ya es posible realizar la transferencia de archivos entre el sitio local y el sitio remoto (recuadro rojo).

Figura 38

Prueba de funcionamiento protocolo FTP



4.2.3 Servidor Streaming

Un servidor de streaming se trata de realizar la transmisión de audio y video en tiempo real, en otras palabras, es la manera de ver o escuchar contenido multimedia sin tener la necesidad de descargar archivos de audio o video al ordenador. El contenido en el streaming

se visualiza siempre de forma online, aunque también se puede en vivo para tener una interacción directa con el usuario, sin embargo, con Streama no se realizan transmisiones en vivo.

Streama trabaja sobre el Protocolo de Control de Transmisión (TCP), este es uno de los protocolos principales en redes TCP/IP debido a que es orientado a la conexión y necesita el apretón de manos para determinar comunicaciones de principio a fin; solo cuando la conexión es establecida se pueden enviar los datos de manera bidireccional. Usa el puerto TCP 8080 (SubNet, 2016) que es denominado HTTP alternativo, este se encarga de garantizar la entrega de los paquetes en el mismo orden en el que han sido enviados, donde a diferencia del puerto UDP 8080 este no garantiza la comunicación como TCP, volviéndose un servicio poco fidedigno y los datagramas pueden llegar en cualquier orden o incluso estos pueden llegar a perderse.

Previamente instalado Java 8, se crea la carpeta que contendrá los archivos que serán necesarios más adelante, como se observa en la Figura 39 con el comando **`mkdir`** se crea un directorio, en este caso se crean dos con los siguientes comandos: **`mkdir /data`** y **`mkdir /data/Streama`**

Figura 39

Creación de directorios

```
root@ubuntu:/home/jhonny# mkdir /data
root@ubuntu:/home/jhonny# mkdir /data/streama
```

Siguiendo el proceso, con el comando **`cd /data/streama`** se ingresa al directorio que se acabó de crear, desde el repositorio de Github con la sentencia **`wget`** **https://github.com/streamaserver/streama/releases/download/v1.6.0-RC7/streama-1.6.0-RC7.war**, se descarga Streama así como lo muestra la Figura 40.

Figura 40

Descarga de Streama desde el repositorio de Github

```
root@ubuntu:/data/streama# wget https://github.com/streamaserver/streama/releases/download/v1.6.0-RC7/streama-1.6.0-RC7.war
--2022-05-08 20:00:19--  https://github.com/streamaserver/streama/releases/download/v1.6.0-RC7/streama-1.6.0-RC7.war
Resolviendo github.com (github.com)... 140.82.112.4
Conectando con github.com (github.com)[140.82.112.4]:443... conectado.
Petición HTTP enviada, esperando respuesta... 302 Found
Ubicación: https://objects.githubusercontent.com/github-production-release-asset-2e65be/39890823/d9e19c5a-64bd-11e8-9e89-9b
%2F20220509%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20220509T010020Z&X-Amz-Expires=300&X-Amz-Signature=b8b38412080b822c1
=0&key_id=0&repo_id=39890823&response-content-disposition=attachment%3B%20filename%3Dstreama-1.6.0-RC7.war&response-content
--2022-05-08 20:00:20--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/39890823/d9e19c5a-64b
NJYAX4CSVEH53A%2F20220509%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20220509T010020Z&X-Amz-Expires=300&X-Amz-Signature=b8b
=host&actor_id=0&key_id=0&repo_id=39890823&response-content-disposition=attachment%3B%20filename%3Dstreama-1.6.0-RC7.war&re
Resolviendo objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.
Conectando con objects.githubusercontent.com (objects.githubusercontent.com)[185.199.108.133]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 94360806 (90M) [application/octet-stream]
Grabando a: “streama-1.6.0-RC7.war”

streama-1.6.0-RC7.war                          19%[====================>
```

Con el archivo descargado se debe convertir en ejecutable, esto se hace mediante el comando **`chmod +x streama-1.6.0-RC7.war`**, finalmente se ejecuta la aplicación con el comando **`./streama-1.6.0-RC7.war`**; todo este proceso se muestra en la Figura 41.

Figura 41

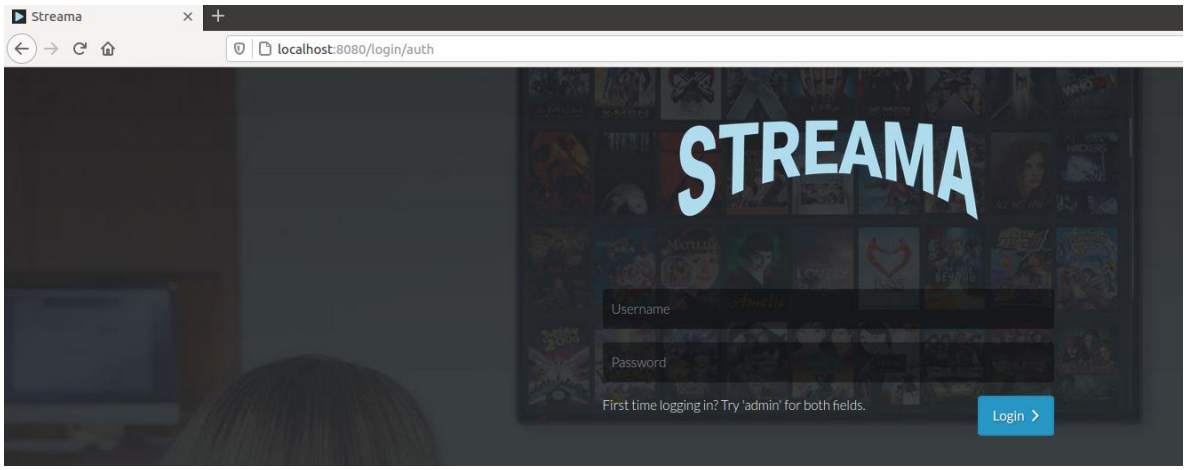
Ejecución de la aplicación Streama

```
root@ubuntu:/data/streama# chmod +x streama-1.6.0-RC7.war
root@ubuntu:/data/streama# ./streama-1.6.0-RC7.war
```

La implementación de Streama se realiza a detalle en el Anexo 4; una vez finalizada la instalación y puesta a punto del servidor se muestra una interfaz como en la Figura 42, en la cual, para ingresar en la plataforma se digita la URL **localhost (IP servidor:8080),** en donde lo siguiente es proceder con la creación de usuarios y carga de contenido para ser compartido.

Figura 42

Servidor de Streaming en funcionamiento



4.2.4 Servidor VoIP

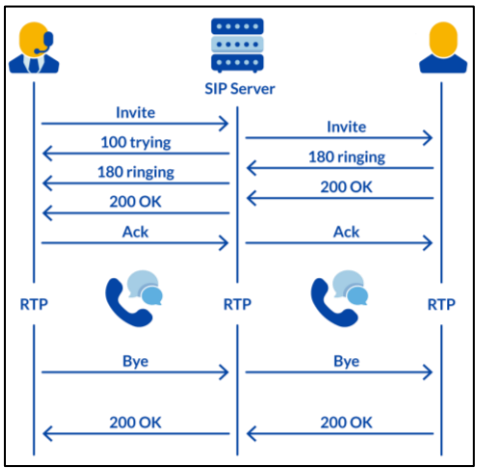
Un servidor VoIP (Voz sobre Protocolo Internet), es una tecnología relativamente nueva que permite la transmisión de voz sobre una red basada en IP, esto supone una alternativa para ahorrar costos en telefonía fija a las empresas o instituciones, debido que con la red IP propia se gestionan los servicios de telefonía.

Su funcionamiento se encuentra basado en dos protocolos: *Protocolo de Inicio de Sesión (SIP)* usado para iniciar, modificar y finalizar una sesión interactiva de usuario; los clientes SIP y centrales de conmutación (PBX) hacen uso del puerto 5060 UDP y TCP para conectarse a servidores SIP (Microsoft, 2022). El siguiente es el *Protocolo de Transporte en Tiempo Real (RTP),* el cual define un formato de paquetes estándar para enviar audio y video sobre una red específica, no requiere un puerto TCP o UDP estático con el que comunicarse, por eso hace uso de puertos dinámicos que van entre 1024 y 65535 UDP (Microsoft, 2022). En la Figura 43 se muestra a detalle el proceso paso a paso de estos protocolos al momento de establecer y finalizar una llamada entre dos estaciones, el cual se explica a continuación.

- Invite: Establece una sesión entre los agentes de usuario y contiene información sobre la llamada, los usuarios llamados y los medios que se intercambiarán.
- 100 Trying: Indica que el Invite ha sido recibido correctamente por el servidor y se está tomando una acción no detallada para tratarlo, con este mensaje se paran las retransmisiones de los mensajes Invite por parte del origen.
- 180 Ringing: Este mensaje origina que quien esté realizando la llamada en la central de origen comience a escuchar un ringback (tono de llamada) en el auricular.
- 200 OK: Indica que la petición se ha gestionado correctamente, aquí hay que tomar en cuenta el encabezado CSeq para conocer a qué mensaje de origen está contestando el 200 OK, si es INVITE indica que la llamada se ha contestado correctamente, en cambio si fuese CANCEL es para cancelar una llamada antes de que haya sido contestado el 200 OK, sería la confirmación de cancelación.
- Bye: Termina una sesión entre dos usuarios y puede ser enviado por cualquier usuario.

Figura 43

Protocolo SIP y RTP en una llamada VoIP



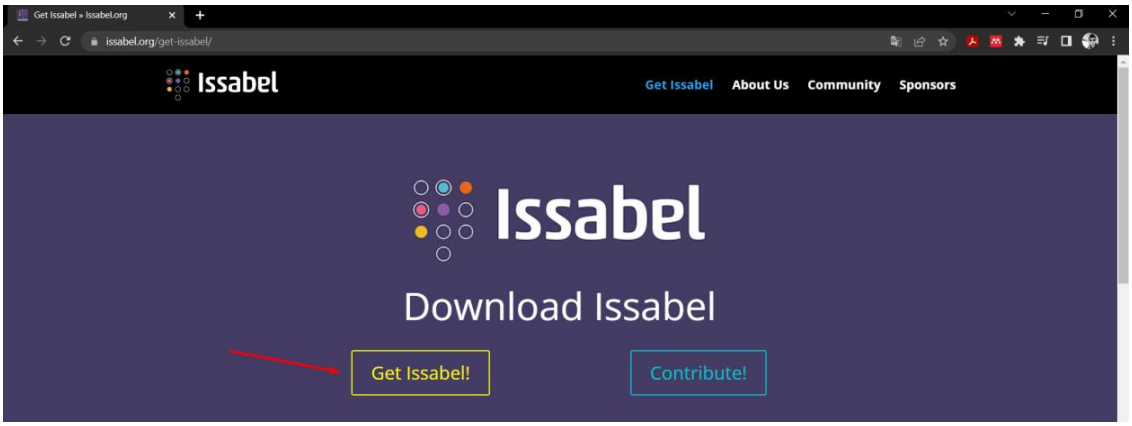
Fuente: (Bai, 2019)

Issabel cumple con el proceso mencionado anteriormente, dado que desciende del pionero en servicios de VoIP llamado Elastix, pero que ahora tiene muchas más funciones que no solo se centran en VoIP, permitiendo servicios de correo o fax. Es un plataforma que se usa con frecuencia en empresas o instituciones, la misma se basa en Centos 7. Todo el proceso de instalación y creación de extensiones se detalla en el Anexo 5.

En este sentido, el primer paso dentro de la instalación consiste en ingresar a la URL https://www.issabel.org/get-issabel/, la Figura 44 muestra el sitio oficial de Issabel. Lo siguiente es seleccionar la opción **Get Issabel** (señalada con la flecha roja), donde empezará la descarga del archivo ISO; este archivo ISO posteriormente se instala en una VM bajo la plataforma de VirtualBox.

Figura 44

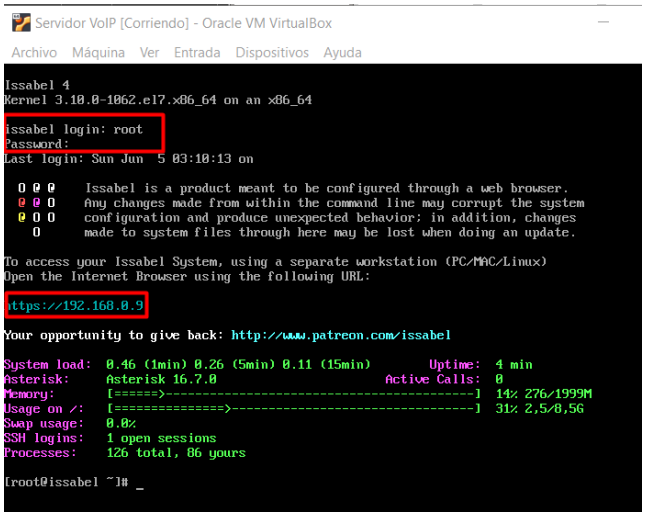
Sitio Oficial de Issabel



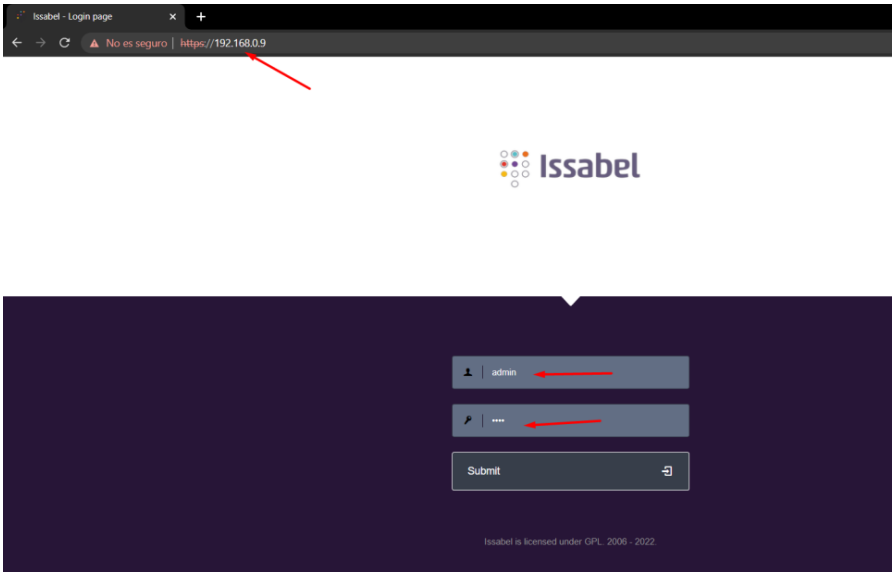
Una vez completada la instalación, se inicia Issabel mostrando su interfaz en la Figura 45 a, en donde al iniciar con la contraseña de root aparece la URL con la que se debe ingresar en el navegador. Al ingresar dicha la URL en la barra de navegación, se accede a la página de inicio de Issabel mostrada en la Figura 45 b, lo siguiente es digitar las credenciales respectivas; en usuario se coloca admin y el password es la contraseña de root.

Figura 45

a) Issabel en ejecución



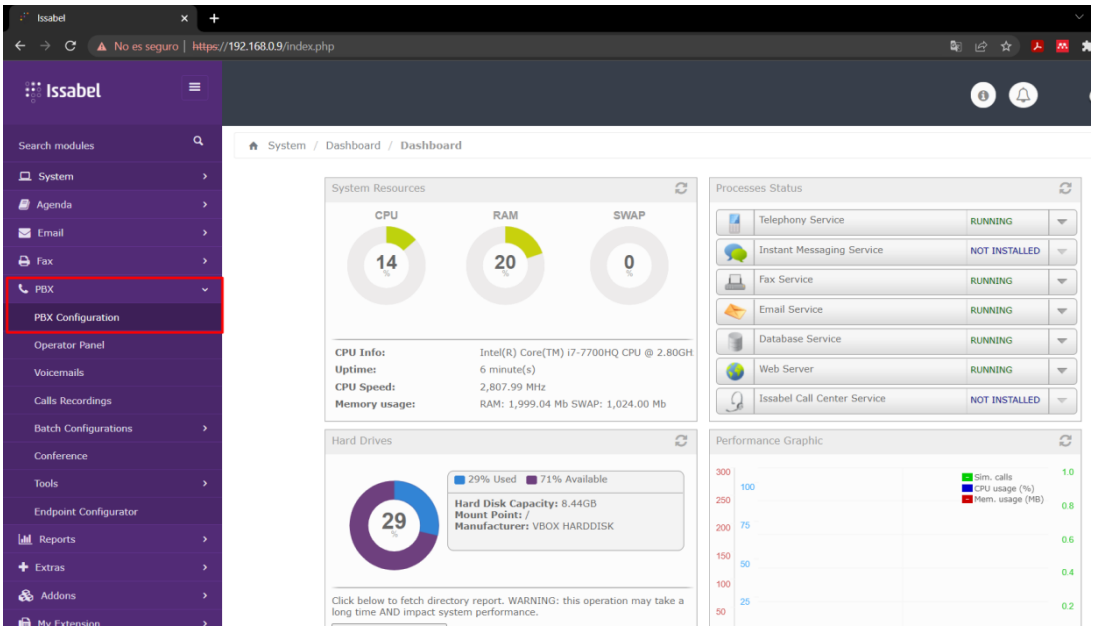
b) Página de inicio de Issabel



Al ingresar, en la Figura 46 se muestra la interfaz para realizar las respectivas configuraciones y el estado actual de la CPU y RAM del sistema. Para crear las extensiones se selecciona el apartado PBX y se escoge la opción Configuración PBX resaltado en el recuadro rojo.

Figura 46

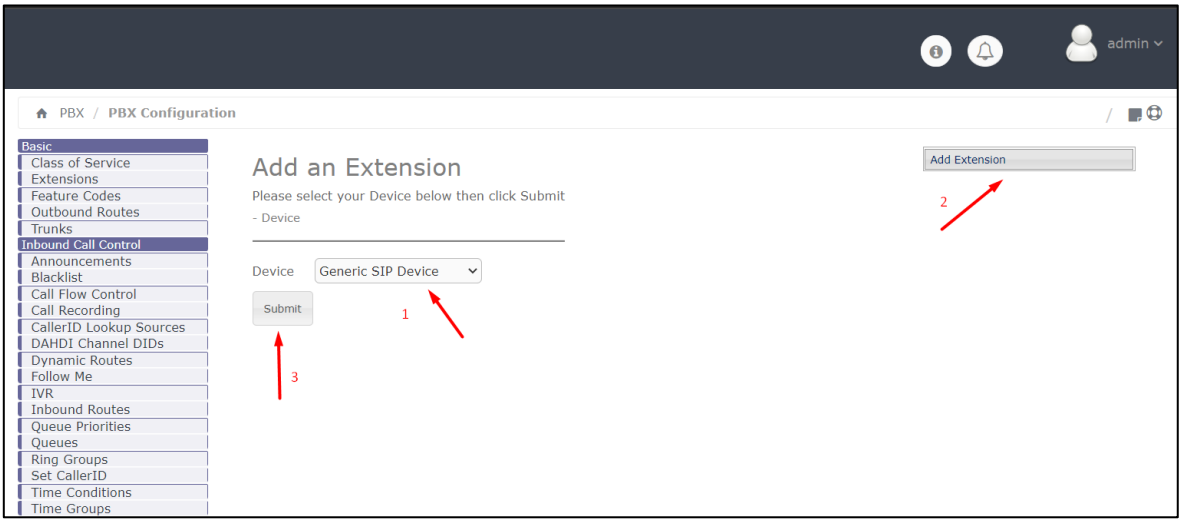
Interfaz de configuraciones de Issabel



Dentro de la configuración de PBX, en la Figura 47 se muestra el proceso a seguir; primero se selecciona el tipo SIP, segundo añadir extensión y tercero submit para aceptar la creación de esta.

Figura 47

Creación de una extensión SIP



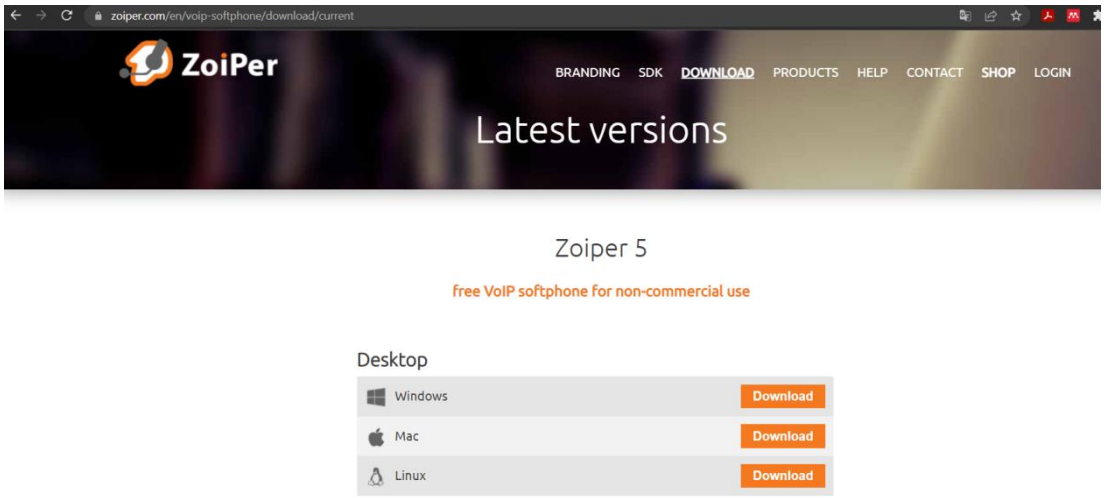
Para comprobar el funcionamiento del servidor Issabel, es necesario hacer uso de un softphone, en este caso se usa ZoiPer en su versión 5 dado que se trata de software libre, el cual posee algunas ventajas como lo menciona (Zoiper.com, 2019), entre las principales se encuentran: Ser multiplataforma (Windows, Linux, macOS), posee aplicación para

smartphones, interfaz dinámica para los usuarios, tiene consumo mínimo de memoria RAM y CPU

Como primer paso, se descarga el intalador de la aplicación de acuerdo al SO desde la URL https://www.zoiper.com/en/voip-softphone/download/current, en la Figura 48 se muestra la página oficial para descargar la aplicación Zoiper.

Figura 48

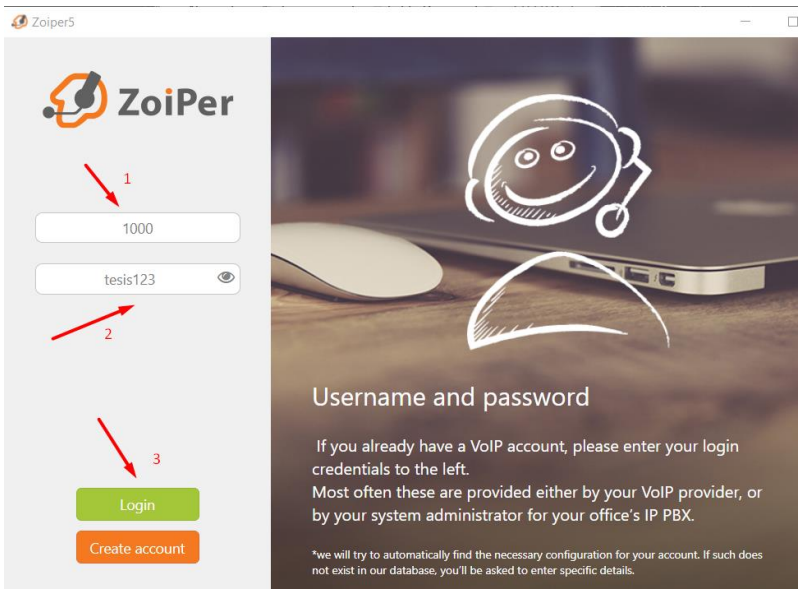
Página oficial de descarga de Zoiper



Una vez instalada la aplicación se muestra una interfaz como la Figura 49, además en ella se detalla el procedimiento a seguir; primero se debe digitar el número de extensión, segundo la contraseña de la extensión y tercero se da click en Login.

Figura 49

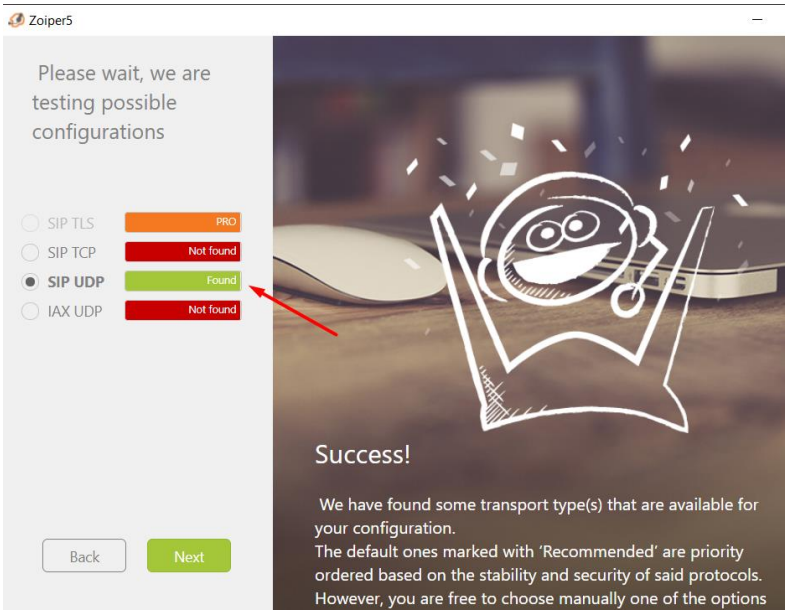
Ingreso de credenciales de la extensión SIP



Con todo lo anterior establecido, Zoiper añadirá con éxito la extensión SIP como lo muestra la Figura 50 y de esta manera la estación está lista para realizar y recibir llamadas de otras extensiones de la red.

Figura 50

Extensión SIP registrada con éxito en Zoiper



4.3 Priorización de tráfico y marcaje de servicios

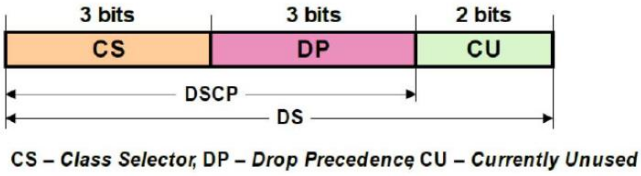
Para aplicar QoS a la red es necesario realizar el marcaje de los servicios, para lo cual se escoge un modelo de servicio, en este caso se utiliza el modelo de Diferenciación de Servicios (DiffServ) explicado en la sección 2.5; al ser un modelo basado en uso de múltiples clases especificadas en cada paquete, permite satisfacer diferentes requerimientos de QoS.

Una vez seleccionado el modelo DiffServ, el marcaje de los paquetes se realiza mediante DiffServ Code Point (DSCP), el cual se encarga de clasificar, gestionar y proporcionar QoS en una red IP, hace uso del campo de servicios diferenciados (DS) de 6 bits en el encabezado IP para fines de clasificación de paquetes.

La Figura 51 muestra la estructura del campo DS; para DSCP se divide en dos partes que contienen 3 bits cada una de ellas: la primera se denomina selección de clase (CS) y la segunda eliminación de precedencia (DP), por último, para completar el campo DS se encuentra una sección de 2 bits que actualmente no se usa.

Figura 51

Estructura del campo DS

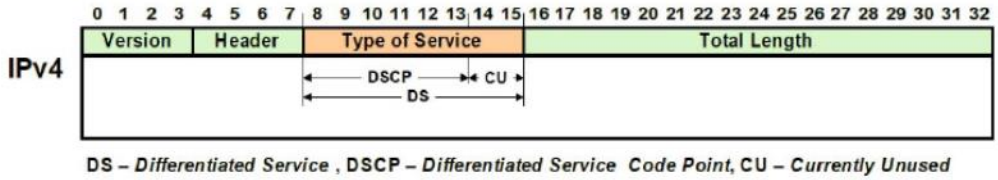


Fuente: (Omiotek, 2012)

Una vez detallado el campo DS, la Figura 52 muestra la estructura de un paquete IPv4 con el campo DS incluido en la cabecera o header, con el objetivo de diferenciar el tipo de servicio que viaja a través de la red.

Figura 52

Campo DS en la cabecera de un paquete IP



Fuente: (Omiotek, 2012)

Tomando en cuenta lo anterior, se establece en la Tabla 16 la clasificación de los servicios, que son: VoIP, Web, FTP y Streaming; en la misma se detalla la prioridad, aplicación, clase, puertos de operación y por último la codificación DSCP correspondiente.

Tabla 16

Clasificación de servicios mediante DSCP

PRIORIDAD	APLICACIÓN	CLASE	PUERTOS	DSCP
CRÍTICA	Transmisión de Voz	VOZ	10000 – 20000	EF
	Señalización	SIGNA	5060	CS3
	SignaVideo	Streaming	8080	AF43
ALTA	FTP	FTP	20,21,22	AF33
MEDIA	WEB	HTTP / HTTPS	80. 443	AF23

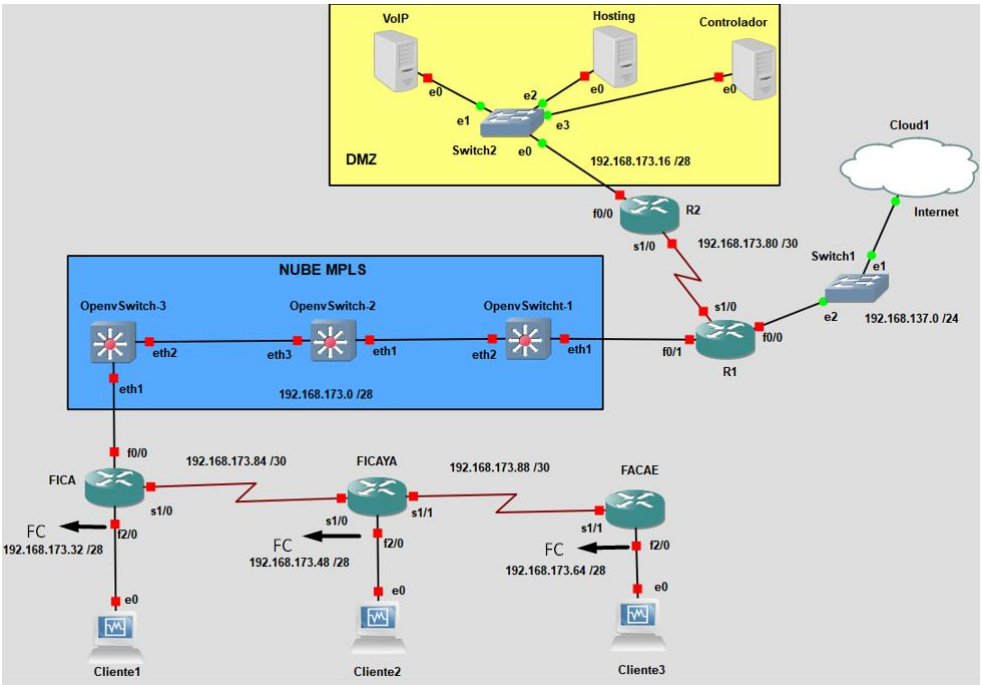
4.4 Frontera de confianza

La frontera de confianza o también denominada Trust Boundary, es el perímetro en el cual la red confía y respeta el marcaje que se ha realizado en un equipo dentro o sobre este perímetro, algo a remarcar es que una frontera de confianza se aplica lo más cerca de la fuente

de tráfico. La Figura 53 muestra la ubicación de la frontera de confianza en la topología de red, estas son las interfaces f2/0 de las respectivas redes LAN.

Figura 53

Designación de frontera de confianza



Para implementar las políticas de QoS se hace uso de *"interfaz de línea de comando (CLI) de calidad de servicio (QoS) modular"* (MQC), que se encuentra diseñado para simplificar la configuración de QoS en routers y switches, mediante la definición de una sintaxis de comandos y un conjunto final de comportamientos de QoS (CISCO, 2009). El MQC requiere de tres principales pasos.

a) Definir las clases de servicios mediante la clasificación de paquetes, usando ACL's para poder clasificar el tráfico de la red de datos.

b) Establecer políticas para las clases anteriormente definidas, mediante la elaboración de las diferentes políticas de calidad de servicio.

c) Aplicar las políticas de QoS dependiendo de la dirección del tráfico, ya sea de salida o entrada en una interfaz.

4.4.1. Aplicación de frontera de confianza

Para empezar, la Figura 54 muestra la configuración aplicada en el router FICA para el tráfico proveniente de la red LAN y detalla la secuencia de comandos a seguir, empezando con

la creación de las respectivas ACL's de tipo extendida, no con el objetivo de restringir el tráfico, sino más bien diferenciar el mismo que está pasando por la red de acuerdo con el protocolo y los puertos que se maneja en cada servicio.

Figura 54

Establecimiento de ACL´s para cada servicio en router FICA

```
FICA#configure terminal
Enter configuration commands, one per line.  End with CNTL
FICA(config)#ip access-list extend VOZ
FICA(config-ext-nacl)#permit udp any any range 10000 20000
FICA(config-ext-nacl)#permit udp any any range 40000 50000
FICA(config-ext-nacl)#exit
FICA(config)#ip access-list extend SignaVOip
FICA(config-ext-nacl)#permit udp any any eq 5060
FICA(config-ext-nacl)#exit
FICA(config)#ip access-list extend SignaSTREAM
FICA(config-ext-nacl)#permit udp any any eq 8080
FICA(config-ext-nacl)#permit tcp any any eq 8080
FICA(config-ext-nacl)#exit
FICA(config)#ip access-list extend FTP
FICA(config-ext-nacl)#permit tcp any any eq 20
FICA(config-ext-nacl)#permit tcp any any eq 21
FICA(config-ext-nacl)#permit tcp any any eq 22
FICA(config-ext-nacl)#exit
FICA(config)#ip access-list extend WEB
FICA(config-ext-nacl)#permit tcp any any eq 80
FICA(config-ext-nacl)#permit tcp any any eq 443
FICA(config-ext-nacl)#end
```

La Figura 55 muestra la configuración de las clases, donde es mejor nombrarlas de acuerdo con el servicio al que será destinada cada una. Se procede a crear las clases mediante el comando **`class-map`**, luego se asocian las respectivas ACL's que fueron configuradas anteriormente con el comando **`match access-group`**, para de esta manera continuar con la configuración de las políticas QoS.

Figura 55

Creación de clases y establecimiento de ACL´s para cada servicio

```
FICA#configure terminal
Enter configuration commands, one per line.  End with
FICA(config)#class-map VOZ
FICA(config-cmap)#match access-group name VOZ
FICA(config-cmap)#class-map SignaVOip
FICA(config-cmap)#match access-group name SignaVOip
FICA(config-cmap)#class-map SignaSTREAM
FICA(config-cmap)#match access-group name SignaSTREAM
FICA(config-cmap)#class-map FTP
FICA(config-cmap)#match access-group name FTP
FICA(config-cmap)#class-map WEB
FICA(config-cmap)#match access-group name WEB
FICA(config-cmap)#end
```

Siguiendo con el proceso, la Figura 56 muestra paso a paso las sentencias correspondientes que se hacen uso para establecer la política de tráfico, empezando con la política que englobará a las clases creadas previamente mediante el comando **`policy-map`**, se hace uso del código DSCP planteado en la Tabla 10 para cada servicio.

Figura 56

Configuración y establecimiento de la política de tráfico

```
FICA#configure terminal
Enter configuration commands, one per
FICA(config)#policy-map front-conf
FICA(config-pmap)#class VOZ
FICA(config-pmap-c)#set ip dscp ef
FICA(config-pmap-c)#exit
FICA(config-pmap)#class SignaVOip
FICA(config-pmap-c)#set ip dscp cs3
FICA(config-pmap-c)#exit
FICA(config-pmap)#class SignaSTREAM
FICA(config-pmap-c)#set ip dscp af43
FICA(config-pmap-c)#exit
FICA(config-pmap)#class FTP
FICA(config-pmap-c)#set ip dscp af33
FICA(config-pmap-c)#exit
FICA(config-pmap)#class WEB
FICA(config-pmap-c)#set ip dscp af23
FICA(config-pmap-c)#end
```

En la Figura 57 se muestra la aplicación de la política de tráfico en la frontera de confianza mediante el comando **`service-policy`**, donde hay que tener en cuenta la dirección del tráfico de la red, en este caso al tratarse de la interfaz F2/0 se establece como entrada, dado que por esta pasan los paquetes que se generan en la red LAN, y como se aprecia su secuencia de comandos es bastante sencilla.

Figura 57

Frontera de confianza aplicada para la red LAN FICA

```
FICA(config)#interface f2/0
FICA(config-if)#service-policy input front-conf
FICA(config-if)#end
```

4.5 Teoría de colas

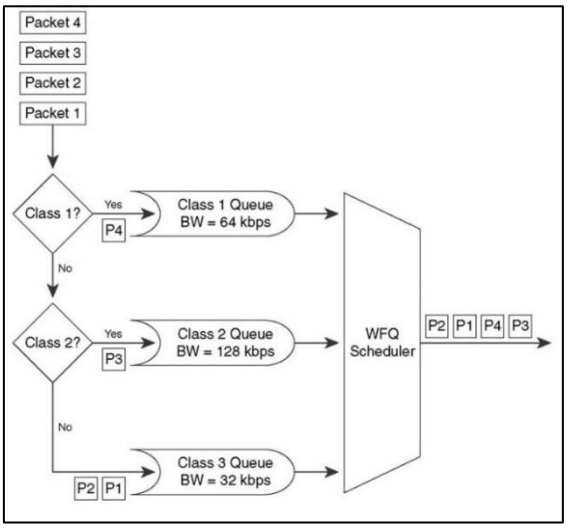
Existen diferentes tipos de encolamiento, pero en el presente trabajo se hace uso de *"Colas justas ponderadas basadas en la clase"* (CBWFQ), la cual, define las clases de tráfico según los criterios de coincidencia, incluidos los protocolos, ACL's y las interfaces de entrada, los paquetes que cumplen con los criterios de coincidencia para una clase, constituyen el tráfico para dicha clase.

La Figura 58 muestra el proceso de funcionamiento para CBWQ mediante un diagrama de flujo, donde en resumen establece que cada paquete entrante se comprueba a qué clase pertenece, luego se reserva una cola para cada clase y el tráfico perteneciente a una de estas, se dirige a la cola de dicha clase, finalmente el programador WFQ (Cola Justa Ponderada), es el encargado de manejar los flujos de cada clase.

Para caracterizar una clase, se asigna ancho de banda, peso y máxima longitud de transmisión del paquete, el ancho de banda asignado a una clase es garantizado durante períodos de congestión. La suma de los anchos de banda no debe superar el 75% del disponible, dado que el restante se usa para información de control (Classification, 2016).

Figura 58

Encolamiento CBWFQ



Fuente: (Classification, 2016)

4.5.1. Cálculo del ancho de banda VoIP

En (Cisco, 2008) se hace mención, que para calcular el respectivo AB necesario para VoIP, se utiliza la ecuación 1:

$$\frac{Tamaño\ total\ del\ paquete}{Tamaño\ del\ Payload} = \frac{Ancho\ de\ banda\ total\ requerido}{Ancho\ de\ banda\ nominal} \quad (1)$$

El cálculo del tamaño total del paquete se realiza en la ecuación 2, se basa en el códec G. 711 que es el estándar que maneja Issabel, fórmula:

$$Tamaño\ total\ del\ paquete = Layer\ 2\ header + (IP + UDP + RTP)\ headers + tamaño\ del\ payload\ (bytes) \quad (2)$$

Se reemplazan los valores correspondientes en la ecuación 3:

$$Tamaño\ total\ del\ paquete = 18 + 40\ +\ 160\ \text{(bytes)} \quad (3)$$

$$Tamaño\ total\ del\ paquete = 218\ \text{bytes}$$

En la ecuación 4 con el valor obtenido en la ecuación 3, se procede a despejar el ancho de banda total requerido de la ecuación 1 y se reemplaza cada uno de sus valores.

$$Ancho\ de\ banda\ total\ requerido = \frac{Tamaño\ total\ del\ paquete * Ancho\ de\ banda\ nominal}{Tamaño\ del\ Payload}(Kbps) \quad (4)$$

$$Ancho\ de\ banda\ total\ requerido = \frac{218\ bytes * 64Kbps}{160\ bytes}$$

$$Ancho\ de\ banda\ total\ requerido = 87{,}2\ Kbps$$

El valor obtenido en la ecuación 4 se trata de una sola petición, para lo cual en el presente trabajo se tomará en cuenta ocho peticiones simultáneas. Se calcula en la ecuación 5 el porcentaje de ancho de banda necesario para cumplir con las políticas de QoS; se toma en cuenta el AB Total como 1.544Mbps dado que se trata de una conexión de línea fija digital arrendada.

$$\%AB = (AB\ servicio * \#peticiones)\ * \frac{75\%}{AB\ Total} \quad (5)$$

$$\%AB = (87{,}2\ Kbps * 8) * \frac{75\%}{1.544Mbps} \qquad \%AB = 37{,}5\ \%$$

4.5.2. Cálculo del ancho de banda WEB

Para calcular el AB para el servicio WEB, se toma como referencia (Paessler, 2016) que menciona el tamaño promedio de una página WEB en la actualidad , para lo cual se hace uso de la ecuación 1:

$$AB = T * t * N \quad (1)$$

T = *tamaño promedio de consulta de una página*

t = *tiempo de carga de la página*

N = *el número de peticiones simultaneas*

En la ecuación 2, se procede con el reemplazo de sus respectivos valores para obtener el ancho de banda.

$$AB = \frac{0{,}5Mbps}{1\ Sitio\ Web} * \frac{1\ Sitio\ Web}{10s} * 75 \quad (2)$$

$$AB = 3{,}75Mbps$$

Una vez obtenido el valor de ancho de banda del servicio web para 50 peticiones simultáneas, se calcula en la ecuación 3 el porcentaje de AB necesario para cumplir con las políticas de QoS; se toma en cuenta el AB Total como 1.544Mbps dado que se trata de una conexión de línea fija digital arrendada.

$$\%AB = AB\ servicio\ * \frac{75\%}{AB\ Total}$$

$$\%AB = 3{,}75Mbps * \frac{75\%}{1.544Mbps}$$

$$\%AB = 18{,}21\%$$

4.5.3. Tabulación de resultados

Una vez que los cálculos se realizan, en la Tabla 17 se plantean los resultados correspondientes a la asignación de ancho de banda a cada servicio, hay que tomar en cuenta que el porcentaje establecido satisface todas y cada una de las necesidades de los usuarios de la red.

Tabla 17

Asignación de ancho de banda a cada servicio

PRIORIDAD	APLICACIÓN	CLASE	DSCP	%AB
CRÍTICA	Transmisión de Voz	VOZ	EF	38
	Señalización	SIGNA	CS3	7
	SigneVideo	Streaming	AF43	7
ALTA	FTP	FTP, SFTP	AF33	3
MEDIA	WEB	HTTP / HTTPS	AF23	19

4.5.4. Aplicación de Teoría de Colas

Para empezar al igual que en la frontera de confianza, la Figura 59 muestra el proceso de configuración que se aplica en el router R2, dado que es el más cercano a los servidores o DMZ. Se empieza con la creación de las respectivas ACL's de tipo extendida, pero la diferencia es que ahora estas ACL's no son configuradas con puertos, sino que se utiliza la codificación DSCP.

Figura 59

Establecimiento de ACL´s en router R2

```
R2#configure terminal
Enter configuration commands, one per line.  End with CN
R2(config)#access-list 110 permit udp any any dscp ef
R2(config)#access-list 111 permit udp any any dscp cs3
R2(config)#access-list 112 permit tcp any any dscp af23
R2(config)#access-list 113 permit tcp any any dscp af33
R2(config)#access-list 114 permit udp any any dscp af43
R2(config)#access-list 115 permit tcp any any dscp af43
R2(config)#end
```

En la Figura 60, se muestra el proceso para crear las respectivas clases correspondientes a cada servicio, a través del comando **`class-map`**, luego se les asocia las ACL's que han sido establecidas anteriormente con el comando **`match access-group`**, lo más importante es identificarlas de acuerdo con el servicio que cumple cada una.

Figura 60

Establecimiento de clases y asignación de ACL's

```
R2#configure terminal
Enter configuration commands, one per lin
R2(config)#class-map Q-VOZ
R2(config-cmap)#match access-group 110
R2(config-cmap)#class-map Q-SIGNA
R2(config-cmap)#match access-group 111
R2(config-cmap)#class-map Q-WEB
R2(config-cmap)#match access-group 112
R2(config-cmap)#class-map Q-FTP
R2(config-cmap)#match access-group 113
R2(config-cmap)#class-map Q-SignaSTREAM
R2(config-cmap)#match access-group 114
R2(config-cmap)#end
```

Con el escenario de teoría de colas listo, en la Figura 61 se muestra la configuración en el que cada clase ahora va a tener un ancho de banda definido con el comando **`bandwidth percent`**, para esto se establece las políticas de QoS que le corresponde a cada una, de acuerdo con lo establecido en la Tabla 16 sobre el ancho de banda designado a cada servicio.

Figura 61

Establecimiento de ancho de banda a cada servicio

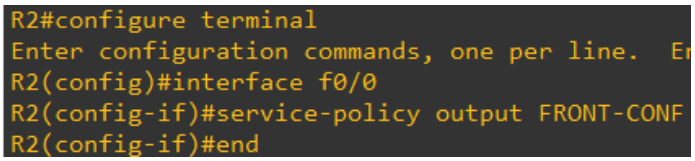
```
R2#configure terminal
Enter configuration commands, one per l
R2(config)#policy-map FRONT-CONF
R2(config-pmap)#class Q-VOZ
R2(config-pmap-c)#bandwidth percent 38
R2(config-pmap-c)#class Q-SIGNA
R2(config-pmap-c)#bandwidth percent 7
R2(config-pmap-c)#class Q-SignaSTREAM
R2(config-pmap-c)#bandwidth percent 7
R2(config-pmap-c)#class Q-FTP
R2(config-pmap-c)#bandwidth percent 3
R2(config-pmap-c)#class Q-WEB
R2(config-pmap-c)#bandwidth percent 19
R2(config-pmap-c)#end
```

Finalmente, la Figura 62 muestra cómo por medio del comando **`service-policy`**, se aplica la política de tráfico para teoría de colas, tomando en cuenta la dirección del tráfico de la red, en este caso lo que se busca es poder entregar QoS y diferenciar los servicios de los paquetes que ingresan hacia la DMZ cuando estos son solicitados por los usuarios.

La interfaz F0/0 del router R2 es la que tiene el enlace hacia la DMZ por lo que se establece como salida, esto es porque lo que se busca es tratar los paquetes que llegan marcados desde las redes LAN, de acuerdo con la prioridad y ancho de banda que les corresponde, evitando de esta manera la congestión de la red, para así tener gestión de servicios diferenciados con políticas de QoS.

Figura 62

Teoría de colas aplicada a los servicios de la DMZ



CAPÍTULO V

PRUEBAS DE FUNCIONALIDAD Y RESULTADOS

Este capítulo final consiste en el planteamiento de las pruebas de funcionamiento de la red híbrida SDN/MPLS propuesta en el presente trabajo de titulación, que ofrece una solución para QoS en servicios diferenciados brindando un servicio confiable y seguro a los usuarios que utilicen los servicios de la DMZ, mediante el marcaje DSCP y encolamiento CBWFQ priorizando una gestión inteligente del tráfico de red.

5.1. Direccionamiento y enrutamiento aplicado en los equipos de la red

Para el desarrollo de la topología de red en base con el diseño propuesto, es necesario verificar el direccionamiento designado a cada una de las interfaces involucradas de los equipos que componen la misma. En la Figura 63 se muestra el direccionamiento aplicado a cada uno de los routers C7200 de acuerdo con lo contemplado en la Tabla 10, para lo cual se hace uso del comando **`show ip interface brief`**.

Figura 63

Direccionamiento aplicado en las interfaces de los Routers

```
R1#show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            192.168.137.2   YES NVRAM  up                    up
FastEthernet0/1            192.168.173.1   YES NVRAM  up                    up
Serial1/0                  192.168.173.81  YES NVRAM  up                    up
```

```
R2#show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            192.168.173.17  YES NVRAM  up                    up
Serial1/0                  192.168.173.82  YES NVRAM  up                    up
```

```
FICA#show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            192.168.173.8   YES NVRAM  up                    up
Serial1/0                  192.168.173.85  YES NVRAM  up                    up
```

```
FICAYA#show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            unassigned      YES NVRAM  administratively down down
Serial1/0                  192.168.173.86  YES NVRAM  up                    up
Serial1/1                  192.168.173.89  YES NVRAM  up                    up
Serial1/2                  unassigned      YES NVRAM  administratively down down
Serial1/3                  unassigned      YES NVRAM  administratively down down
FastEthernet2/0            192.168.173.49  YES NVRAM  up                    up
```

```
FACAE#show ip interface brief
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            unassigned      YES NVRAM  administratively down down
Serial1/0                  unassigned      YES NVRAM  administratively down down
Serial1/1                  192.168.173.90  YES NVRAM  up                    up
Serial1/2                  unassigned      YES NVRAM  administratively down down
Serial1/3                  unassigned      YES NVRAM  administratively down down
FastEthernet2/0            192.168.173.65  YES NVRAM  up                    up
```

Para el verificar que el enrutamiento se ha establecido correctamente, mediante el comando **`show ip ospf neighbor`** se consulta la tabla de vecinos de cada router de la topología, estas tablas se detallan y analizan a continuación.

a) Router 1

Al analizar la tabla de vecinos en R1 en la Figura 64, se muestra que los router ID pertenecen a R2 (2.2.2.2) y FICA (3.3.3.3), a través de las interfaces de red S1/0 y F0/1 respectivamente. En R2 se tiene estado **FULL/-** debido a que se trata de un enlace serial, y FICA se encuentra en estado **FULL/DR** al tener el router ID más alto**,** indicando plena adyacencia con el vecino DR.

Figura 64

Vecinos OSPF de R1

```
R1#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
2.2.2.2           0   FULL/  -        00:00:30    192.168.173.82  Serial1/0
3.3.3.3           1   FULL/DR         00:00:31    192.168.173.8   FastEthernet0/1
```

b) Router 2

La Figura 65 muestra que en R2 existe un solo vecino, se trata de R1 (1.1.1.1), que está en estado **FULL/-** al tratarse de un enlace serial, en este caso no existe un DR o BDR definido, pero de igual manera cumple con la función de enviar paquetes de saludo, actualizaciones respuestas, etc.

Figura 65

Vecinos OSPF de R2

```
R2#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
1.1.1.1           0   FULL/  -        00:00:31    192.168.173.81  Serial1/0
```

c) Router FICA

La Figura 66 muestra que en el router FICA se tiene dos vecinos, los cuales pertenecen a R1 (1.1.1.1) y a FICAYA (4.4.4.4) respectivamente. El router FICAYA que es de mayor ID, se encuentra en estado **FULL/-** debido a que se trata de un enlace serial, en cambio R1 se encuentra en estado **FULL/BDR** debido a que tiene un ID menor.

Figura 66

Vecinos OSPF de router FICA

```
FICA#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
4.4.4.4           0   FULL/  -        00:00:38    192.168.173.86  Serial1/0
1.1.1.1           1   FULL/BDR        00:00:34    192.168.173.1   FastEthernet0/0
```

d) Router FICAYA

Para el router FICAYA la Figura 67 muestra que se tiene dos vecinos, estos son FICA (3.3.3.3) y FACAE (5.5.5.5), los cuales se encuentran en estado **FULL/-** dado que se tratan de enlaces seriales, en esta tabla como se puede apreciar a pesar de tener dos vecinos ninguno se define como DR o BDR, sin embargo, cumplen la misma función de permitir la interconexión de la red.

Figura 67

Vecinos OSPF de router FICAYA

```
FICAYA#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
3.3.3.3           0   FULL/  -        00:00:33    192.168.173.85  Serial1/0
5.5.5.5           0   FULL/  -        00:00:37    192.168.173.90  Serial1/1
```

e) Router FACAE

Finalmente, la Figura 68 muestra la tabla de vecinos del router FACAE, en donde se encuentra un solo vecino que es el router FICAYA (4.4.4.4), este se encuentra en estado **FULL/-** al tratarse de un enlace serial, más allá de eso, con esto se verifica que existe interconexión entre redes y que todos en la red se pueden comunicar entre sí.

Figura 68

Vecinos OSPF de router FACAE

```
FACAE#show ip ospf neighbor

Neighbor ID     Pri   State           Dead Time   Address         Interface
4.4.4.4           0   FULL/  -        00:00:33    192.168.173.89  Serial1/1
```

5.2. Tablas de enrutamiento

Una tabla de este tipo contiene toda la información necesaria para hacer que uno o varios paquetes de datos puedan viajar a través de la red utilizando el mejor camino. Establecido el enrutamiento y los vecinos OSPF, ahora cada equipo es capaz de aprender las diferentes

redes de la topología, permitiendo así la interconexión entre estaciones para llevar sin interrupciones un paquete desde su IP origen hacia la IP destino por una ruta establecida.

Para visualizar la tabla de enrutamiento, se hace uso del comando **`show ip route`** que permite saber el estado actual de dicha tabla, en la Figura 69 se muestran las tablas de los routers de la topología de red (R1, R2, FICA, FICAYA).

Figura 69

Tablas de enrutamiento

```
R1#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is 192.168.137.1 to network 0.0.0.0

     192.168.173.0/24 is variably subnetted, 6 subnets, 2 masks
O       192.168.173.84/30
           [110/65] via 192.168.173.8, 07:41:57, FastEthernet0/1
C       192.168.173.80/30 is directly connected, Serial1/0
C       192.168.173.0/28 is directly connected, FastEthernet0/1
O       192.168.173.16/28 [110/65] via 192.168.173.82, 07:40:28, Serial1/0
O       192.168.173.32/28 [110/2] via 192.168.173.8, 07:41:57, FastEthernet0/1
O       192.168.173.48/28
           [110/66] via 192.168.173.8, 07:41:57, FastEthernet0/1
C    192.168.137.0/24 is directly connected, FastEthernet0/0
S*   0.0.0.0/0 [1/0] via 192.168.137.1
```

```
R2#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     192.168.173.0/24 is variably subnetted, 6 subnets, 2 masks
O       192.168.173.84/30 [110/129] via 192.168.173.81, 11:21:24, Serial1/0
C       192.168.173.80/30 is directly connected, Serial1/0
O       192.168.173.0/28 [110/65] via 192.168.173.81, 11:21:24, Serial1/0
C       192.168.173.16/28 is directly connected, FastEthernet0/0
O       192.168.173.32/28 [110/66] via 192.168.173.81, 11:21:24, Serial1/0
O       192.168.173.48/28 [110/130] via 192.168.173.81, 11:21:24, Serial1/0
O    192.168.137.0/24 [110/65] via 192.168.173.81, 11:21:24, Serial1/0
```

```
FICA#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     192.168.173.0/24 is variably subnetted, 6 subnets, 2 masks
C       192.168.173.84/30 is directly connected, Serial1/0
O       192.168.173.80/30
           [110/65] via 192.168.173.1, 11:22:03, FastEthernet0/0
C       192.168.173.0/28 is directly connected, FastEthernet0/0
O       192.168.173.16/28
           [110/66] via 192.168.173.1, 11:21:53, FastEthernet0/0
C       192.168.173.32/28 is directly connected, FastEthernet2/0
O       192.168.173.48/28 [110/65] via 192.168.173.86, 11:23:41, Serial1/0
O    192.168.137.0/24 [110/2] via 192.168.173.1, 11:23:31, FastEthernet0/0
```

```
FICAYA#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     192.168.173.0/24 is variably subnetted, 6 subnets, 2 masks
C       192.168.173.84/30 is directly connected, Serial1/0
O       192.168.173.80/30 [110/129] via 192.168.173.85, 11:23:05, Serial1/0
O       192.168.173.0/28 [110/65] via 192.168.173.85, 11:24:30, Serial1/0
O       192.168.173.16/28 [110/130] via 192.168.173.85, 11:22:55, Serial1/0
O       192.168.173.32/28 [110/65] via 192.168.173.85, 11:24:30, Serial1/0
C       192.168.173.48/28 is directly connected, FastEthernet2/0
O    192.168.137.0/24 [110/66] via 192.168.173.85, 11:24:30, Serial1/0
```

En la Tabla 18 tomando en cuenta la tabla de enrutamiento de R1, se explica cada uno de los parámetros que nos muestra una tabla de enrutamiento, entre los cuales están: Protocolo de enrutamiento, redes remotas, distancia administrativa, métrica de la ruta y el tiempo de actualización de la ruta.

Tabla 18

Descripción de la tabla de enrutamiento de R1

Campo	Descripción
O	Indica el protocolo de la ruta, en este caso OSPF
192.168.173.84	Indica la dirección de la red remota.
[110/65]	El primer número entre corchetes es la distancia administrativa de la fuente de información; el segundo número es la métrica de la ruta.
via 192.168.173.8	Especifica la dirección del siguiente enrutador a la red remota.
07:41:57	Especifica la última vez que se actualizó la ruta en horas:minutos:segundos

Una vez se han detallado los parámetros que componen una tabla de enrutamiento, en la Figura 70 se muestra la tabla de enrutamiento del router FACAE, en donde al analizar las redes que se muestran se aprecia cuáles han sido aprendidas a través de OSPF.

La letra **C** define a las redes conectadas directamente, estas son la **F2/0 192.168.173.64/28** y la **S1/1 192.168.173.88/30;** con la letra **O** en cambio se definen las redes aprendidas por OSPF y al revisar cada una de ellas, se confirma que todas las redes de la topología han sido descubiertas por el router a través de la interfaz **S1/1** conectada directamente hacia el router FICAYA.

Figura 70

Tablas de enrutamiento router FACAE

```
FACAE#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     192.168.173.0/24 is variably subnetted, 8 subnets, 2 masks
C       192.168.173.64/28 is directly connected, FastEthernet2/0
O       192.168.173.84/30 [110/128] via 192.168.173.89, 00:00:07, Serial1/1
O       192.168.173.80/30 [110/193] via 192.168.173.89, 00:00:07, Serial1/1
C       192.168.173.88/30 is directly connected, Serial1/1
O       192.168.173.0/28 [110/129] via 192.168.173.89, 00:00:07, Serial1/1
O       192.168.173.16/28 [110/194] via 192.168.173.89, 00:00:07, Serial1/1
O       192.168.173.32/28 [110/129] via 192.168.173.89, 00:00:07, Serial1/1
O       192.168.173.48/28 [110/65] via 192.168.173.89, 00:00:07, Serial1/1
O    192.168.137.0/24 [110/130] via 192.168.173.89, 00:00:07, Serial1/1
```

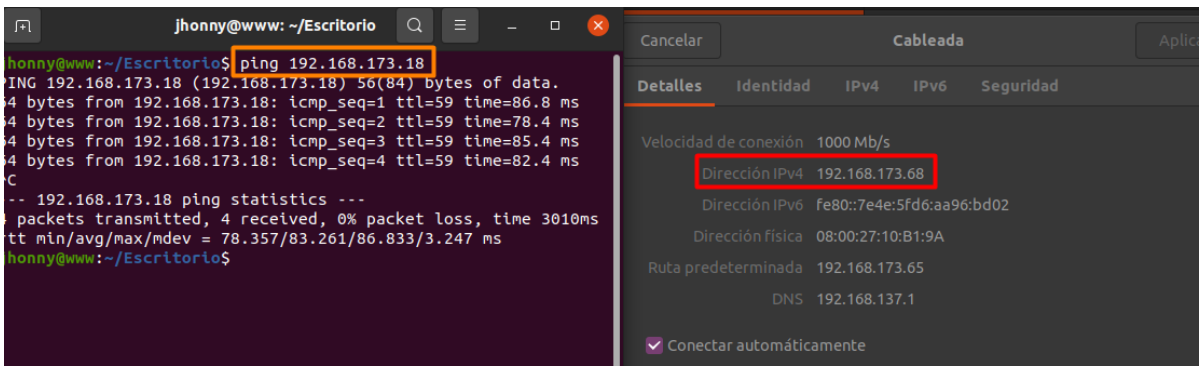
5.3. Pruebas de conectividad

a) Caso 1

Para comprobar el funcionamiento de la topología, mediante un PING se realiza el diagnóstico del estado de la comunicación de la red entre los diferentes equipos que la componen, concretamente se envían paquetes ICMP de solicitud (Echo Request) y de respuesta (Echo Reply). En la Figura 71 con la consola de Ubuntu, se muestra un ping desde el host de la LAN FACAE (Cliente 3) marcando en el recuadro rojo su dirección IP, hacia el servidor de VoIP marcando su dirección IP en el recuadro naranja, el cual es exitoso, pero analizando más a detalle se puede observar que el mensaje ICMP que se envía se encuentra incrustado dentro de un paquete IP, incluye un número identificador y una secuencia de números que debe coincidir con el mensaje ICMP de respuesta.

Figura 71

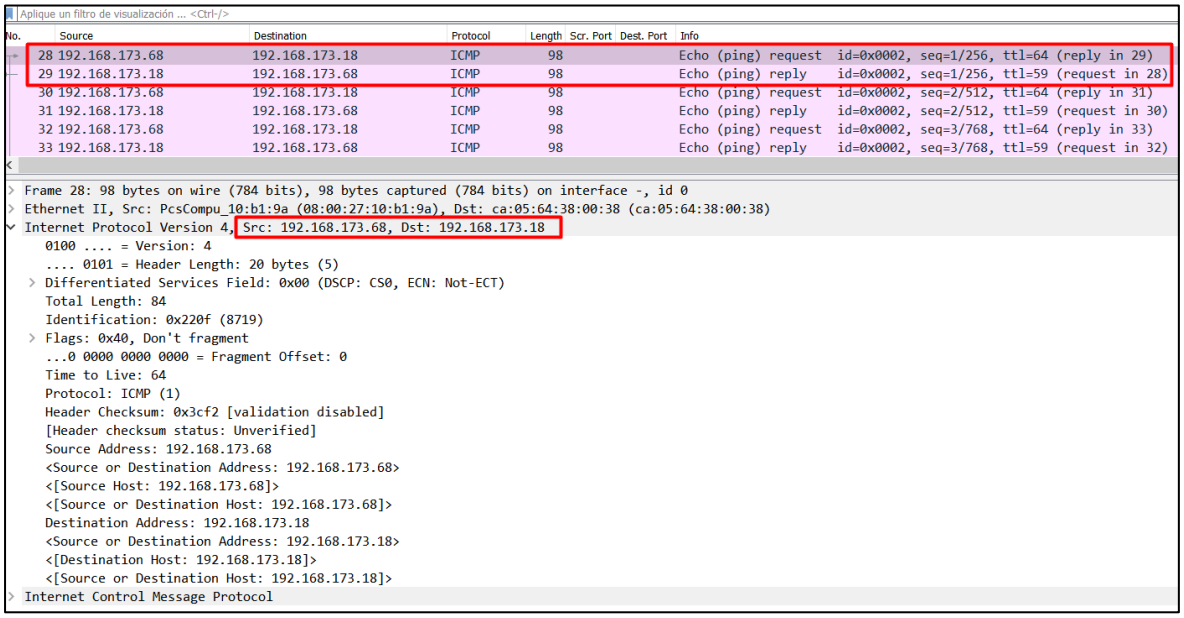
Ping Cliente 3 - Servidor VoIP



Para entender lo explicado anteriormente, con el sniffer Wireshark se capturan los paquetes ICMP que se generan y se envían en la red, en la Figura 72 se observa la IP de origen y la IP de destino, donde una vez la petición de ping se ejecuta el Cliente 3 envía el Echo request, al ser aceptada el servidor de VoIP que en este caso está involucrado envía el Echo reply, de esta manera se realiza el intercambio hasta que se decida finalizar la solicitud de ping.

Figura 72

Captura de paquetes ICMP en el sniffer Wireshark caso 1

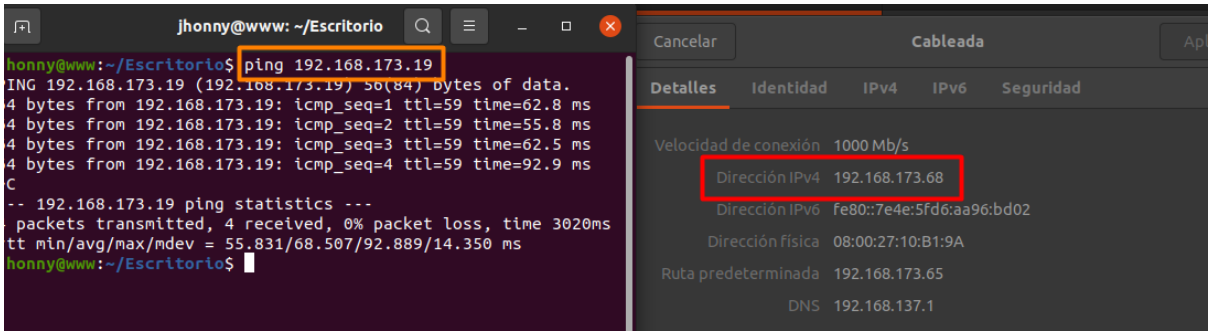


b) Caso 2

De la misma manera se procede a realizar una prueba de ping desde el Cliente 3 hacia el servidor de Hosting representado en la Figura 73, en el recuadro rojo se encuentra la dirección IP configurada en el host y en el recuadro naranja está la dirección IP del servidor, mediante la consola de Ubuntu se realiza la prueba de conectividad que es exitosa, demostrando de esta manera que tenemos comunicación con el servidor de Hosting.

Figura 73

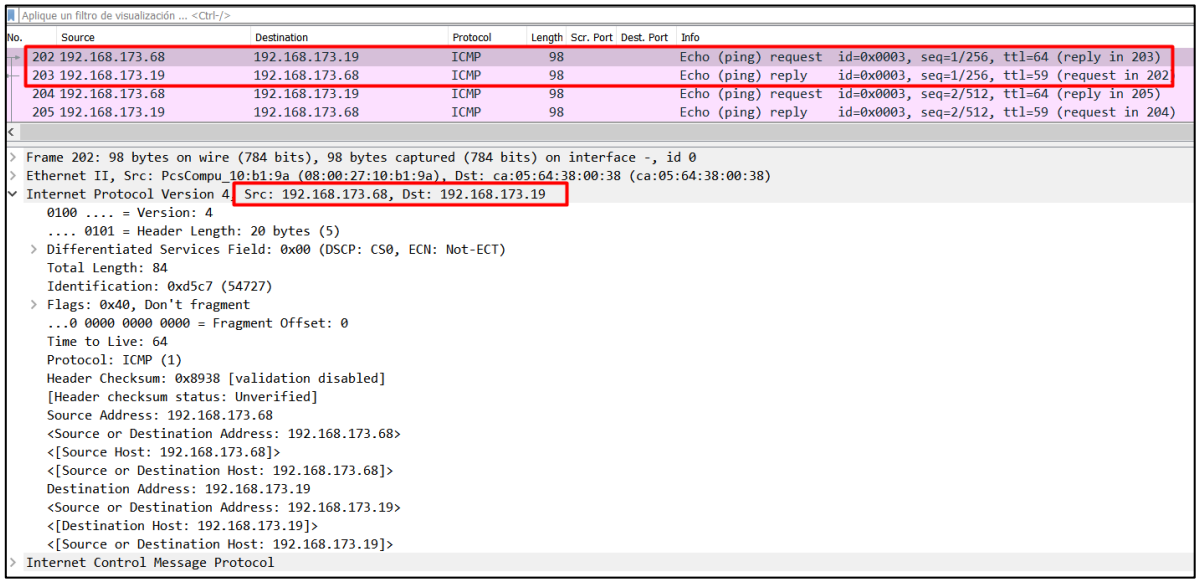
Ping Cliente 3 - Servidor Hosting



Con la captura de paquetes en marcha, en la Figura 74 se observan los paquetes ICMP que se generan al realizar un ping entre en el Cliente 3 y el servidor Hosting, el cual se realiza con éxito y cada Echo request obtiene su respectivo Echo reply en el orden que va llegando cada paquete, la ventaja de Wireshark es que nos da un informe detallado de la capa física, enlace de datos y red en base al modelo OSI.

Figura 74

Captura de paquetes ICMP en el sniffer Wireshark caso 2

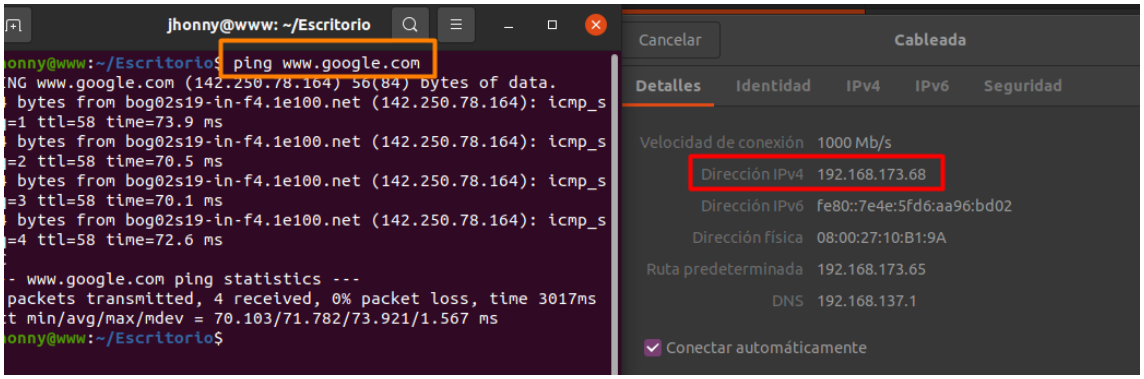


c) Caso 3

Por último para comprobar que también se tiene conexión hacia el Internet, se realiza un ping desde el Cliente 3 hacia la URL de Google (www.google.com), y como se puede apreciar en la Figura 75 el ping se realiza correctamente, dando como resultado que la LAN FACAE tiene salida hacia los servicios del Internet para navegar libremente, eso hay que configurar una dirección de DNS para resolver los dominios, en este caso la dirección IP es la 192.168.137.1 o también la de Google 8.8.8.8, el funcionamiento no cambia si se usa una u otra.

Figura 75

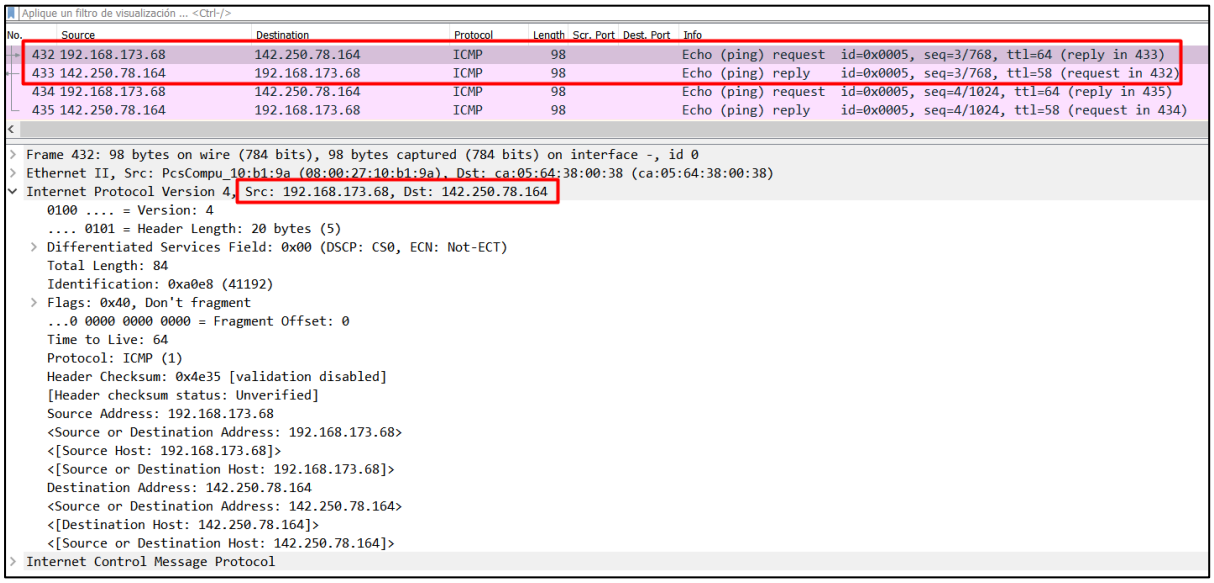
Ping Cliente 3 - Internet (Google)



Finalmente se capturan los paquetes ICMP y como se muestra en la Figura 76, aunque se haya digitado una URL para realizar el ping, dicha URL siempre va a estar ligada a una dirección IP, en este caso es una dirección IP pública 142.250.78.164 que responde a la petición del Cliente 3.

Figura 76

Captura de paquetes ICMP en el sniffer Wireshark caso 3



5.4. Pruebas de funcionamiento de servicios

5.4.1. Servidor Hosting

Una vez que comprobada la conectividad en la red, se procede con la validación de funcionamiento de cada servidor, para lo cual desde el Cliente 3 abriendo el navegador (Mozilla) en la barra de navegación resaltada en el recuadro rojo, se digita la dirección IP 192.168.173.19 del servidor Web. En la Figura 77 se realiza un acceso mediante HTTP, es

decir, una página sin ningún tipo de seguridad que carga correctamente mostrando el contenido de esta.

Figura 77

Acceso al servicio Web desde el Cliente 3

Se comprueba de igual manera el acceso al servicio Web seguro o HTTPS, y como se aprecia en la Figura 78 ahora en la barra de navegación se debe digitar https://IP-Servidor para poder acceder al contenido, una vez carga la página se accede al certificado de esta para corroborar que cumple con los requisitos de seguridad HTTPS.

Figura 78

Acceso al servicio Web Seguro desde el Cliente 3

Continuando con la validación del servicio, en el Cliente 2 de igual manera se procede a accla página Web desde el navegador, esto se comprueba en la Figura 52 dado que el contenido carga por completo, eso sí aclarando que se accede mediante HTTP a través del puerto 79.

Figura 79

Acceso al servicio Web desde el Cliente 2

En la Figura 80 se accede a través de HTTPS con el puerto 443, donde nuevamente el contenido carga con éxito refutando que la conectividad en la red se encuentra establecida, sin olvidarse que para HTTPS se debe hacer uso de un certificado, para el presente trabajo se establece este con encriptación asimétrica RSA (Rivest-Shamir-Adleman) de 64 bits, garantizando así que todo el tráfico que viaja en la red no se encuentra en texto plano.

Figura 80

Acceso al servicio Web Seguro desde el Cliente 2

Finalmente, en la Figura 81 se accede desde el Cliente 1 hacia el servidor Web para comprobar que todas las redes LAN (FICA, FICAYA, FACAE) tienen acceso al servicio, al igual que en los otros hosts se accede mediante HTTP puerto 80, y como era de esperarse el contenido carga con éxito.

Figura 81

Acceso al servicio Web desde el Cliente 1

En la Figura 82, se realiza la última prueba de funcionamiento al servidor web mediante HTTPS con el respectivo certificado, al cual accede cargando el contenido correctamente y así dar el visto el bueno que el servicio Web seguro se ejecuta como lo esperado en todos los hosts de la red, a través del puerto 443.

Figura 82

Acceso al servicio Web Seguro desde el Cliente 1

5.4.2. Captura y análisis de paquetes Servidor Hosting

Al momento de iniciar una conexión TCP, se lo hace mediante un intercambio de mensajes de sincronismo (SYN) a tres vías, para establecer o inicializar una conexión, los dos hosts deben sincronizar sus números de secuencia iniciales o ISN (Initial Sequence Numbers). Esta solución necesita un mecanismo adecuado para elegir un número de secuencia inicial, la sincronización requiere que tanto el cliente como el servidor envíen su propio número de secuencia inicial y así recibir una confirmación del intercambio en un acuse de recibo (ACK) de la otra parte. La secuencia de mensajes TCP se detalla en la en la Figura 83 y se describe a continuación.

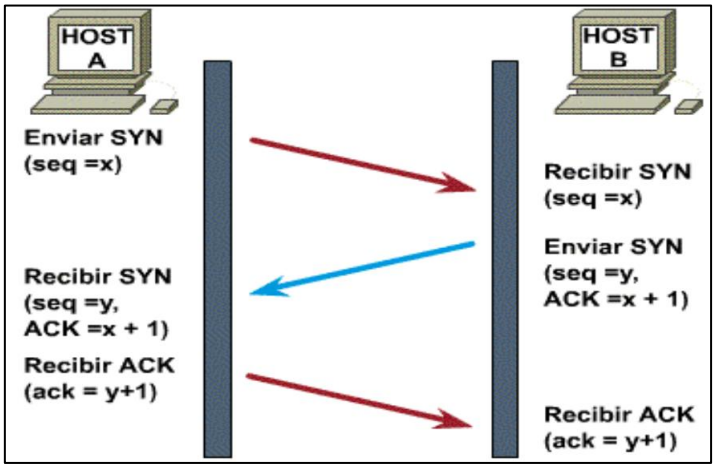
- El origen (A) inicializa una conexión mandando un paquete de SYN hacia el host destino (B) indicando su INS = X:

  A—>B SYN, seq de A = X

- B recibe el paquete, graba que el seq de A = X, responde con un ACK de X + 1, e indica que su INS = Y. El ACK de X + 1 significa que el host B recibió todos los octetos incluyendo X y ahora espera X + 1 siguiente:
  B—>A ACK, seq de A = X, SYN seq de B = Y, ACK = X + 1
- A recibe el paquete de B, y sabe que el seq de B = Y, y responde con un ACK de Y + 1, el cual termina el proceso de conexión:
  A—>B ACK, seq de B = Y, ACK = Y + 1

Figura 83

Intercambio de señales de tres vías



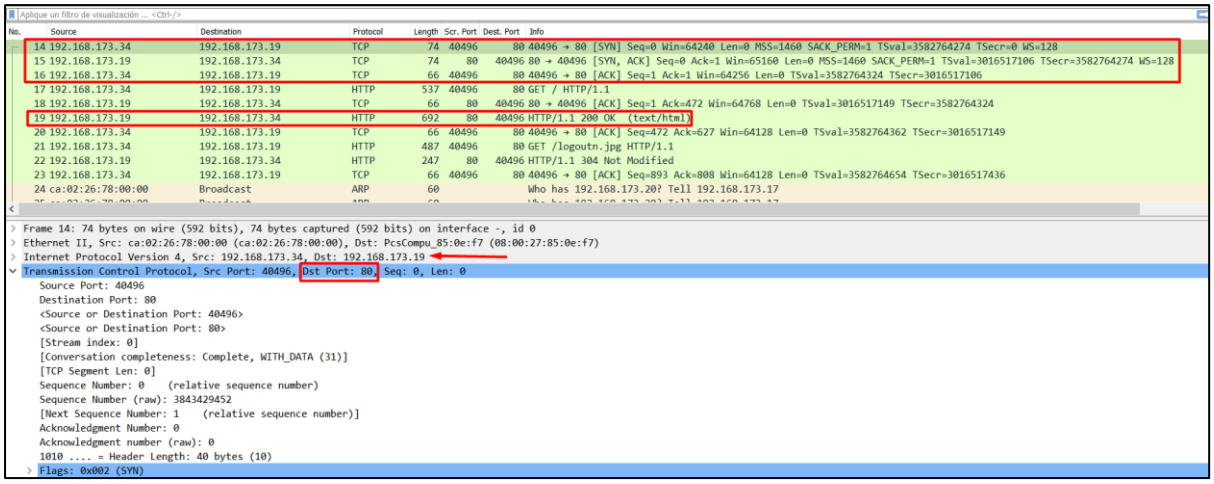
Fuente: (Martín & Cely, 2015)

Validado en la sección 5.4.1 a disponibilidad del servicio en todas las redes LAN, se procede con la captura de paquetes en el sniffer Wireshark, en este caso se realizó con el Cliente1 (LAN FICA) de dirección IP 192.168.173.34. Detallado el proceso de intercambio de señales de tres vías, desde el navegador se utiliza el protocolo HTTP con puerto 80 para ingresar hacia la página web, una vez la solicitud se realiza y la página web carga su contenido, en el primer recuadro rojo de la Figura 84 se puede apreciar dicho intercambio.

Primero el cliente envía el paquete SYN hacia el servidor solicitando el servicio, segundo el servidor envía el paquete SYN + ACK hacia el cliente indicando que el servicio se encuentra disponible y tercero el cliente envía el ACK al servidor para terminar de establecer la comunicación. Finalmente, el servidor envía el código de respuesta 200 OK señalado en el segundo recuadro rojo de la Figura 84, para indicar que la solicitud ha tenido éxito.

Figura 84

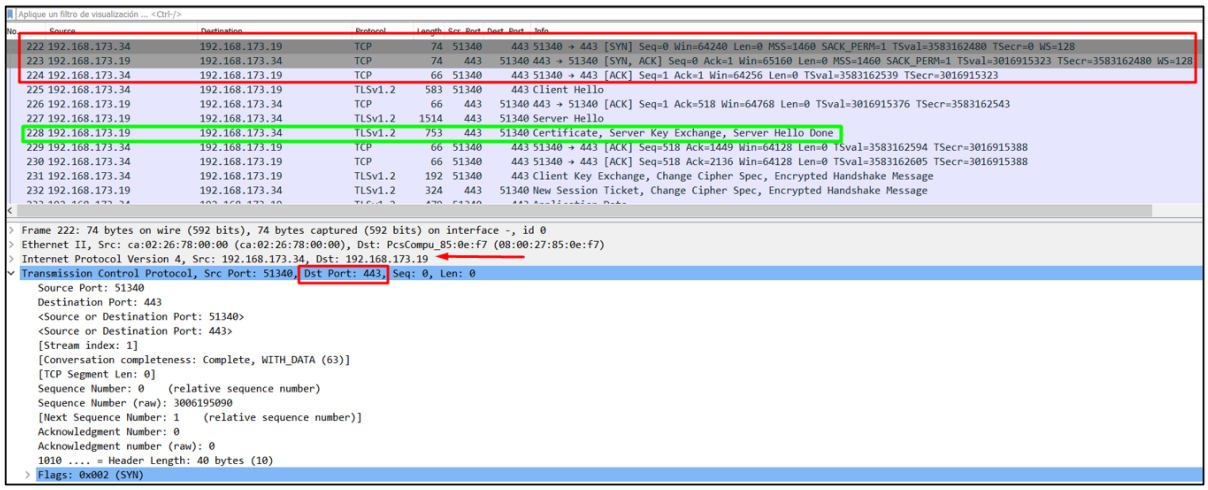
Intercambio de señales de tres vías HTTP



De igual manera con HTTPS se realizó el mismo proceso desde el Cliente 1, se realiza el mismo intercambio de tres vías resaltado en el primer recuadro rojo de la Figura 85, pero a diferencia de HTTP se tiene la presencia de paquetes TLS (Seguridad de la Capa de Transporte), donde este protocolo se encarga de manejar certificados de seguridad como se puede apreciar en el recuadro de color verde, es así como el servidor envía el intercambio de la llave hacia el Cliente 1 y viceversa para establecer una nueva sesión para visualizar el contenido de la página Web.

Figura 85

Intercambio de señales de tres vías HTTPS



5.4.3. Servidor VoIP

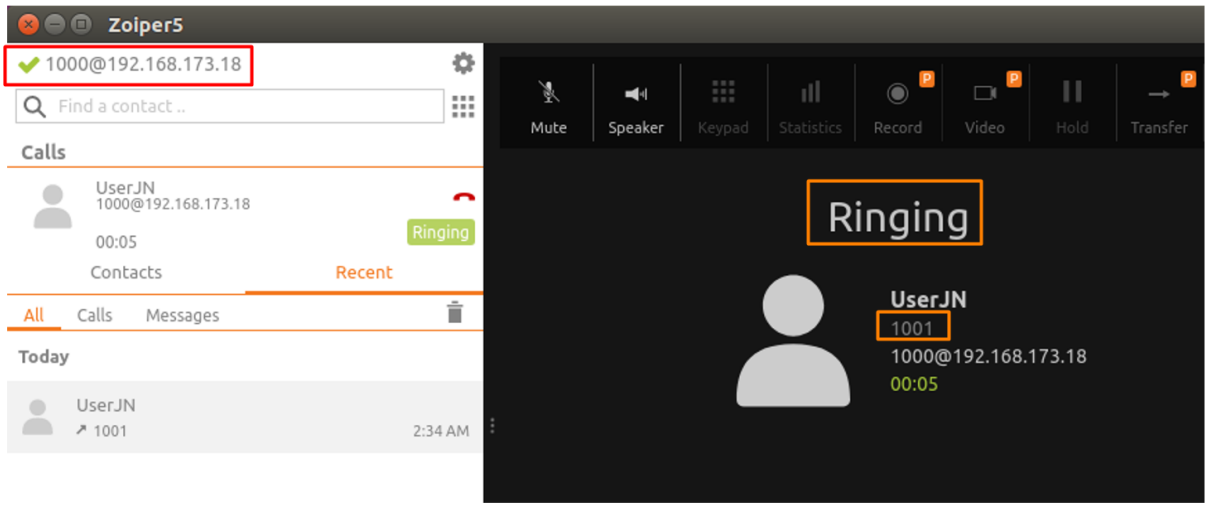
El establecimiento de la comunicación se hace mediante el Protocolo de Inicio de Sesión (SIP) explicado a detalle en la sección 4.2.4, donde básicamente se usa para establecer, modificar y terminar eventualmente una sesión entre 2 o más participantes.

Una vez creadas las respectivas extensiones telefónicas en el servidor de VoIP (Issabel), por medio del softphone Zoiper para comprobar el funcionamiento se registró la extensión 1000 en el Cliente1 (FICA) y la extensión 1001 en el Cliente3 (FACAE), una vez ingresada la clave correspondiente a cada extensión, con un tick verde se verifica que ya está registrada en el servidor y lista para la comunicación entre ellas.

En primera instancia, en la Figura 86 se realiza una llamada del Cliente1 (1000) al Cliente3 (1001), en el recuadro rojo se remarca que la extensión se encuentra registrada correctamente y en los recuadros naranja se hace referencia que el Cliente1 está timbrando al Cliente3.

Figura 86

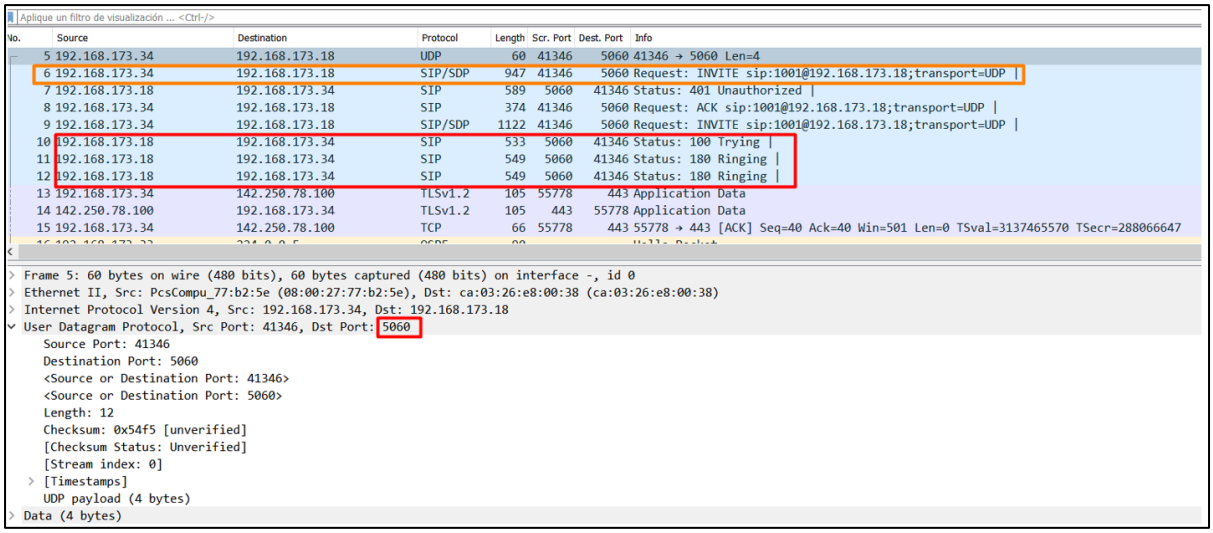
Cliente1 timbrando al Cliente3



Una vez se realiza la llamada al Cliente3, por medio del sniffer los paquetes son capturados, en base a la Figura 44 donde es explicado el intercambio de los paquetes SIP puerto UDP 5060, en el recuadro naranja de la Figura 87 está el paquete INVITE que va de la IP .34 (Cliente1) a la IP.18 (Servidor VoIP), en el recuadro rojo están los paquetes Trying que va de la IP .18 a la IP .34 y Ringing igualmente de la IP .18 a la IP .34, para dar la orden al servidor que se quiere comunicar con la otra extensión.

Figura 87

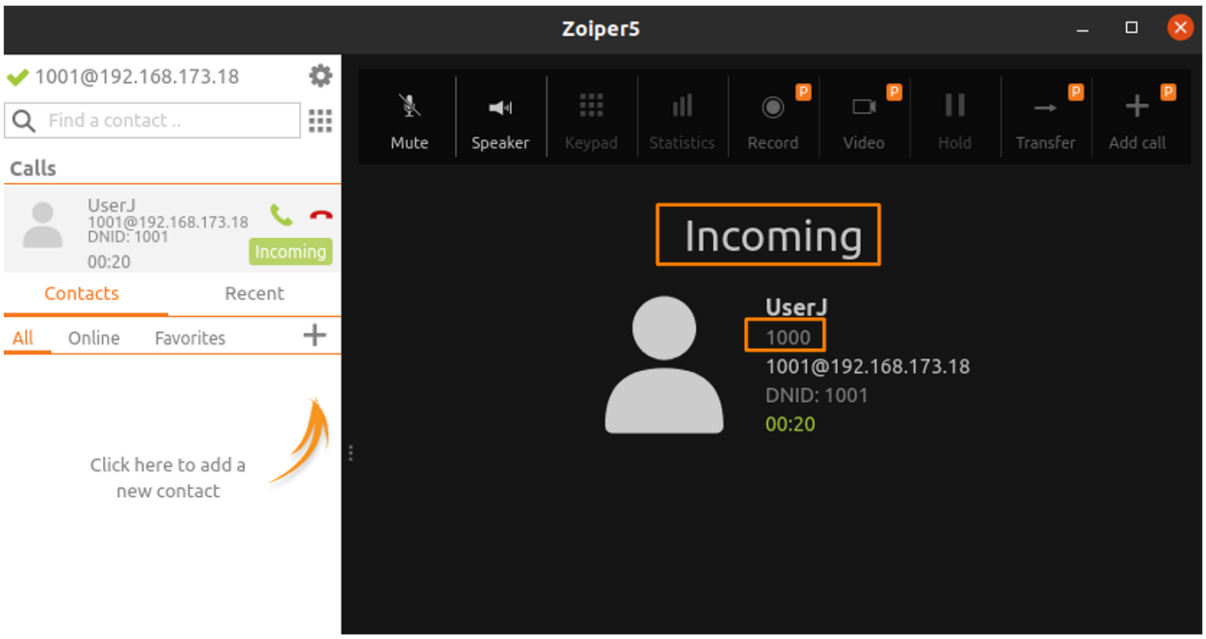
Intercambio de paquetes Invite, Trying y Ringing



En el lado del Cliente3, en la Figura 88 se observa que el Cliente1 está solicitando comunicarse con esta extensión, en el recuadro naranja se resalta que está entrando una llamada por parte del Cliente1 a la espera de que ésta sea contestada.

Figura 88

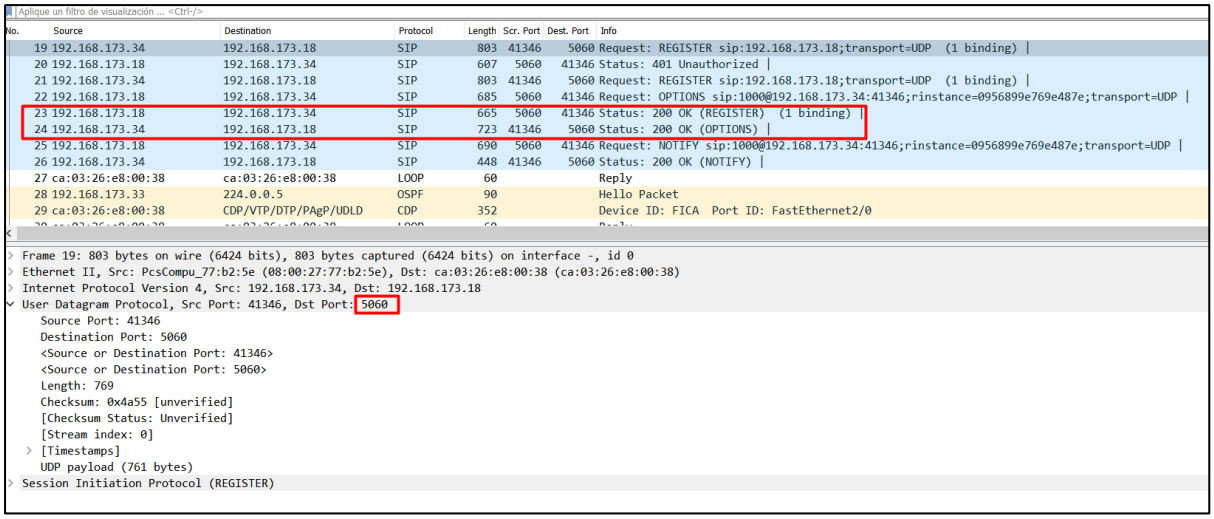
Llamada entrante en el Cliente3 de parte del Cliente1



Una vez la llamada es contestada de parte del Cliente3, al momento de la captura de paquetes en el recuadro rojo de la Figura 89 se resalta los paquetes 200 OK desde la IP .18 a la IP .34 y viceversa, para indicar que la petición se ha realizado correctamente y la llamada fue contestada de parte del Cliente3.

Figura 89

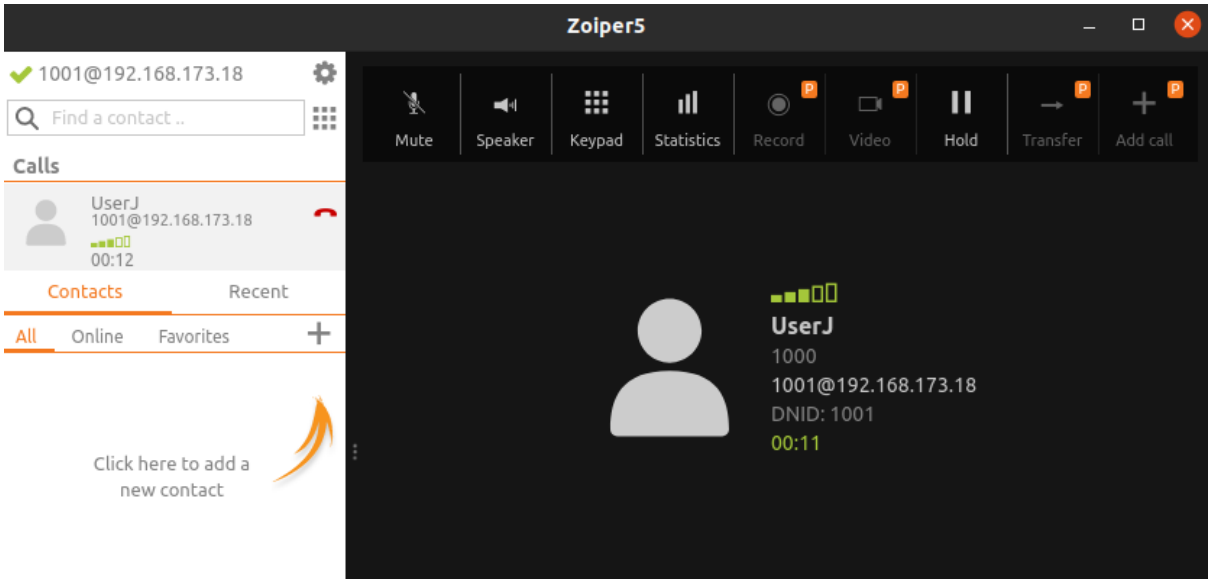
Intercambio de paquetes 200 OK



En la Figura 90 se aprecia que la llamada se encuentra en cursos entre las extensiones, es decir, el protocolo RTP es el que comienza a trabajar hasta el momento de finalizar dicha llamada y liberar nuevamente el canal.

Figura 90

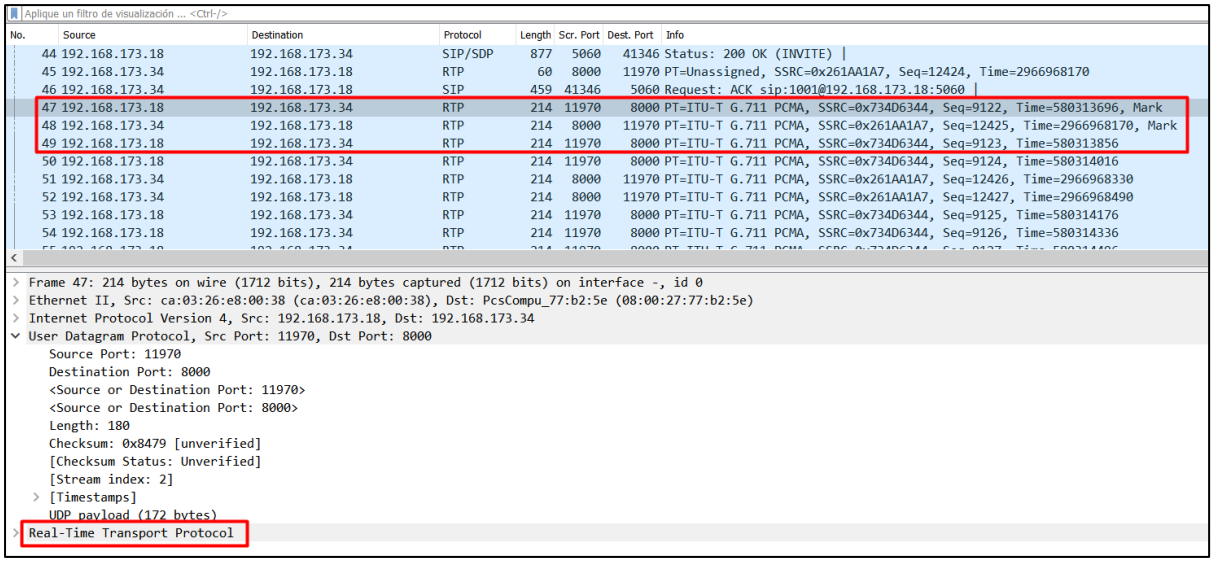
Llamada en curso entre el Cliente1 y el Cliente3



Al momento de realizar la captura de paquetes en el sniffer, en el recuadro rojo de la Figura 91 se puede corroborar que mientras la llamada se encuentra en curso, el protocolo RTP es el que trabaja con el códec G. 711 que se utiliza para la voz digital sin comprimir.

Figura 91

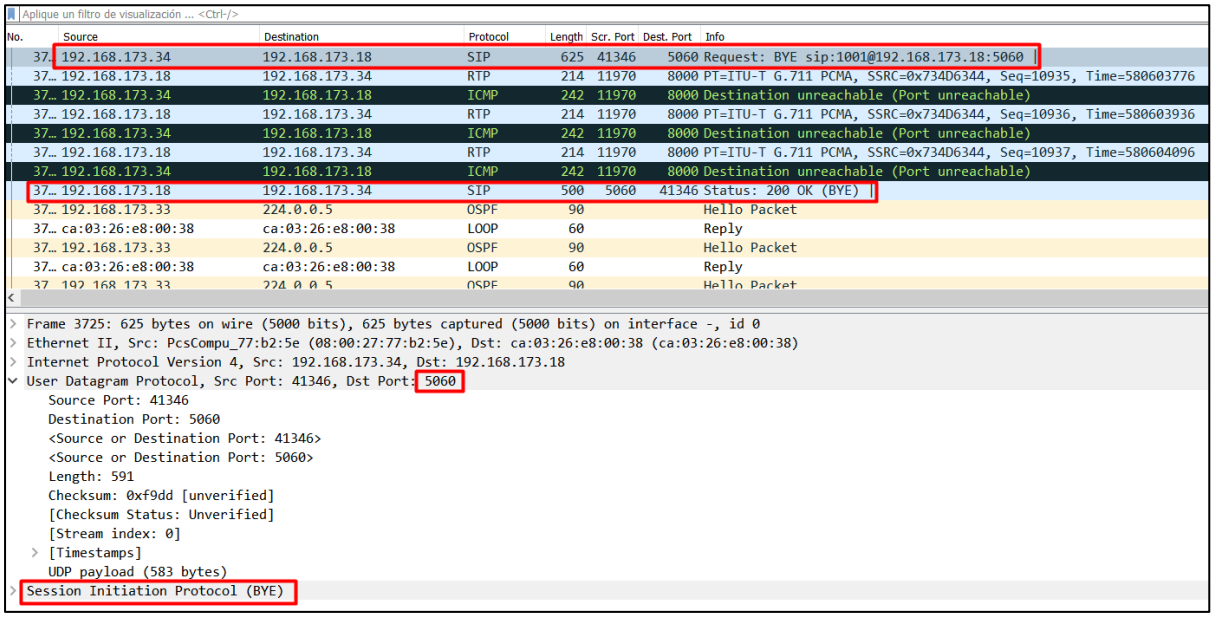
Captura de paquetes RTP



Finalmente, cuando la llamada finaliza al momento de capturar los paquetes con el sniffer, se aprecia en la Figura 92 que nuevamente el protocolo SIP es el que comienza a trabajar y es el que envía el paquete BYE remarcado en el primer recuadro rojo, para indicar la finalización de la llamada entre las extensiones implicadas, por último se envía el paquete 200 OK (BYE) remarcado en el segundo cuadro rojo, dando a conocer al servidor que la comunicación a finalizado y el canal se encuentra libre nuevamente.

Figura 92

Captura de paquetes BYE

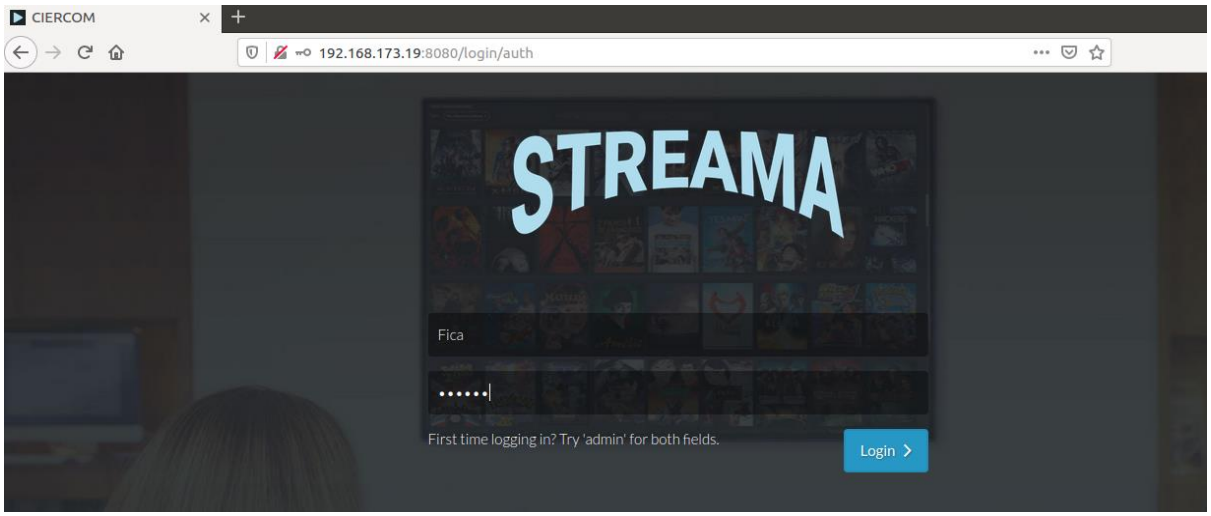


5.4.4. Servidor Streaming

Por último, se comprueba el funcionamiento del servidor de streaming, para lo cual desde el Cliente1 se ingresa en la barra de navegación la dirección IP 192.168.173.19 del servidor, en la Figura 93 se muestra que la plataforma STREAMA carga y se encuentra lista a la espera de ingresar el correspondiente usuario y su contraseña.

Figura 93

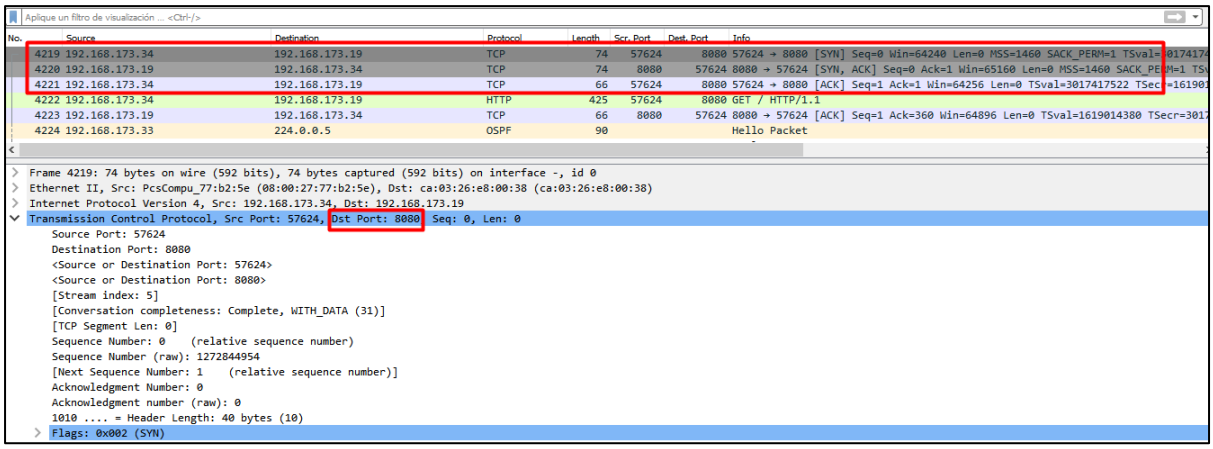
Acceso al servicio de Streaming desde el Cliente 1



Desde la perspectiva del sniffer al momento de realizar la captura de paquetes, en la Figura 94 se verifica que cumple con el intercambio de señales de tres vías, a través del puerto 8080. Primero el Cliente1 con IP .34 envía el paquete SYN al servidor IP .19, luego el servidor responde con el paquete SYN ACK al Cliente1 y finalmente el Cliente1 envía el ACK al servidor para indicar que el acceso se realizó exitosamente.

Figura 94

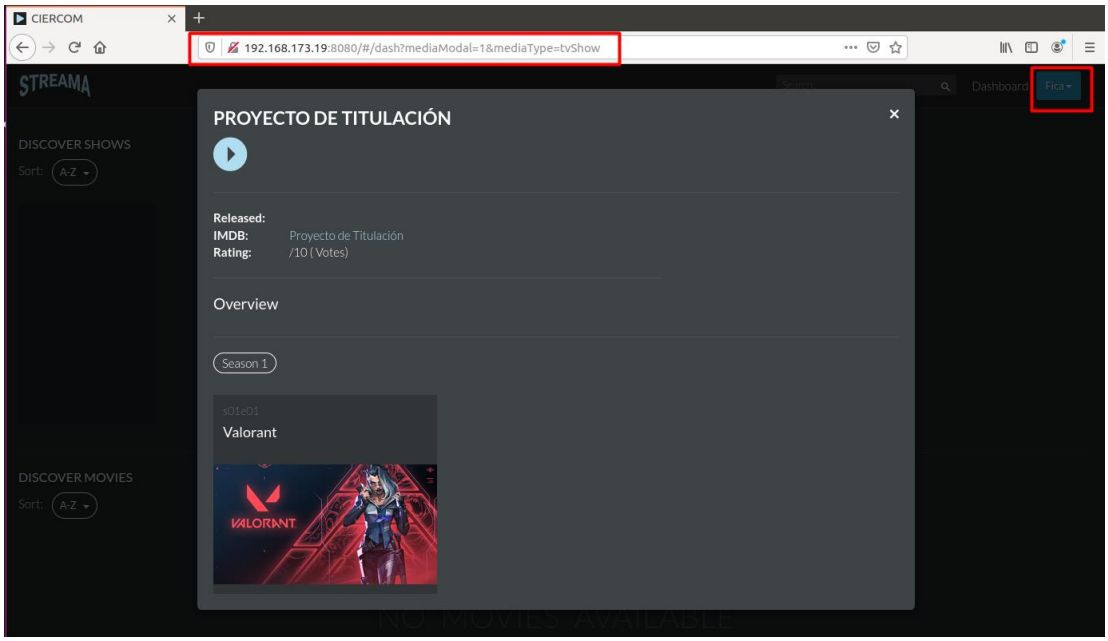
Intercambio de señales de tres vías servidor Streaming



Una vez ingresadas las credenciales de usuario en el Cliente1, a través de la Figura 95 en el recuadro rojo del lado superior derecho se comprueba que pertenecen al usuario FICA, dentro de la plataforma ya se puede disfrutar del contenido que contiene esta.

Figura 95

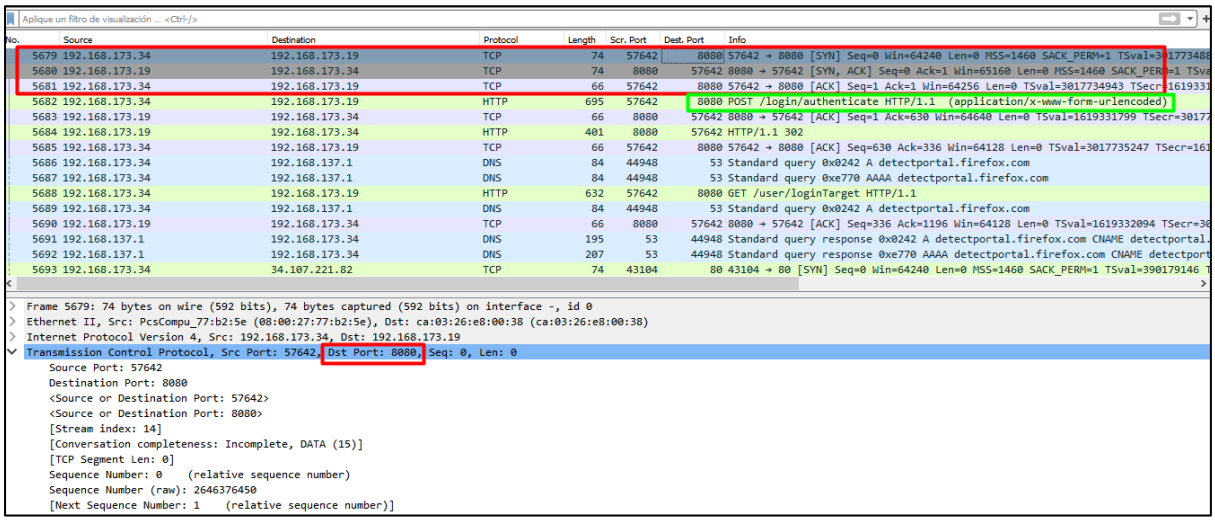
Ingreso del usuario FICA en el Cliente1



Del lado del sniffer con la captura de paquetes, en el recuadro rojo de la Figura 96 nuevamente aparece el intercambio de señales de tres vías, pero a diferencia de la Figura 94 en el recuadro verde de la Figura 96 se observa que el Cliente1 envía la autenticación de ingreso al servidor, donde finalmente éste acepta dicha solicitud con un ACK.

Figura 96

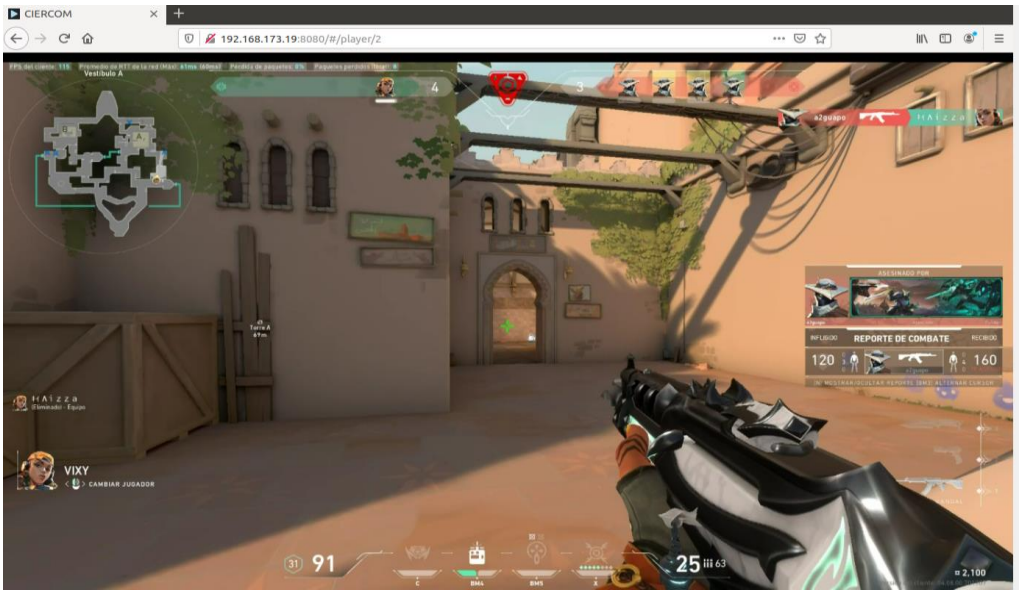
Capturas de paquetes de Login o Ingreso para el usuario FICA



Una vez carga el vídeo que se desea reproducir, en la Figura 97 se observa que el contenido se reproduce y con una gran calidad, demostrando de esta manera que el servicio que brinda el servidor Streaming funciona correctamente y que cada usuario independiente de la red LAN a la que pertenezcan lo va a disfrutar de la misma manera.

Figura 97

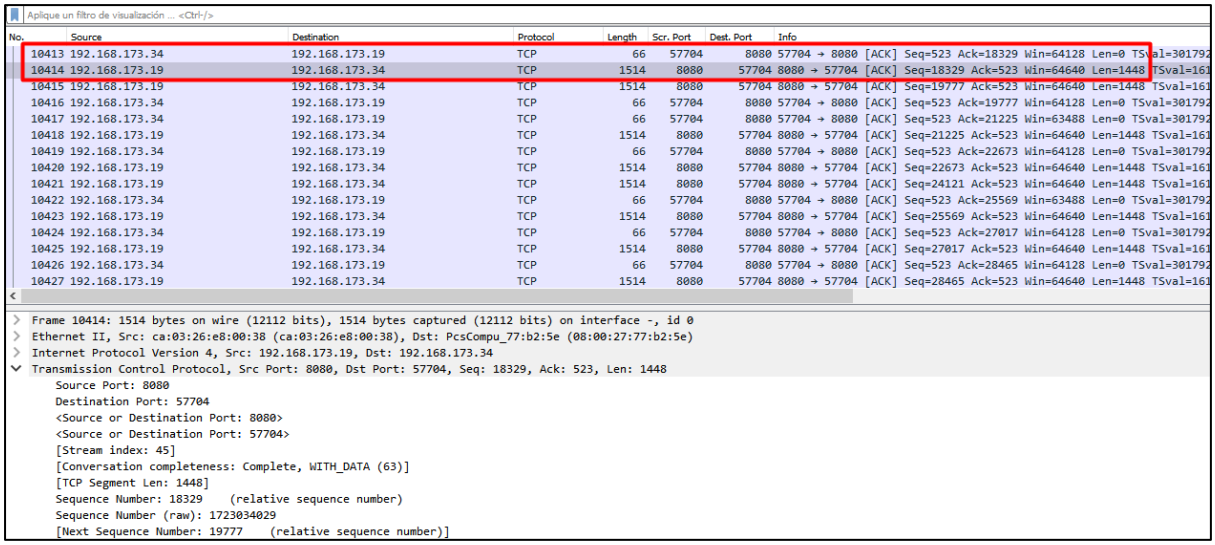
Reproducción de contenido en el Cliente1



Para finalizar con este apartado, con el sniffer se realizó la captura de paquetes mientras el contenido es reproducido, donde en la Figura 98 se observa que es un intercambio constante de paquetes ACK entre el Cliente1 y el servidor sin importar el número de secuencia que contiene cada paquete, esto se debe a que, si se hace una validación de que cada uno cumpla con la secuencia que es enviada, el contenido no sería reproducido de manera fluida y se tendría todo el momento una carga lenta, además que no sería un contenido que el usuario pueda disfrutar como se busca hacerlo.

Figura 98

Intercambio de paquetes ACK mientras el contenido es reproducido



5.5. Análisis de tablas de flujo y captura de paquetes MPLS/SDN

Primero para entender las tablas de flujo, se realiza configuración MPLS de acuerdo a lo planteado en la Tabla 9, donde se hace mención a la etiqueta designada en cada puerto, para lo cual se detallan las sentencias base para establecer dicha configuración, las mismas que se explican a continuación en la Tabla 19.

Tabla 19

Lista de sentencias OpenFlow

Sentencia	Descripción
ovs-ofctl	Es una herramienta de línea de comandos para monitorear y administrar conmutadores OpenFlow.
add-flow	Permite crear o agregar un nuevo flujo en una interfaz bridge
table=number	Permite determinar en qué tabla se va a guardar el flujo establecido
in_port=port	Coincide con el puerto de OpenFlow, que puede ser un número de puerto de OpenFlow o una palabra clave
dl_type	Coincide con el valor de forma exacta o con una máscara opcional en el campo de metadatos, estos valores son números enteros de 64 bits, de forma predeterminada en decimal (se usa un prefijo 0x para especificar hexadecimal). Para la implementación de MPLS

	se utilizan tres tipos: 0x0800 (IP), 0x8847 (MPLS) y 0x0806 (ARP)
actions=push_mpls	Permite realizar la operación push de MPLS para permitir el envió de las etiquetas al siguiente salto, es importante usar el ethertype 0x8847
actions=pop_mpls	Permite realizar la operación pop de MPLS que es para desempaquetar las etiquetas y poder tener comunicación con el resto de la red, es importante usar el ethertype 0x8847
o**utput**:port	Envía el paquete al puerto del número de puerto de OpenFlow, si si es puerto de entrada del paquete, el paquete no se envía.

Fuente: Adaptado de https://www.openvswitch.org/support/dist-docs-2.5/ovs-ofctl.8.txt

Una vez con los conceptos establecidos se procede con la configuración de cada Open vSwitch, a continuación se muestra la configuración establecida de acuerdo a las funciones POP, SWAP y PUSH de MPLS, además de las etiquetas correspondiente a cada equipo.

a) Switch 1 (S1)

```
ovs-ofctl add-flow br0
"table=0,in_port=1,dl_type=0x0800,actions=goto_table:1"

ovs-ofctl add-flow br0
"table=0,in_port=2,dl_type=0x8847,actions=goto_table:1"

ovs-ofctl add-flow br0
"table=1,in_port=1,dl_type=0x0800,actions=push_mpls:0x8847,set_field
:32->mpls_label,output:2"

ovs-ofctl add-flow br0
"table=1,in_port=2,dl_type=0x8847,mpls_bos=0,actions=pop_mpls:0x8847
,resubmit(,1)"

ovs-ofctl add-flow br0
"table=1,in_port=2,dl_type=0x8847,mpls_bos=1,actions=pop_mpls:0x0800
,output:1"

ovs-ofctl add-flow br0
"table=0,in_port=1,dl_type=0x0806,actions=output:2"
```

```
ovs-ofctl add-flow br0
"table=0,in_port=2,dl_type=0x0806,actions=output:1"
```

b) Switch 2 (S2)

```
ovs-ofctl add-flow br0
"table=0,in_port=1,dl_type=0x8847,actions=goto_table:1"

ovs-ofctl add-flow br0
"table=0,in_port=3,dl_type=0x8847,actions=goto_table:1"

ovs-ofctl add-flow br0
"table=1,in_port=3,dl_type=0x8847,actions=push_mpls:0x8847,set_field
:23->mpls_label,push_mpls:0x8847,set_field:100->mpls_label,output:1"

ovs-ofctl add-flow br0
"table=1,in_port=1,dl_type=0x8847,actions=push_mpls:0x8847,set_field
:21->mpls_label,push_mpls:0x8847,set_field:200->mpls_label,output=3"

ovs-ofctl add-flow br0
"table=0,in_port=3,dl_type=0x0806,actions=output:1"

ovs-ofctl add-flow br0
"table=0,in_port=1,dl_type=0x0806,actions=output:3"
```

c) Switch 3 (S3)

```
ovs-ofctl add-flow br0
"table=0,in_port=1,dl_type=0x0800,actions=goto_table:1"

ovs-ofctl add-flow br0
"table=0,in_port=2,dl_type=0x8847,actions=goto_table:1"

ovs-ofctl add-flow br0
"table=1,in_port=1,dl_type=0x0800,actions=push_mpls:0x8847,set_field
:10->mpls_label,output:2"

ovs-ofctl add-flow br0
"table=1,in_port=2,dl_type=0x8847,mpls_bos=0,actions=pop_mpls:0x8847
,resubmit(,1)"
```

```
ovs-ofctl add-flow br0
"table=1,in_port=2,dl_type=0x8847,mpls_bos=1,actions=pop_mpls:0x0800
,output:1"

ovs-ofctl add-flow br0
"table=0,in_port=1,dl_type=0x0806,actions=output:2"

ovs-ofctl add-flow br0
"table=0,in_port=2,dl_type=0x0806,actions=output:1"
```

Con la configuración implementada se procede al análisis de las tablas de flujo para corroborar el funcionamiento de MPLS, hay que aclarar que el plano de datos está en los Open vSwitch, pero el encargado de realizar todo el proceso de conmutación y distribución de los flujos es el controlador Open Daylight a través de los módulos instalados anteriormente.

Las Figuras 99-101 por medio del comando **`ovs-ofctl dump-flows br0`**, muestran en la consola todas las entradas de flujo en las tablas del switch que coinciden con los flujos MPLS de S1, S2 y S3.

El recuadro verde hace énfasis al flujo de paquetes que atraviesan la nube respetando las sentencias que están establecidas para cada puerto, y además con esto comprobar que se encuentra operativo el protocolo MPLS en la red SDN; pero analizando a detalle cada línea de S1 se tiene:

- En el puerto 1 y 2, vuelve a buscar la tabla de flujo OpenFlow con el campo in_port reemplazado por puerto y ejecuta las acciones encontradas, si las hay, además de cualquier otra acción en esta entrada de flujo .
- Al momento de realizar la función push MPL,S todos los paquetes que vienen del puerto 1 pasan al puerto 2 a través de la etiqueta 32.
- Finalmente cuando realiza la función pop MPLS, todas las etiquetas que vienen del puerto 2 se desencapsulan y pasan hacia el puerto 1 para que los paquetes puedan llegar a su destino y así poder acceder a los difetentes servicios de la DMZ.

Lo resaltado en color rojo de las Figuras 99-101, hace mención al puerto que está asignado al flujo programado y que a través de este sale los datos al siguiente salto de la red, como se muestra en cada uno de los Open vSwitch las etiquetas se conmutan entre puertos de acuerdo a las configuraciones que se han establecido.

Figura 99

Tablas de flujo en S1

Figura 100

Tablas de flujo en S2

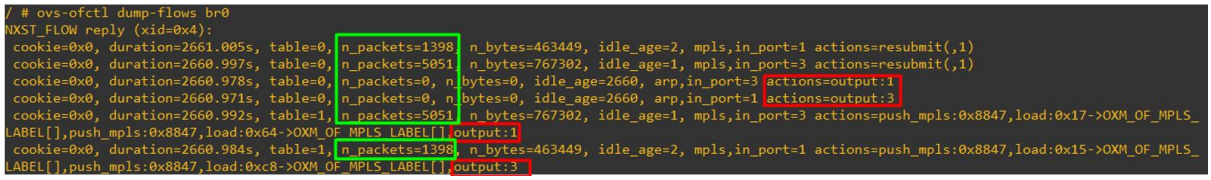
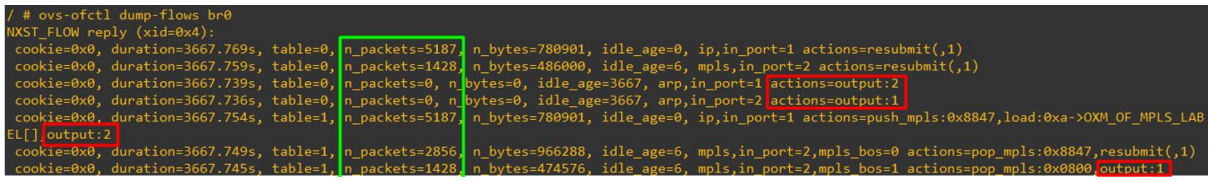


Figura 101

Tablas de flujo en S3



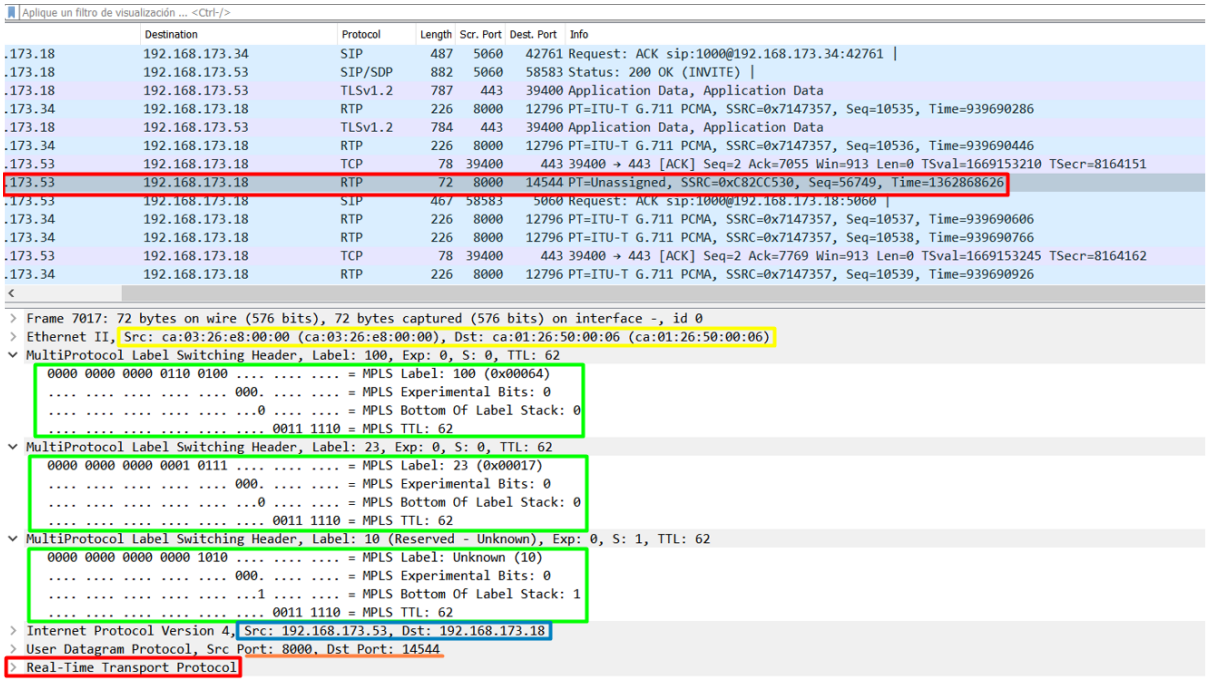
Para comprobar lo que se tiene en las tablas de flujo, por medio del sniffer se captura el tráfico de la red MPLS, la Figura 102 proporciona toda la información requerida para verificar que en realidad MPLS se encuentra operativo, esta captura es durante una llamada telefónica del Cliente2 al Cliente1.

Los recuadros de color rojo muestra el protocolo RTP que se captura con el sniffer, el recuadro de color amarillo detalla la capa de enlace de datos, es decir, la dirección MAC de origen y la MAC de destino, los recuadros de color verde son los más importantes para analizar, estos muestran las etiquetas MPLS (100,23,10), así se puede verificar que la pila de etiquetas establecida está trabajando de acuerdo a lo diseñado en la Tabla 9, la misma que menciona una función swap MPLS entre S2 y S1.

Continuando con el análisis del paquete capturado, el recuadro de color azul muestra la capa de Red, además de la dirección IP origen y dirección IP destino; por último lo subrayado en color marrón hace mención a la capa transporte y los puertos que operan en esta capa.

Figura 102

Captura de paquetes RTP aplicando MPLS



Las Tablas 15-17 tomando en cuenta las etiquetas 100,23,10; desglosan la estructura del shim header MPLS obtenido en la Figura 102, los valores que contiene cada campo se presentan en formato decimal y en formato binario, corroborarando el tamaño en bits de cada campo según lo que se planteó en la sección 2.1.2 acerca de las etiquetas.

Tabla 20

Shim Header etiqueta 100

Campo (bits) / Formato	Etiqueta (20 bits)	EXP (3 bits)	S (1 bit)	TTL (8 bits)
Decimal	100	0	0	62
Binario	0000 0000 0000 0110 0100	000	0	0011 1110

Tabla 21

Shim Header etiqueta 23

Campo (bits) / Formato	Etiqueta (20 bits)	EXP (3 bits)	S (1 bit)	TTL (8 bits)
Decimal	23	0	0	62
Binario	0000 0000 0000 0001 0111	000	0	0011 1110

Tabla 22

Shim Header etiqueta 10

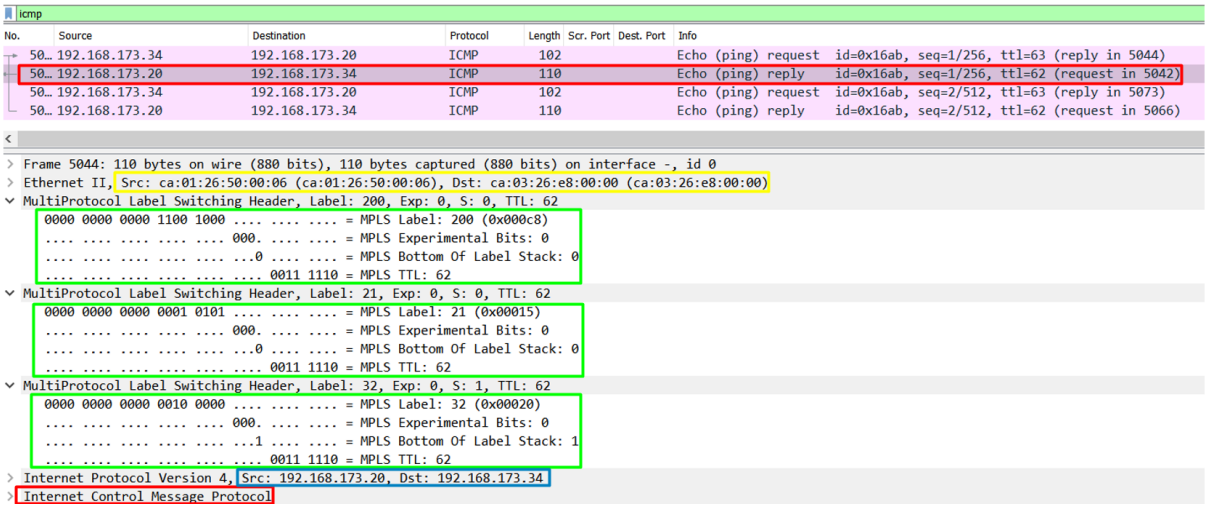
Campo (bits) / Formato	Etiqueta (20 bits)	EXP (3 bits)	S (1 bit)	TTL (8 bits)
Decimal	10	0	0	62
Binario	0000 0000 0000 000 1010	000	0	0011 1110

La Figura 103 muestra una captura de tráfico en el enlace de S3 – S2, comprobando una vez más que el protocolo MPLS se encuentra operativo, esta captura de paquetes ICMP se realizó durante un ping entre Cliente1 a Controlador.

Los reacuadros de color rojo hacen énfasis al protocolo ICMP, mediante el intercambio de paquetes echo request y echo reply, el recuadro de color amarillo detalla la capa de enlace de datos con la dirección MAC de origen y la MAC de destino, los recuadros de color verde muestran las diferentes etiquetas de la pila MPLS, finalmente el recuadro de color azul proporciona la información de la capa de red con su dirección IP origen e IP destino.

Figura 103

Captura de paquetes ICMP aplicando MPLS



Las Tablas 18-20 tomando en cuenta las etiquetas 200,21,32; desglosan la estructura del shim header MPLS obtenido en la Figura 103, estableciendo los valores que contiene cada campo presentados en formato decimal y en formato binario.

Tabla 23

Shim Header etiqueta 200

Campo (bits) / Formato	Etiqueta (20 bits)	EXP (3 bits)	S (1 bit)	TTL (8 bits)
Decimal	200	0	0	62
Binario	0000 0000 0000 1100 1000	000	0	0011 1110

Tabla 24

Shim Header etiqueta 21

Campo (bits) / Formato	Etiqueta (20 bits)	EXP (3 bits)	S (1 bit)	TTL (8 bits)
Decimal	21	0	0	62
Binario	0000 0000 0000 0001 0101	000	0	0011 1110

Tabla 25

Shim Header etiqueta 32

Campo (bits) / Formato	Etiqueta (20 bits)	EXP (3 bits)	S (1 bit)	TTL (8 bits)
Decimal	32	0	1	62
Binario	0000 0000 0000 0010 0000	000	1	0011 1110

5.6. Marcaje asociado para frontera de confianza

Para visualizar el marcaje aplicado en la frontera de confianza, La Figura 104 con el comando **`policy-map`** muestra el mapa de políticas, el cual, contiene las clases y el marcaje DSCP asociado a estas de acuerdo con su servicio, en base a las configuraciones realizadas en la sección 4.4.1.

Figura 104

Mapa de Políticas de Servicio con marcaje DSCP

```
FICA#show policy-map
  Policy Map front-conf
    Class VOZ
      set ip dscp ef
    Class SignaVOip
      set ip dscp cs3
    Class SignaSTREAM
      set ip dscp af43
    Class FTP
      set ip dscp af33
    Class WEB
      set ip dscp af23
```

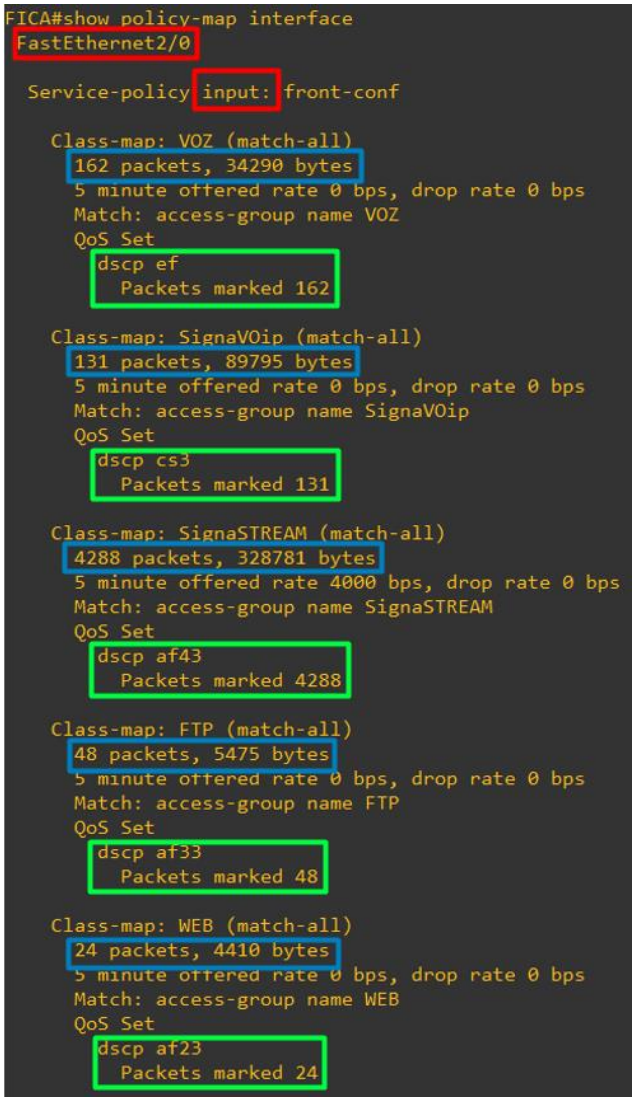
Continuando con el análisis, la Figura 105 por medio del comando **`show policy-map interface`,** proporciona información detallada sobre la frontera de confianza aplicada en el router FICA, los recuadros en color rojo muestran la interfaz FastEthernet 2/0 y la dirección en la que se aplica la configuración, la cual es de entrada o input.

Los recuadros de color azul hacen énfasis en el tráfico que circula por cada clase, detalla el número de paquetes y la cantidad de bytes de información, además como era de esperarse el servicio de Streaming es el que más paquetes y bytes genera a diferencia de los otros servicios.

Por último, los recuadros de color verde muestran el marcaje DSCP de cada clase y además el total de paquetes que se marcan, como se puede apreciar cada paquete de información que sale de la red es marcado al realizar match con el marcaje DSCP que le corresponde, dando como resultado que la frontera de confianza funciona correctamente.

Figura 105

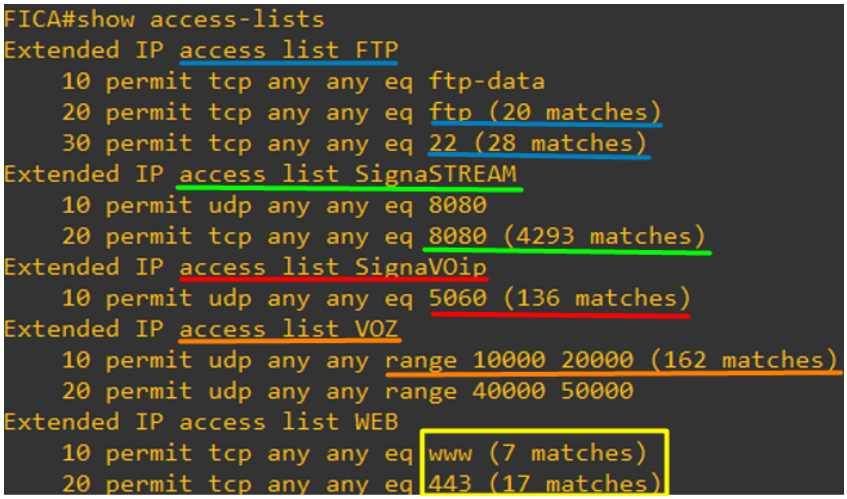
Validación de la Frontera de Confianza en el router FICA



La Figura 106 el comando **`show access-lists`**, muestra las ACL's establecidas de acuerdo con el puerto de operación de cada servicio, en color azul se resalta el servicio FTP y SFTP, con color verde se resalta el servicio de Streaming, para la señalización de VoIP es el color rojo y para la transmisión de voz el color naranja, finalmente en color amarrillo se señala el servicio web y web seguro. Se valida que los matches se están realizando correctamente con el puerto que le pertenece al tráfico generado en la red LAN FICA.

Figura 106

Validación de matches en las ACL's de la Frontera de Confianza

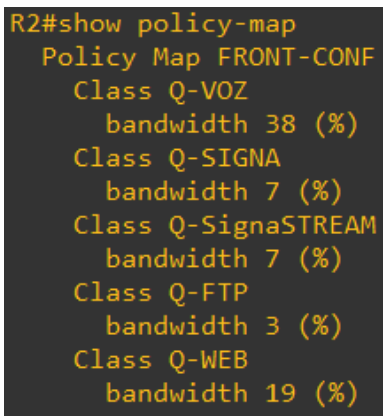


5.7. Marcaje asociado para teoría de colas

Para visualizar el encolamiento aplicado, La Figura 107 con el comando **`policy-map`** muestra el mapa de políticas, el cual, contiene las clases y el ancho de banda establecido de acuerdo con lo calculado para cada servicio, en base a las configuraciones realizadas en la sección 4.5.4 y que se aplica en R2 al tener la interfaz más cercana a la DMZ.

Figura 107

Mapa de políticas con encolamiento CBWFQ

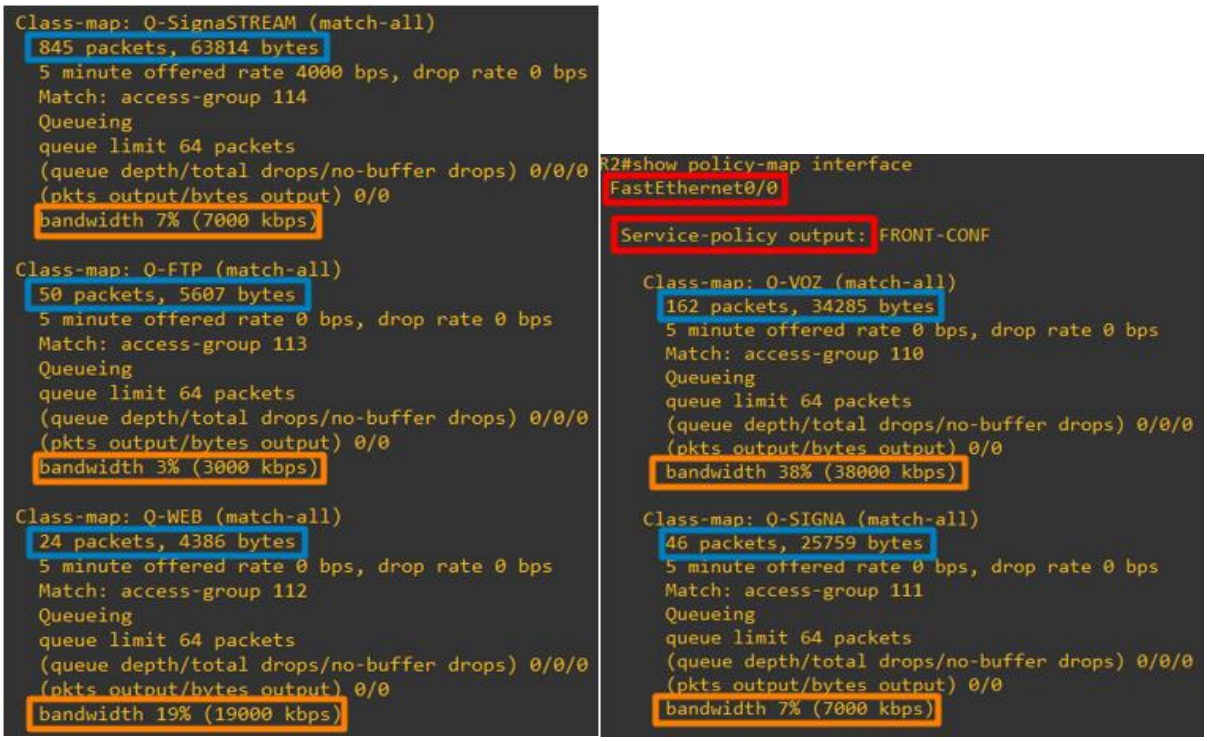


La Figura 108 a través del comando **`show policy-map interface`,** muestra información detallada de la teoría de colas aplicada en el router R2, los recuadros en color rojo muestran la interfaz FastEthernet 0/0 y la dirección en la que se aplica la configuración, la cual es de output o salida, esto se debe a que se deben tratar con políticas de QoS a todas las peticiones provenientes de las redes LAN hacia la DMZ.

En los recuadros de color azul muestran el tráfico que circula por cada clase, detalla el número de paquetes y la cantidad de bytes de información, además como era de esperarse al igual que en la frontera de confianza el servicio de Streaming es el que más paquetes y bytes genera con respecto a los otros servicios. Por último, los recuadros de color naranja muestran el encolamiento CBWFQ con el ancho de banda calculado y establecido para cada servicio.

Figura 108

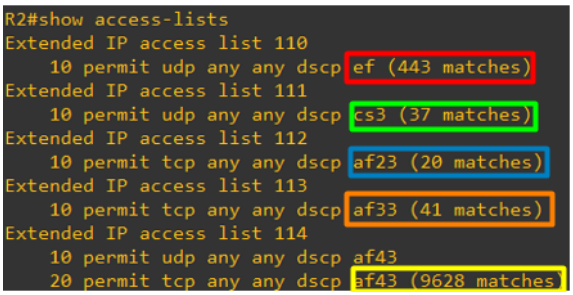
Teoría de colas con encolamiento CBWFQ



La Figura 109 con el comando **`show access-lists`,** muestra las ACL's establecidas de acuerdo con el marcaje DSCP de cada servicio, en color rojo se resalta el servicio de voz con marcaje **ef**, con color verde el servicio de señalización de VoIP con marcaje **cs3**, el color azul para el servicio web y web seguro con marcaje **af23** el color naranja, finalmente en color amarrillo se señala el servicio de Streaming con marcaje **af43**. Validando de esta manera con los matches, que los paquetes que vienen marcados desde las redes LAN hacia la DMZ, están siendo procesados con políticas de QoS y con administración eficiente del ancho de banda.

Figura 109

Validación de matches en las ACL's para Teoría de Colas



5.8. Captura de paquetes con marcaje DSCP y validación de resultados

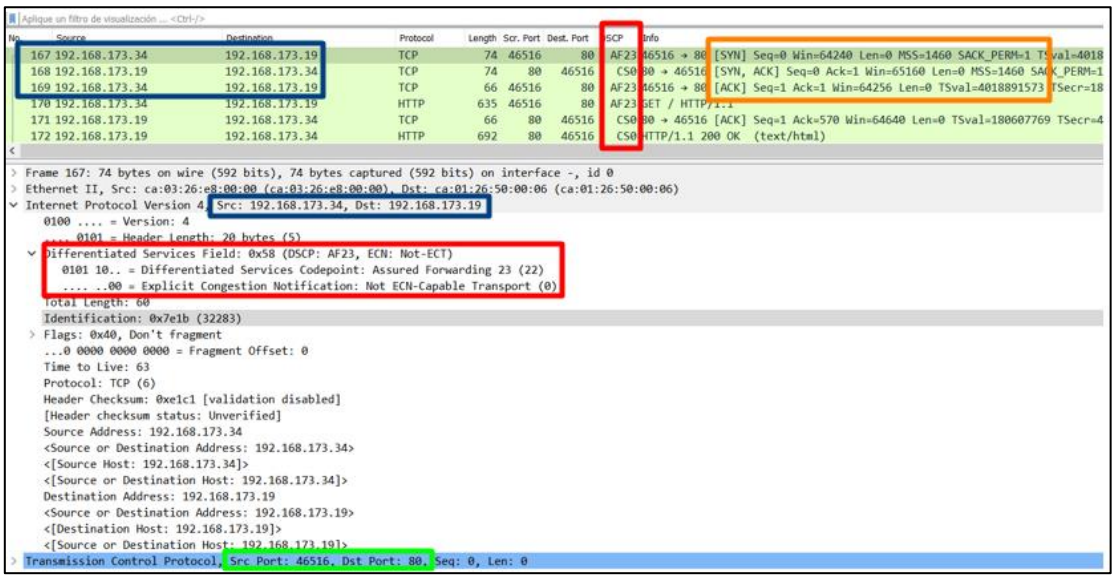
5.8.1. Servicio HTTP y HTTPS

La Figura 110 muestra una captura del tráfico HTTP con puerto 80, el cual se generó al acceder al servicio web desde el Cliente1, con recuadro azul se marcan las direcciones IP de origen que es la .34 del Cliente1 y la dirección IP destino del servidor Web que es la .19, cumpliendo con el intercambio de señales de tres vías resaltado en el cuadro naranja.

Gracias a las herramientas que brinda el sniffer de Wireshark, permite capturar el marcaje DSCP aplicado al servicio, en este caso al tratarse de web se aplica AF23 y como se muestra en los recuadros de color rojo es aplicado de acuerdo con las configuraciones que se realizaron, con esto se demuestra que ya existen políticas de QoS en la red LAN FICA. Finalmente, el recuadro verde proporciona la información de los puertos de operación, el más importante el puerto 80 que proporciona el servidor Web.

Figura 110

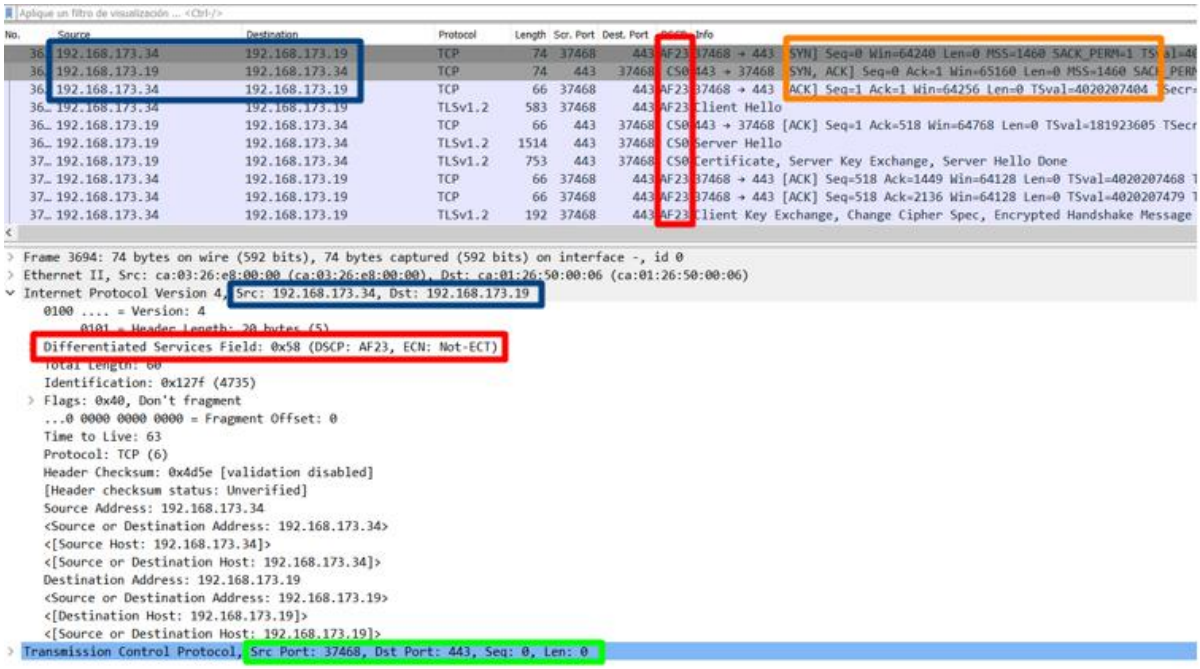
Tráfico HTTP con marcaje DSCP AF23



La Figura 111 muestra al igual que HTTP, que las políticas de QoS al ser establecidas de acuerdo con el puerto de operación, funcionan de la misma manera con HTTPS puerto 443 con el marcaje DSCP AF23 (recuadro rojo), de dirección IP origen Cliente1 y dirección IP destino servidor Web (recuadro azul), cumpliendo con el intercambio de señales de tres vías (recuadro naranja), además de la validación del certificado para el sitio web.

Figura 111

Tráfico HTTPS con marcaje DSCP AF23



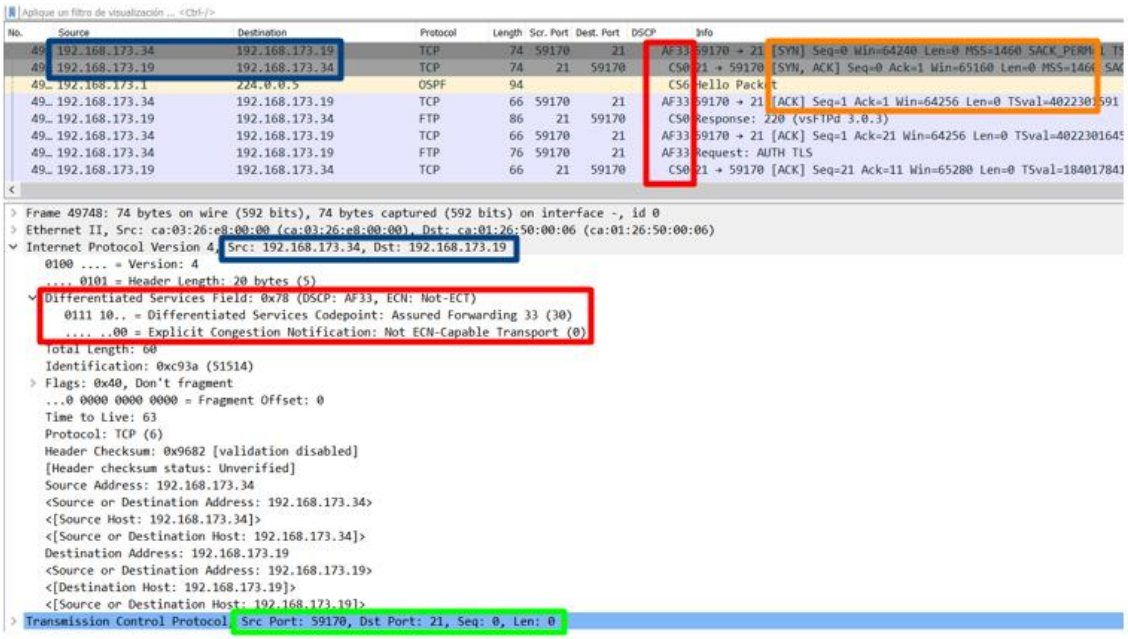
5.8.2. Servicio FTP y SFTP

La Figura 112 muestra una captura del tráfico FTP con puerto 21, generado a través de la aplicación FileZilla al momento de conectarse el Cliente1 con el servidor FTP, con recuadro azul se marcan las direcciones IP de origen que es la .34 del Cliente1 y la dirección IP destino del servidor FTP que es la .19, cumpliendo con el intercambio de señales de tres vías resaltado en el cuadro naranja.

En este caso al tratarse de FTP se aplica un marcaje DSCP AF33 y como se muestra en los recuadros de color rojo se aplica de acuerdo con las configuraciones establecidas previamente, demostrando así que las políticas de QoS para FTP cumplen su función. Finalmente, el recuadro verde proporciona la información de los puertos de operación, el más importante el puerto 21 que proporciona el servidor FTP.

Figura 112

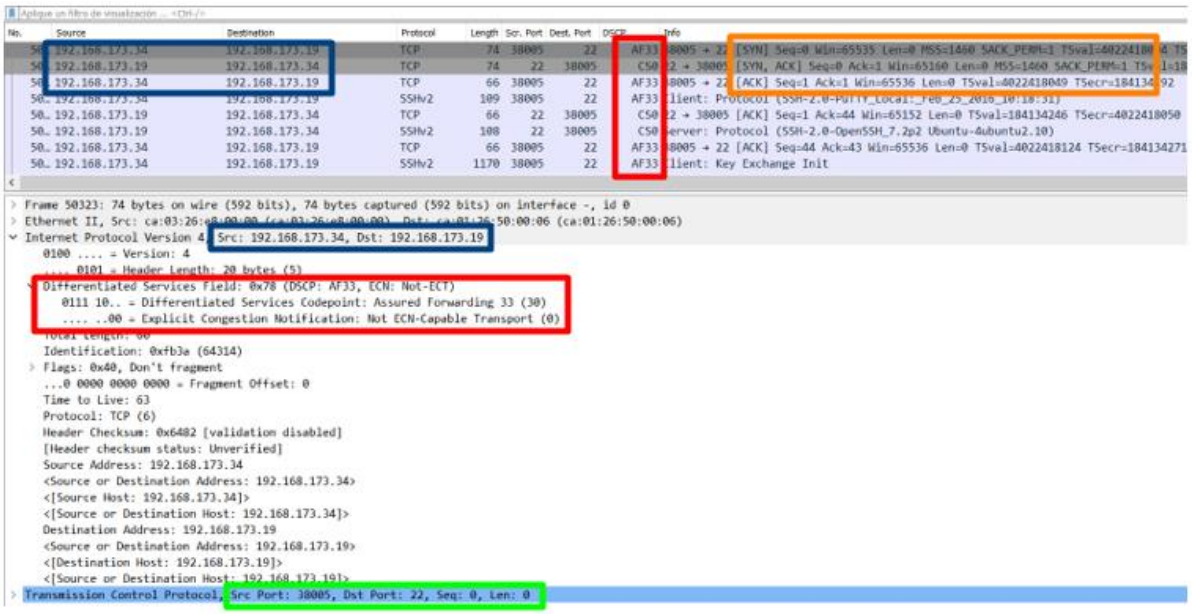
Tráfico FTP con marcaje DSCP AF33



La Figura 113 muestra que las políticas de QoS al ser establecidas de acuerdo con el puerto de operación, funcionan de la misma manera con SFTP puerto 22 con el marcaje DSCP AF33 (recuadro rojo), de dirección IP origen Cliente1 y dirección IP destino servidor FTP (recuadro azul), sin olvidarse del intercambio de señales de tres vías (recuadro naranja) para establecer la comunicación.

Figura 113

Tráfico SFTP con marcaje DSCP AF33



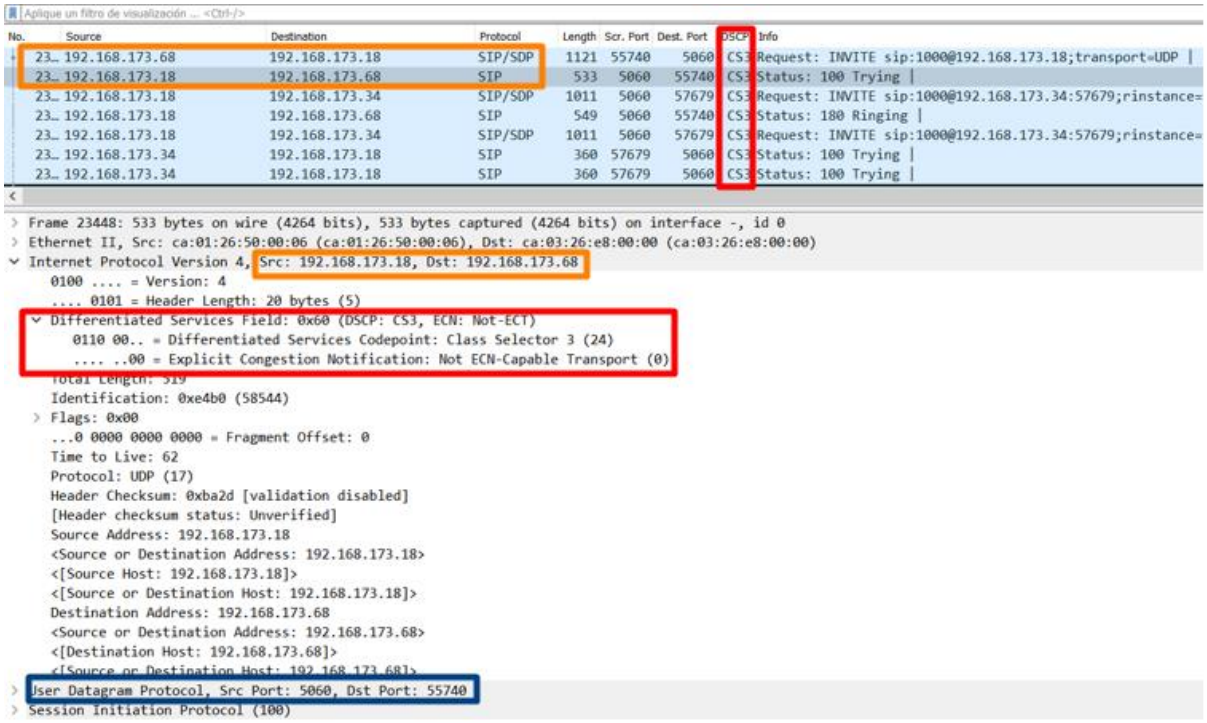
5.8.3. Servicio VoIP

Para el servicio de VoIP se compone de dos partes: En la primera, la Figura 114 muestra el marcaje DSCP para el proceso de señalización con el protocolo SIP, la captura de paquetes se realizó al momento de establecer una llamada entre el Cliente3 hacia el Cliente1. En el recuadro de color naranja se muestra las direcciones IP origen .68 (Cliente3) y dirección IP destino .18 (Servidor VoIP), esto cuando se realiza el intercambio de señales para establecer la llamada.

En el recuadro rojo se muestra el marcaje DSCP CS3, establecido para el protocolo SIP con puerto 5060 que se resalta en el recuadro de color azul, una vez verificado que el marcaje está establecido, se puede afirmar que las políticas de QoS para VoIP en la red LAN FACAE operan correctamente para diferenciación de servicios.

Figura 114

Tráfico SIP con marcaje DSCP CS3

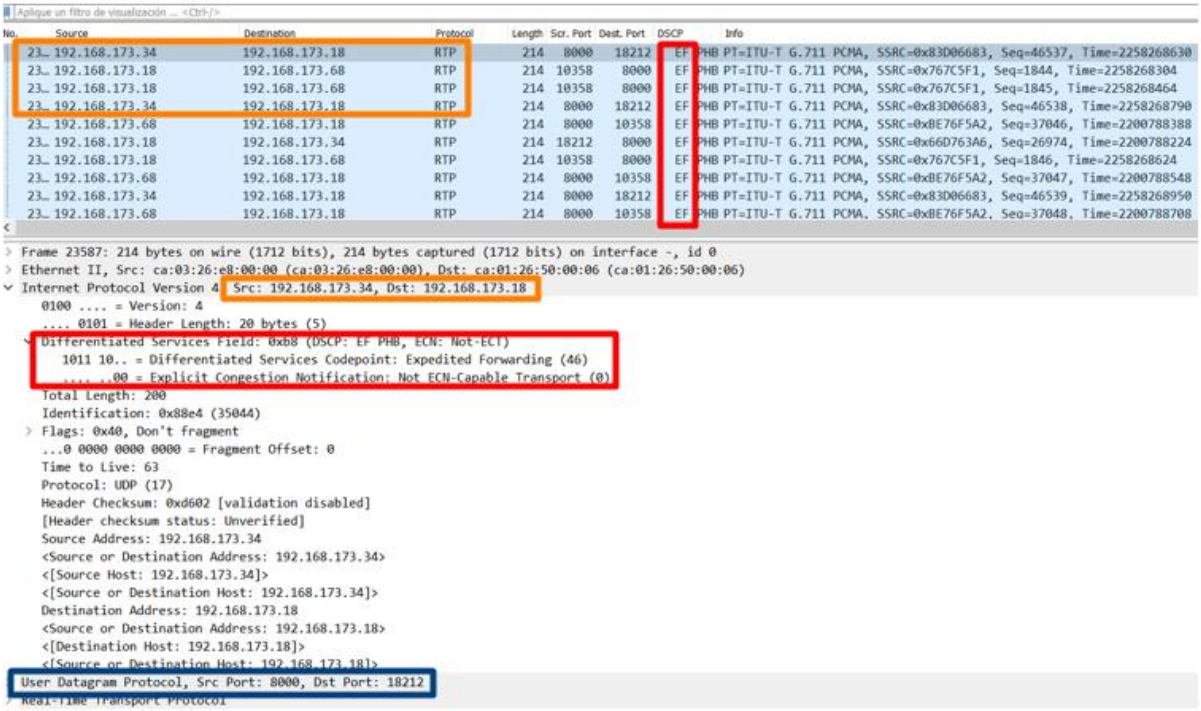


Para la segunda parte, la Figura 115 muestra el marcaje DSCP aplicado para voz con el protocolo RTP, el recuadro naranja muestra el proceso de intercambio de direcciones IP entre el Cliente3 al servidor VoIP y Cliente1 al servidor VoIP al momento de establecerse la llamada y empezar la comunicación entre las extensiones.

En el recuadro rojo se muestra el marcaje DSCP EF aplicado para el protocolo RTP, el cual opera correctamente al haber designado un rango de puertos en las políticas QoS, esto debido a que se tratan de puertos dinámicos, a diferencia de SIP que utiliza un puerto estático. Además, se puede apreciar que todos y cada uno de los paquetes RTP son marcados sin excepción durante la llamada.

Figura 115

Tráfico RTP con marcaje DSCP EF



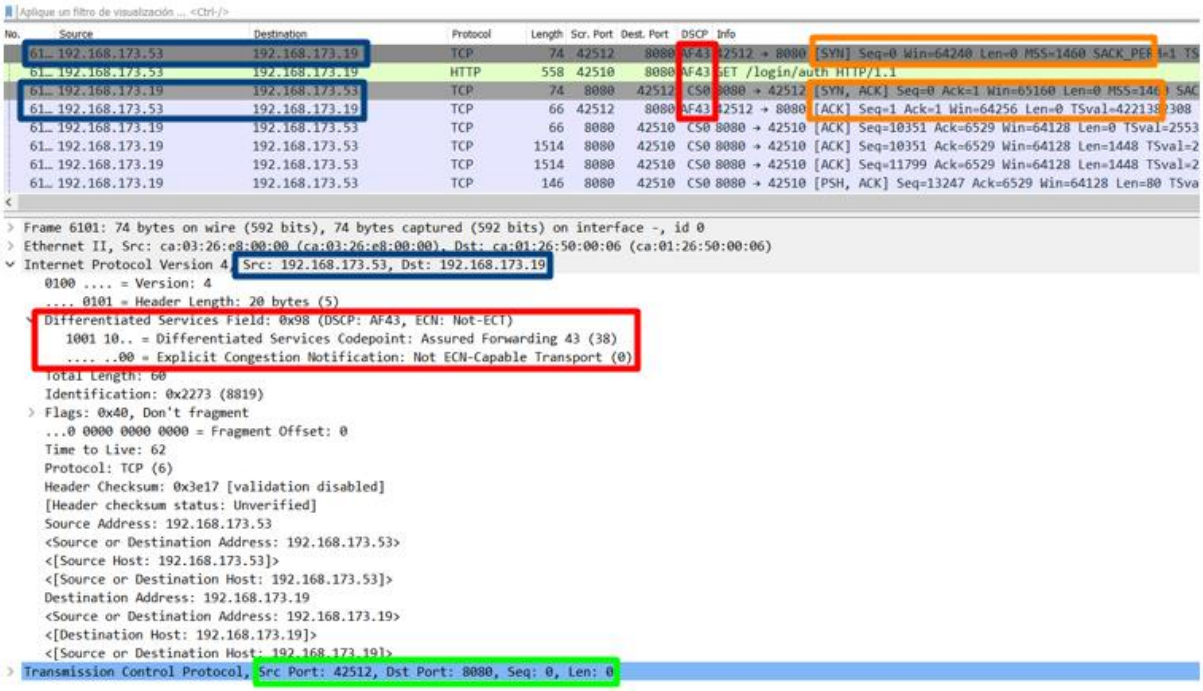
5.8.4. Servicio Streaming

La Figura 116, muestra una captura de tráfico TCP al momento de acceder a la página de Streama desde el Cliente2, en el recuadro azul se muestra el intercambio de direcciones IP origen y destino, y aunque en un principio pareciese que se accede a un sitio web, al trabajar el servidor de Streaming con el puerto 8080, una vez más las políticas de QoS para diferenciar servicios son las encargadas de establecer el marcaje DSCP AF43, establecido anteriormente y que se valida en el recuadro de color rojo, demostrando que los paquetes son marcados al salir de la red LAN FICAYA.

En el recuadro amarillo se puede apreciar el intercambio de tres vías al momento de acceder al sitio web de Streama, finalmente en el recuadro verde se corrobora que el servicio trabaja con el puerto 8080 definido por el servidor de Streaming.

Figura 116

Tráfico TCP del servicio de Streaming con marcaje DSCP AF43

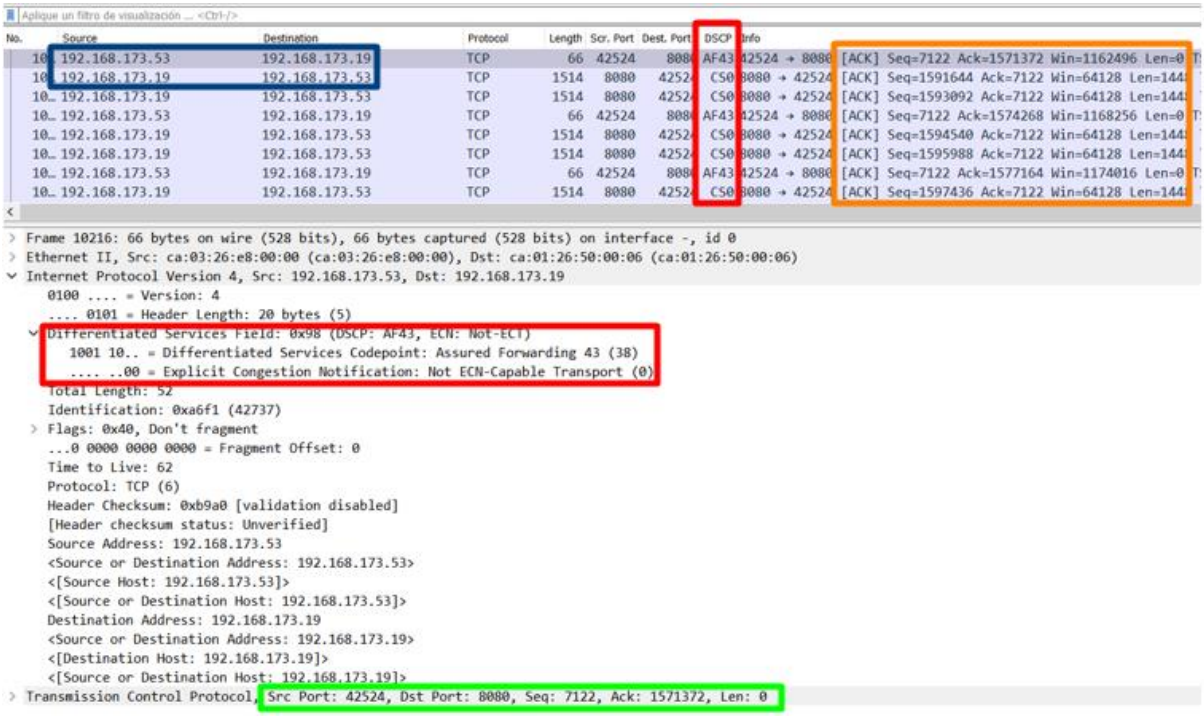


Continuando con el análisis del servicio de Streaming, la Figura 117 muestra una captura de paquetes al momento de reproducir el contenido de la plataforma Streama en el Cliente2, como se puede apreciar en el recuadro azul se intercambian mutuamente los paquetes entre la IP origen Cliente2 con la IP destino Servidor Streaming y viceversa, en el recuadro naranja se muestra el intercambio constante de paquetes ACK, algo a resaltar es que cada paquete tiene un número de secuencia, la cual debe coincidir con el ACK de respuesta a dicha secuencia.

En el recuadro de color rojo, se puede apreciar que se realiza el marcaje DSCP AF43 establecido para Streaming, eso sí solamente los paquetes que salen de la LAN FICAYA son marcados, los paquetes provenientes del servidor hacia la red LAN no se encuentran marcados, y finalmente en el recuadro de color verde se muestra el puerto 8080 que establece el servidor de Streaming y además aquí se puede ver el número de secuencia y la identificación del ACK.

Figura 117

Intercambio de paquetes ACK en el servicio de Streaming con marcaje DSCP AF43



5.8.5. Validación de resultados

Entre todas las utilidades que brinda el sniffer de Wireshark, se destaca la herramienta que permite analizar el flujo de tráfico TCP mediante gráficas, la Figura 118 proporciona a detalle el intercambio de paquetes, desde que se solicita el acceso al sitio web a través de HTTP puerto 80 con IP origen Cliente1 e IP destino servidor Web remarcadas en el recuadro naranja. Esta gráfica, además proporciona de manera más clara el intercambio de señales de tres vías para acceder al sitio web y también el intercambio de señales de cuatro vías al finalizar la conexión.

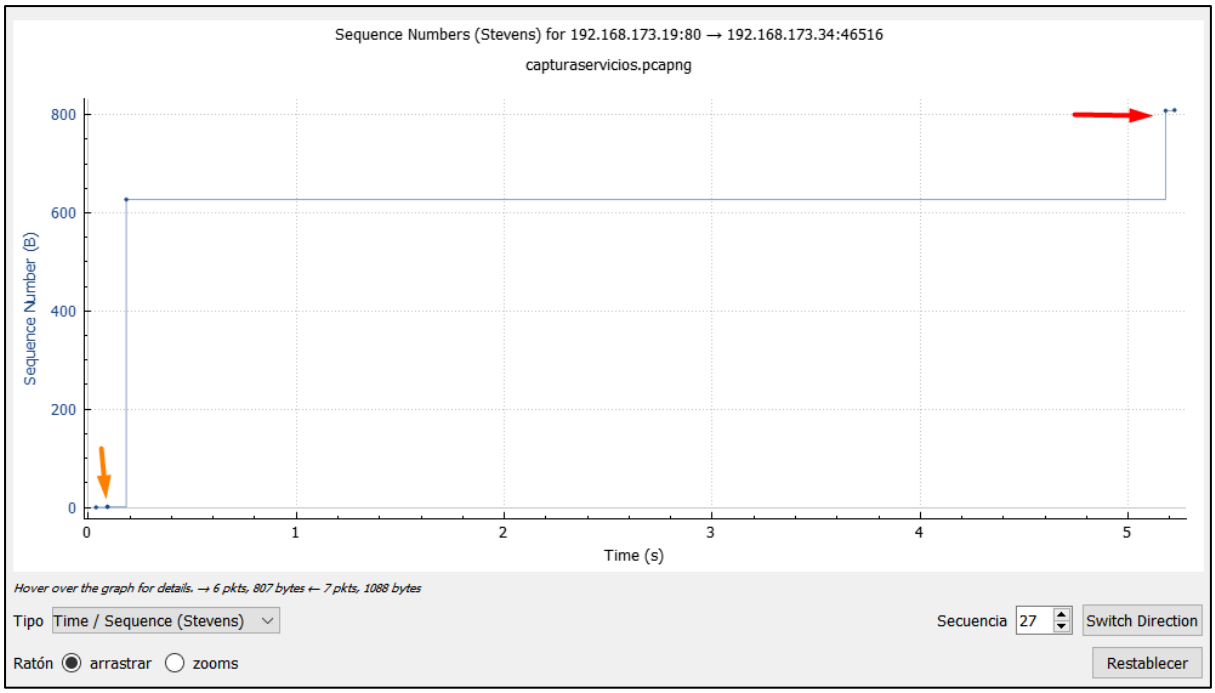
Figura 118

Secuencia de paquetes al acceder al servicio Web

Una vez se tiene la secuencia de paquetes a analizar, es necesario determinar el tiempo que tarda en cargar el contenido del sitio Web, para esto la Figura 119 muestra la gráfica de flujo TCP con secuencia de tiempo (Stevens), esta herramienta es un gráfico simple del número de secuencia de TCP a lo largo del tiempo, por lo tanto se identifica con la flecha naranja la secuencia cuando el contenido de la página web carga por completo y en la flecha roja es el momento en que se establece el cierre de conexión por parte del servidor Web.

Figura 119

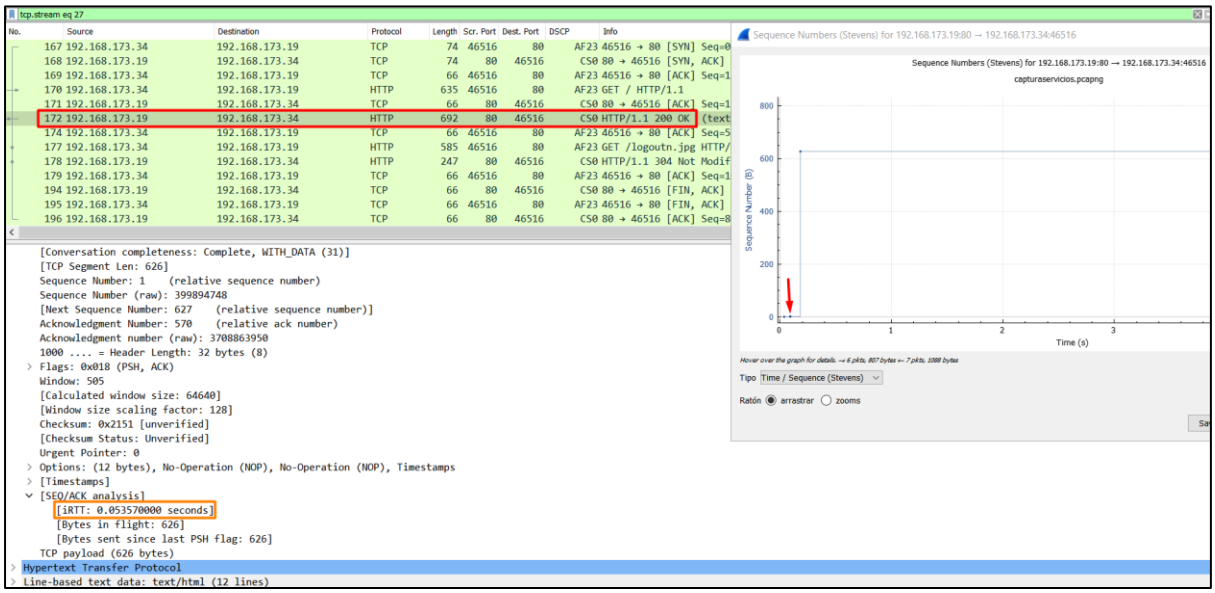
Gráfica de flujo TCP con secuencia de tiempo (Stevens)



La Figura 120, proporciona la información del paquete señalado con la flecha roja y como aprecia pertenece al paquete remarcado en el recuadro rojo, este paquete contiene el código 200 OK que significa que el contenido a cargado correctamente, desglosando la información de la capa transporte se encuentra el apartado iRTT, el cual brinda tiempo desde que se hizo la solicitud al servidor Web y que tiene un tiempo de 53ms resaltado en el recuadro naranja; con este tiempo se demuestra que la red híbrida SDN/MPLS brinda tiempos de carga bastante cortos en el servicio web.

Figura 120

Tiempo de carga del servicio Web



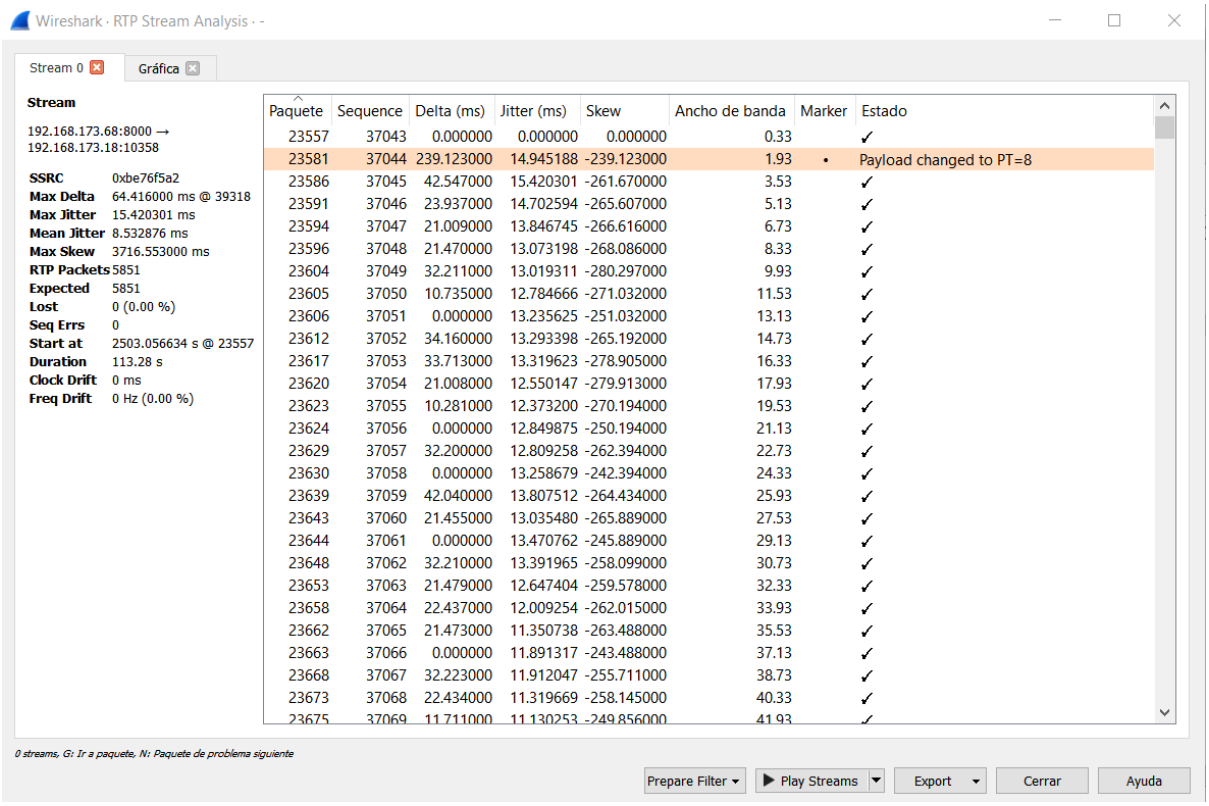
En el caso del servicio de VoIP, el sniffer Wireshark permite analizar el flujo RTP, la Figura 116 muestra el análisis de un Stream RTP al momento de establecer la llamada entre el Cliente3 y el Cliente1. Para determinar el QoS del servicio de VoIP se debe tener en cuenta los valores delta y jitter, los cuales se presentan a continuación.

- ***Max Delta:*** El delta es la diferencia de tiempo entre el paquete actual y el paquete anterior en el flujo, este presenta un valor de 64,41ms.
- ***Max Jitter – Mean Jitter:*** El Jitter o fluctuación de retardo es la variabilidad temporal durante el envío de señales digitales, un retraso de unos 30 ms o más puede ocasionar distorsiones e interrupciones en una llamada, si se analiza ambos campos se puede apreciar que el máximo jitter es de 15,42ms y el jitter promedio es de 8,53ms, demostrando que con esto se tiene una gran calidad en llamadas VoIP.

En los demás apartados que muestra la Figura 121, se destaca que todos los paquetes RTP generados no han sufrido pérdidas durante la duración de 113 segundos que tuvo la llamada entre el Cliente3 al Cliente1.

Figura 121

Análisis RTP Stream

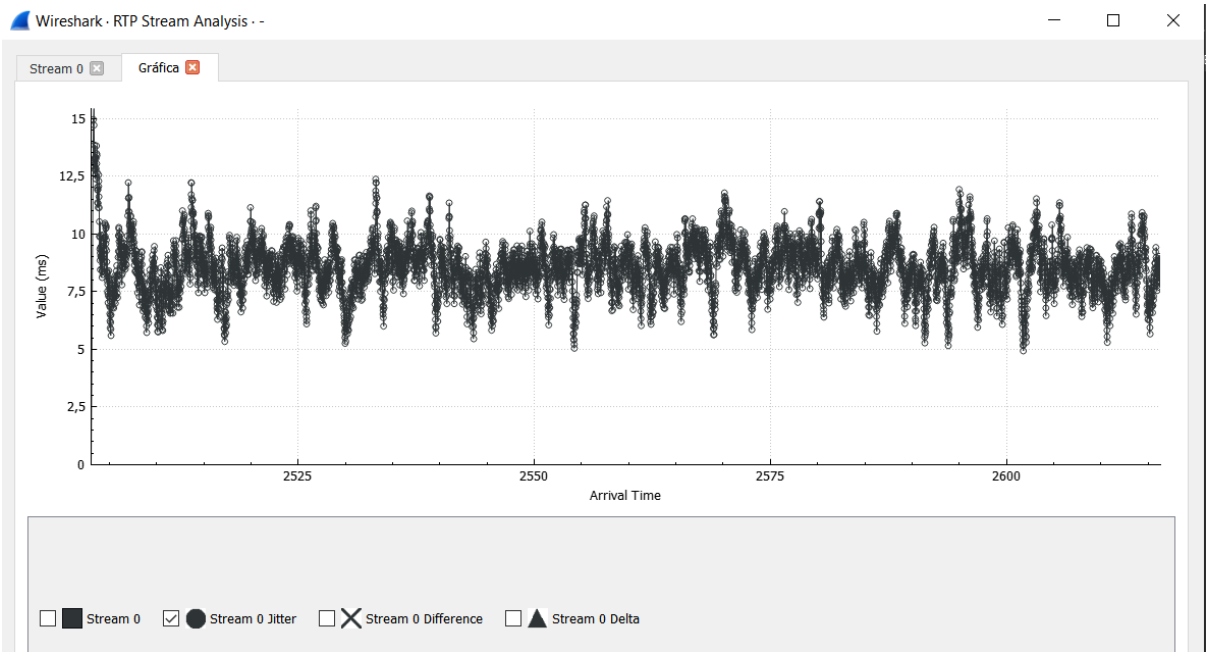


Finalmente, el análisis de Stream RTP permite mostrar una gráfica del flujo RTP, el cual se hace énfasis en el jitter que es de vital importancia al momento de diagnosticar si el servicio presenta problemas.

La Figura 122 proporciona la gráfica del jitter y sus variaciones en el transcurso de la llamada, se aprecia que al momento de iniciar la llamada ocurre el máximo jitter, pero luego en el transcurso de la llamada el jitter va de entre 5ms a 10ms, aunque en ocasiones llegue a 12,5ms. Por medio de esta gráfica se valida que la red SDN/MPLS junto con las políticas de QoS permiten un servicio de VoIP de alta calidad, garantizando una comunicación fluida, sin retardos y con un porcentaje muy bajo de pérdida de paquetes.

Figura 122

Gráfica de análisis de jitter durante una llamada entre el Cliente3 al Cliente1



CAPÍTULO VI
CONCLUSIONES Y RECOMENDACIONES

6.1. Conclusiones

El análisis previo acerca del protocolo MPLS y la arquitectura SDN, permitió establecer una gran base bibliográfica capaz de sustentar el presente trabajo de titulación y además dejar claro los requisitos de hardware y software necesarios para un desarrollo con éxito.

El diseño de la topología de red basado en los requerimientos fundamentales para sustentar la misma, brindó un ambiente óptimo en el simulador de tiempo real GNS3 para construir el banco de pruebas de la red híbrida SDN/MPLS.

Basado en Xie et al. (2019) se valida que el controlador ODL es el cerebro de la red, encargado de administrar los paquetes generados en el plano de datos a través de los Open vSwitch, permitiendo así que la red sea flexible.

Tal y como se comprobó, el protocolo OpenFlow es la pieza fundamental para garantizar la comunicación entre la capa de infraestructura y la capa de control, ratificando lo que Wu et al. (2020) detalla al mencionar que OpenFlow es el encargado de entregar estadísticas precisas al controlador.

La política open source sobre la cual se establecieron los servidores, permitió la optimización y facilitó la puesta a punto, dado que no existió problemas al realizar su configuración e integración en la topología diseñada para trabajar.

Los Open vSwitch empleados para cumplir con la parte SDN, permitieron la interoperabilidad con los routers tradicionales y el controlador ODL, estableciendo de esta manera la red híbrida SDN/MPLS para entregar un servicio óptimo, confiable y seguro para los clientes de las diferentes redes LAN.

Tras el análisis, se puede determinar que utilizar una pila de etiquetas MPLS y establecer tunelling entre interfaces de red, permite gracias a la alta capacidad de procesamiento de los Open vSwitch un flujo más rápido de paquetes en la nube MPLS.

El establecer las políticas de QoS para diferenciación de servicios, permitió determinar una gestión eficiente del ancho de banda al aplicar un cierto porcentaje de acuerdo con el tráfico

generado por cada servicio, además con el marcaje identificar fácilmente a qué servicio pertenece un flujo conforme al marcaje DSCP configurado.

Tal y como se ha comprobado, en el planteamiento de resultados se puede evidenciar que hay una mejora significativa en la optimización y tiempos de respuesta cortos al momento de realizar una solicitud para acceder a los servicios establecidos en la DMZ, también algo a remarcar, es que el servicio de VoIP brinda una comunicación fluida entre extensiones con un jitter óptimo y casi nula pérdida de paquetes.

Gracias a todo lo anterior, se puede interpretar que la implementación de una red híbrida SDN/MPLS para la gestión de servicios diferenciados es un éxito, demostrando en base a los resultados planteado que es una gran alternativa para mejorar las redes tradicionales, teniendo una estructura flexible, escalable, con alta disponibilidad y fácil al momento de detectar fallas que se llegasen a generar.

6.2. Recomendaciones

Establecer previamente una buena base bibliográfica ayuda a guiar de mejor manera el desarrollo del trabajo, esto incluye definiciones concisas de los conceptos básicos y comandos para realizar las respectivas configuraciones.

El escenario propuesto e implementado para la red híbrida SDN/MPLS, permite sentar una base para futuros proyectos, no solo en un ambiente de simulación, sino en un ambiente real para tener un mejor conocimiento de los equipos implicados y que son necesarios para garantizar la interoperabilidad de servicios.

Al momento de establecer la configuración MPLS en los Open vSwitch, es importante que se encuentren enlazados con el controlador ODL, dado que este es el que se encarga de realizar el manejo de los flujos con las etiquetas MPLS configuradas.

El informe de resultados planteado para el servicio de VoIP, sería óptimo realizarlo en un ambiente real a través de un analizador de espectros, para de esta manera poder ver en realidad los niveles jitter y delta para determinar si el servicio es óptimo.

Debido a las grandes ventajas que proporciona el simulador GNS3 y el sniffer Wireshark, brinda un escenario ideal para la continua investigación sobre simulación de redes, gestión de tráfico inteligente, seguridad y manejo eficiente del ancho de banda.

Las pruebas realizadas se podrían realizar en otro tipo de servicios como correo, DNS, servicios en la nube, hasta en juegos en línea inclusive.

Al momento de establecer políticas de seguridad, hay que recordar que el encolamiento CBWFQ y el marcaje DSCP no garantizan al 100% una QoS óptima, es por esto que también se puede indagar y hacer pruebas con otro tipo de marcaje y encolamiento.

BIBLIOGRAFÍA.

About Us » Issabel.org. (2016). https://www.issabel.org/about-us/

Alarcon Aquino, V. (2008). *Introduccion a Redes MPLS*. El Cid Editor. https://elibro.net/es/lc/utnorte/titulos/34951

Andersson, E., & Brohne, J. (2017). *High Quality of Service in SDN*.

Badotra, S., & Singh, J. (2017). OpenDaylight as a controller for Software Defined Networking. *International Journal of Advanced Research in Computer Science*, *8*(5), 7. http://hdl.handle.net/2117/77683

Bahnasse, A., Louhab, F. E., Ait Oulahyane, H., Talea, M., & Bakali, A. (2018). Novel SDN architecture for smart MPLS Traffic Engineering-DiffServ Aware management. *Future Generation Computer Systems*, *87*, 115–126. https://doi.org/10.1016/j.future.2018.04.066

Bai, J. (2019). *An Introduction to SIP Protocol: Definition, Features, & More*. https://www.nextiva.com/blog/sip-protocol.html

Bermúdez, Cristian; Rodríguez, Jesús; Dussan, C. (2020). Beneficios de las Redes Definidas por Software y el Protocolo Openflow. *2020*, 1–15. https://repository.usc.edu.co/bitstream/handle/20.500.12421/4653/BENEFICIOS DE LAS REDES.pdf?sequence=3&isAllowed=y

Beryllium, R., & Project, O. (2017). *OpenDaylight Documentation Documentation Release Beryllium OpenDaylight Project*.

Campista, M. E. M., Laufer, R., Velloso, P. B., & Jamalipour, A. (2016). Software networks. *Annals of Telecommunications*, *71*(11–12), 569–572. https://doi.org/10.1007/s12243-016-0543-6

Cisco. (2008). *Bandwidth Calculation (Cisco VoIP Implementations)*. http://what-when-how.com/ccnp-ont-exam-certification-guide/bandwidth-calculation-cisco-voip-implementations/

CISCO. (2009). *Preguntas frecuentes sobre Calidad de servicio (QoS) - Cisco*. CISCO. https://www.cisco.com/c/es_mx/support/docs/quality-of-service-qos/qos-

policing/22833-qos-faq.html

CiscoDevNet. (2014). *CiscoDevNet/OpenDaylight-Openflow-App: OpenDaylight (ODL) is an open-source application development and delivery platform. OpenFlow Manager (OFM) is an application developed to run on top of ODL to visual...* https://developer.cisco.com/codeexchange/github/repo/CiscoDevNet/OpenDaylight-Openflow-App/

Classification, W. F. Q. (2016). *Weighted Fair Queuing (Congestion Management and Queuing)*. 1–5. http://what-when-how.com/ccnp-ont-exam-certification-guide/class-based-weighted-fair-queuing-congestion-management-and-queuing/

DanielLOP. (2021). *▷ Requisitos de Hardware Asterisk, Elastix o Issabel*. https://daniellop.com/requisitos-de-hardware-para-issabel/

David Bermúdez Escobar, C., Alexander Rodríguez Taborda, J., & Sc Ciro Antonio Dussan Clavijo, M. (s/f). *Beneficios de las Redes Definidas por Software y el Protocolo Openflow Benefits of Software Defined Networks and Openflow Protocol*.

ESGEEKS. (s/f). *Streama: Crea tu propio "Netflix" personal en Linux » EsGeeks*. Recuperado el 28 de abril de 2022, de https://esgeeks.com/streama-crear-propio-netflix/

Experimen, C. (2017). *A n SDN-B ased A rchitecture for N ext -G eneration W ireless N etworks*. *Febrero*.

GNOME, U. (2016). *XenialXerus/ReleaseNotes/UbuntuGNOME/spanish - Ubuntu Wiki*. https://wiki.ubuntu.com/XenialXerus/ReleaseNotes/UbuntuGNOME/spanish

GNS3 Staff. (2020). *Getting Started with GNS3 | GNS3 Documentation*. docs.gns3.com. https://docs.gns3.com/docs/

Ivan Pepelnjak. (2013). *Management, Control and Data Planes in Network Devices and Systems « ipSpace.net blog*. ipspace.net. https://blog.ipspace.net/2013/08/management-control-and-data-planes-in.html

Leitón, O. M. V. (2015). *Reglas OpenFlow: Mejorando el Algoritmo de Detección de Interacciones*.

Lester, C., Santos, D., Antonio, Y., Muro, M., & Cruz-Enriquez, H. (2017). *SDN Network vs.*

Traditional Network Centros de Datos View project Software-Defined Network (SDN) View project. https://www.researchgate.net/publication/319097382

Martín, I. J., & Cely, C. (2015). *Capa de TRANSPORTE*.

Mendez. (2020). *Comparación OSPF vs IS-IS - Comunidad Huawei Enterprise*. Huawei. https://forum.huawei.com/enterprise/es/comparación-ospf-vs-is-is/thread/629449-100235

Microsoft. (2022). *Protocolos, puertos y servicios de mensajería unificada: ayuda de Exchange 2013 | Microsoft Docs*. https://docs.microsoft.com/es-es/exchange/um-protocols-ports-and-services-exchange-2013-help

Omiotek, Z. (2012). A method for QoS differentiation in DiffServ networks based on the long-term properties of a video stream. *Annales UMCS, Informatica*, *12*. https://doi.org/10.2478/v10065-012-0007-1

ONF. (2013). OpenFlow Switch Specification 1.4.0. *Current*, *0*, 1–3205.

ONF. (2015). *CONFORMANCE TEST SPECIFICATION FOR OPENFLOW SWITCH SPECIFICATION V1.3.4 BASIC SINGLE TABLE CONFORMANCE TEST PROFILE*. 1–399. https://www.opennetworking.org

OPEN NETWORKING FOUNDATION. (2012). OpenFlow Switch Specification. *Current*, 1–36. https://www.opennetworking.org

Open Networking Foundation (ONF). (2018). Software-Defined Networking (SDN) Definition - Open Networking Foundation. En *Open Networking Foundation* (p. 1). https://opennetworking.org/sdn-definition/

OpenDaylight Foundation. (2018). *What is SDN? - OpenDaylight*. OpenDaylight. https://www.opendaylight.org/about/what-is-sdn

Paessler. (2016). *Ancho de banda: definición y detalles*. https://www.paessler.com/es/it-explained/bandwidth

Pryslupskyi, A., Panchenko, O., Beshley, M., & Seliuchenko, M. (2019). Improvement of multiprotocol label switching network performance using software-defined controller. *2019 15th International Conference on the Experience of Designing and Application of CAD Systems, CADSM 2019 - Proceedings*, 106–109.

https://doi.org/10.1109/CADSM.2019.8779316

Quality of Service QoS - My IT notes. (s/f). Recuperado el 29 de diciembre de 2021, de https://yunus.fr/quality-of-service-qos/

Quesada, J. A. L. (s/f). *Presentación de la Asignatura | Enhanced Reader*.

rfc3031. (2001). https://datatracker.ietf.org/doc/html/rfc3031#page-3

Rodríguez Ximena. (2019). *Tipos de servidores web OpenWebinars*. https://openwebinars.net/blog/tipos-servidores-web/

Servicio, D. E. (2016). *MAESTRÍA EN REDES DE COMUNICACIÓN TESIS PREVIA A LA OBTENCIÓN DEL TÍTULO DE : MASTER EN REDES DE COMUNICACIÓN TEMA : MIROSLAVA ARACELY ZAPATA RODRÍGUEZ DIRECTOR : MsC . Francisco Chafla . Quito , Marzo 2016 Dedicatoria*.

Sinha, Y., Bhatia, S., Shekhawat, V. S., & Chalapathi, G. S. S. (2017). MPLS based hybridization in SDN. *2017 4th International Conference on Software Defined Systems, SDS 2017*, 156–161. https://doi.org/10.1109/SDS.2017.7939157

Sociedad, U. Y., Wladimir, B., Bayas, O., & Zambrano Vega, C. (2021). *Volumen 13 | Número 2 | Marzo-Abril*. https://orcid.org/0000-0002-5366-5917

Sonny Eli Zaluchu. (2021). *Estudio de Factibilidad para el diseño de una Red MPLS como Backbone de Internet para brindar servicios de Internet fijo y móvil en relación al crecimiento poblacional del Ecuador hasta el año 2021* (Vol. 3, Número 2).

Stallings, W., Agboma, F., & Jelassi, S. T. A.-T. T.-. (2016). Foundations of modern networking : SDN, NFV, QoE, IoT, and Cloud LK - https://glos.on.worldcat.org/oclc/927715441. En *Network* (Vol. 139, Número 3). http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=1601537%0Ahttp://proquest.safaribooksonline.com/9780134175478%0Ahttp://proquest.safaribooksonline.com.mit.idm.oclc.org//9780134175478%0Ahttp://proxy.library.carleton.ca/log

Stephen Richardson. (2021). *The Class Selector PHB and DSCP Values - Traffic Shaping*. https://www.ccexpert.us/traffic-shaping-2/the-class-selector-phb-and-dscp-

values.html

SubNet. (2016). *Puerto 8020 (tcp/udp) - Descubridor de puertos TCP UDP online - adminsub.net*. Búsqueda de puerto TCP/UDP. https://es.adminsub.net/tcp-udp-port-finder/8080

Tajiki, M. M., Akbari, B., Mokari, N., & Chiaraviglio, L. (2019). SDN-based resource allocation in MPLS networks: A hybrid approach. *Concurrency Computation* , *31*(8), 1–13. https://doi.org/10.1002/cpe.4728

The Apache Software Foundation. (2011). *About the Apache HTTP Server Project - The Apache HTTP Server Project*. apache.org. https://httpd.apache.org/ABOUT_APACHE.html

Ungerman, J. (s/f). *Josef Ungerman, CCIE #6167*.

Vissicchio, S., Vanbever, L., & Bonaventure, O. (2014). Opportunities and research challenges of hybrid software defined networks. *Computer Communication Review*, *44*(2), 70–75. https://doi.org/10.1145/2602204.2602216

Wendell, O. (2020). *SDN and Controller-Based Networks > Introduction to Controller-Based Networking | Cisco Press*. Cisco Press. https://www.ciscopress.com/articles/article.asp?p=2995354&seqNum=2

Wiley, J. (2020). New-generation Protocols. *New-generation Protocols*, *2*, 115–132. https://doi.org/10.1002/9781119694748.ch9

Wu, Y. J., Hwang, P. C., Hwang, W. S., & Cheng, M. H. (2020). Artificial intelligence enabled routing in software defined networking. *Applied Sciences (Switzerland)*, *10*(18). https://doi.org/10.3390/APP10186564

Xie, J., Richard Yu, F., Huang, T., Xie, R., Liu, J., Wang, C., & Liu, Y. (2019). A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges. *IEEE Communications Surveys and Tutorials*, *21*(1), 393–430. https://doi.org/10.1109/COMST.2018.2866942

Zoiper.com. (2019). *Zoiper - Free VoIP SIP softphone dialer with voice, video and instant messaging :: Zoiper*. https://www.zoiper.com/

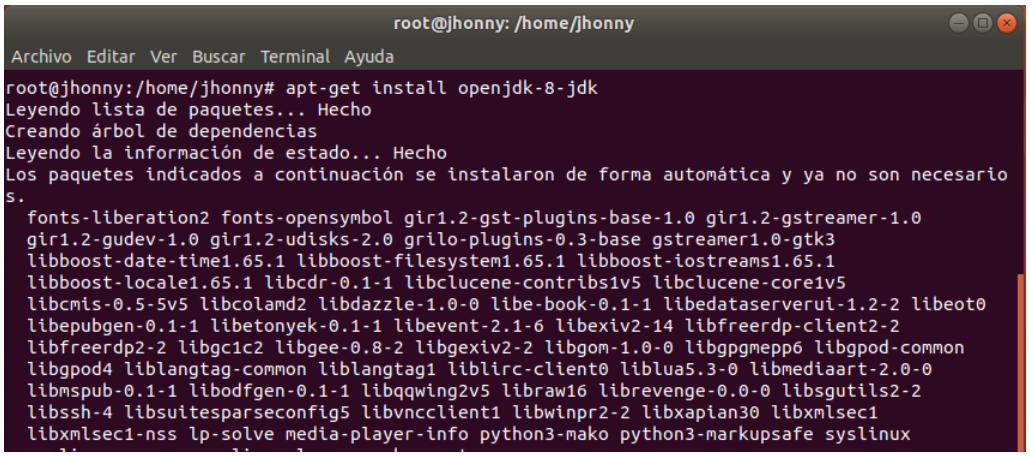
ANEXOS

Anexo 1: Instalación de ODL

OpenDayLight trabaja de la mano de Java, por lo cual, primero se debe realizar la instalación de Java en su versión 8 como se muestra en la Figura 123, dado que es la única versión para operar ODL, esto se realiza en la consola de Linux digitando el siguiente comando: `apt-get install openjdk-8-jdk`

Figura 123

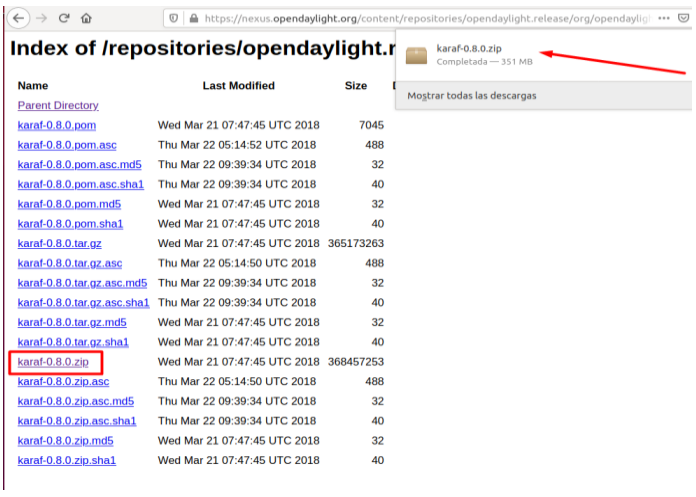
Instalación de Java



ODL se maneja por repositorios de acuerdo con cada versión, en este caso se usa el repositorio:*https://nexus.opendaylight.org/content/repositories/opendaylight.release/org/opendaylight/integration/karaf/0.8.0/*, finalmente se descarga el archivo zip que se hace énfasis en el recuadro rojo de la Figura 124.

Figura 124

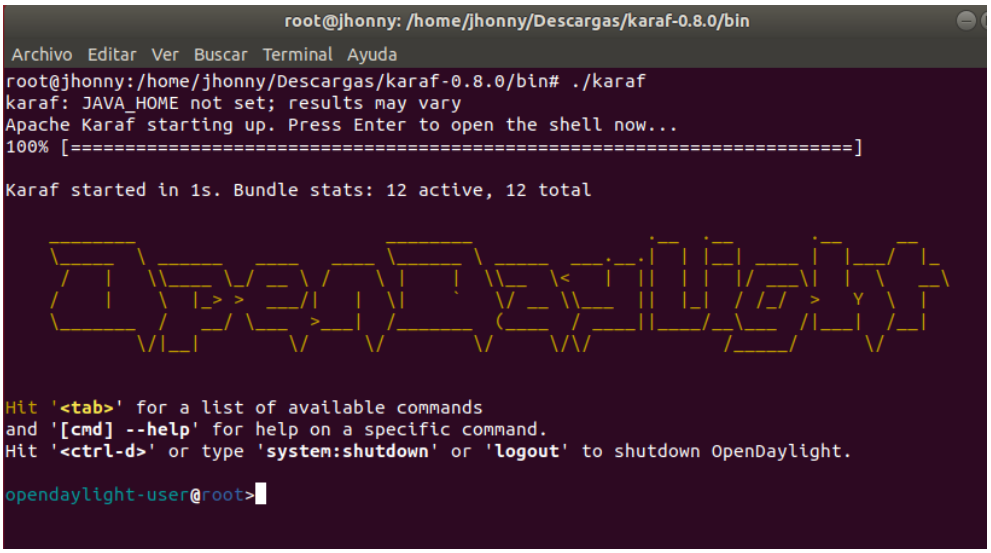
Repositorio para la descarga de OpenDayLight



Una vez la descarga finaliza, lo siguiente es dirigirse a la carpeta donde se encuentra el archivo, el cual se desempaqueta el archivo .zip para tener acceso a todos los archivos y así ejecutar ODL; el comando final sería el siguiente: cd /home/jhonny/Descargas/karaf-0.8.0/bin# ./karaf. Si no hay ningún problema ODL se ejecuta normalmente y como se muestra en la Figura 125 proporciona una carátula con su nombre, con esto ODL ya se encuentra operativo.

Figura 125

ODL operativo



Ahora es momento de instalar los módulos de ODL que fueron descritos en la Tabla 15 y que se muestra en la Figura 126, en la misma consola de ODL se procede con su instalación y el comando que lo permite es el siguiente:

```
feature:install odl-restconf-all odl-openflowplugin-all odl-l2switch-all osl-mdsal-all odl-yangtools-common odl-dlux-all
```

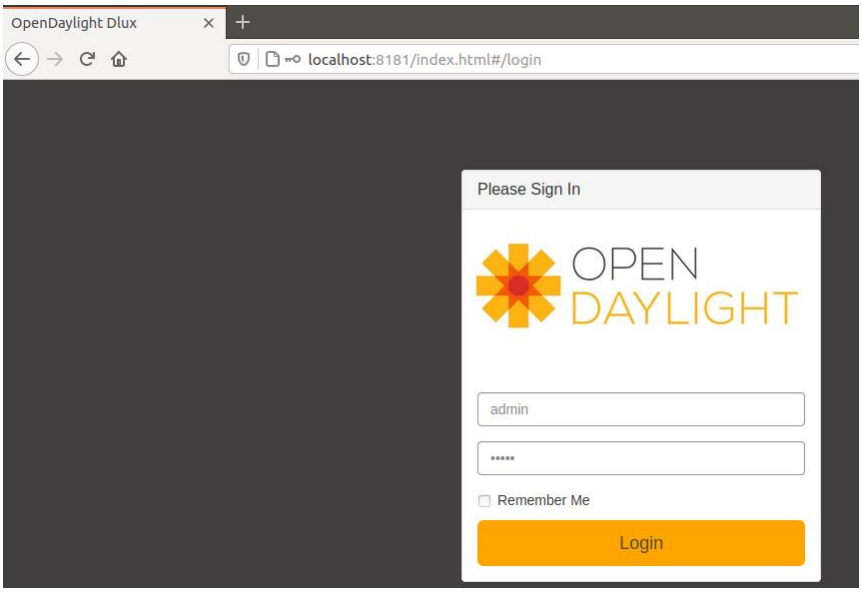
Figura 126

Instalación de módulos de ODL

Con todo el escenario listo, se abre un navegador (Chrome, Mozilla, Opera, etc.) y en la barra de navegación se digita la dirección http://localhost:8181/index.html, de esta manera se accede a la página de inicio de ODL como se muestral en la Figura 127.

Figura 127

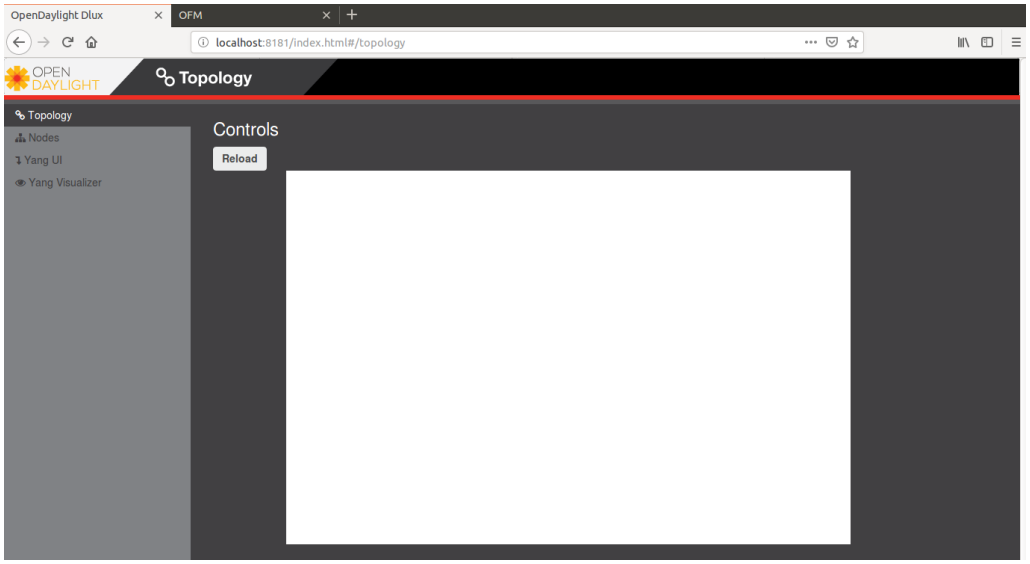
Página de inicio de ODL



Para culminar, se digita las credenciales de acceso que es **admin** tanto para username como contraseña, es normal que cuando recién se ejecuta ODL se muestre un mensaje que no se puede acceder, pero esperando un par de minutos se ingresa sin ningún problema al panel de control como se muestra en la Figura 128.

Figura 128

Panel de control de ODL

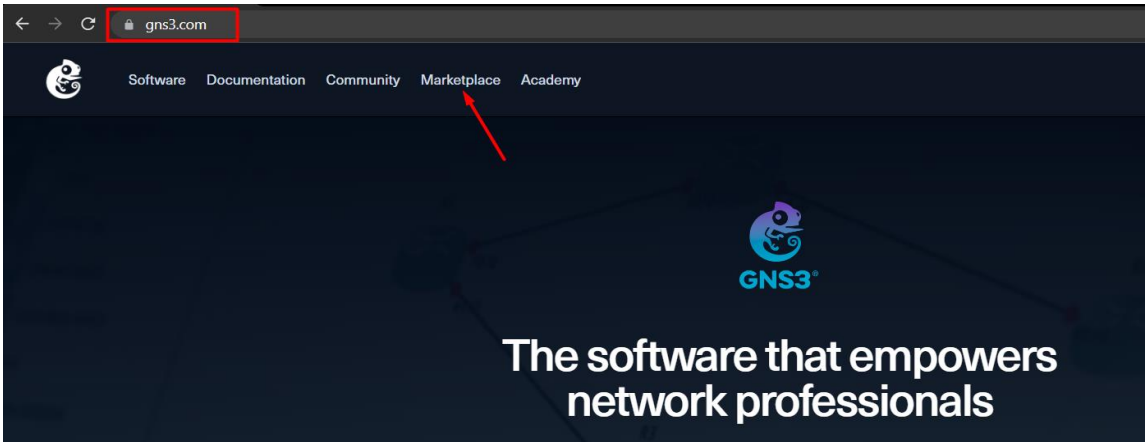


Anexo 2: Instalación de Open vSwitch

Para realizar la instalación de Open vSwitch, lo primero se debe ingresar a la página web de GNS3, esto se hace mediante la dirección https://www.gns3.com/ como se muestra en la Figura 129, luego se debe dirigir a la sección de Marketplace señalada con la flecha de color rojo.

Figura 129

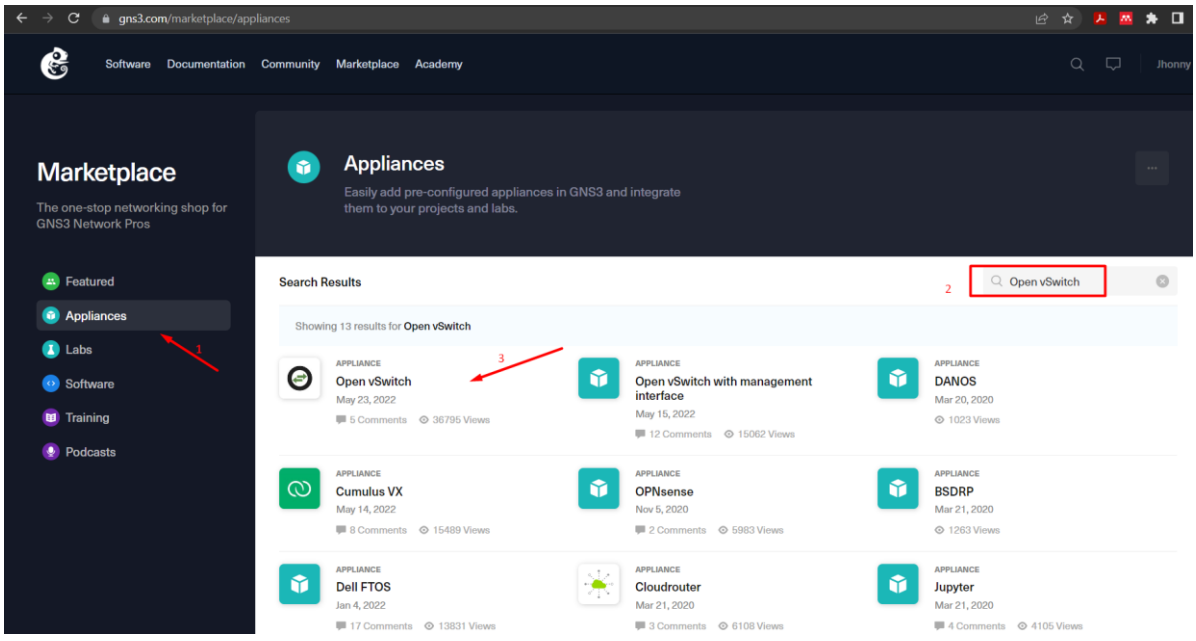
Página web de GNS3



Dentro de la tienda de aplicaciones de GNS3, la Figura 130 muestra el procedimiento a seguir, primero se ingresa en la sección de Appliances, segundo en la barra de búsqueda se escribe Open vSwitch y finalmente se escoge la aplicación a descargar.

Figura 130

Tienda de aplicaciones de GNS3



Con la aplicación descargada se ejecuta el software GNS3, una vez el programa carga en la sección **File** se escoge la opción **Import appliance** tal y como la Figura 131 lo muestra. Al ingresar se debe escoger la aplicación descargada que se muestra en la Figura 132 para continuar con el proceso de instalación.

Figura 131

Importación de aplicación en GNS3

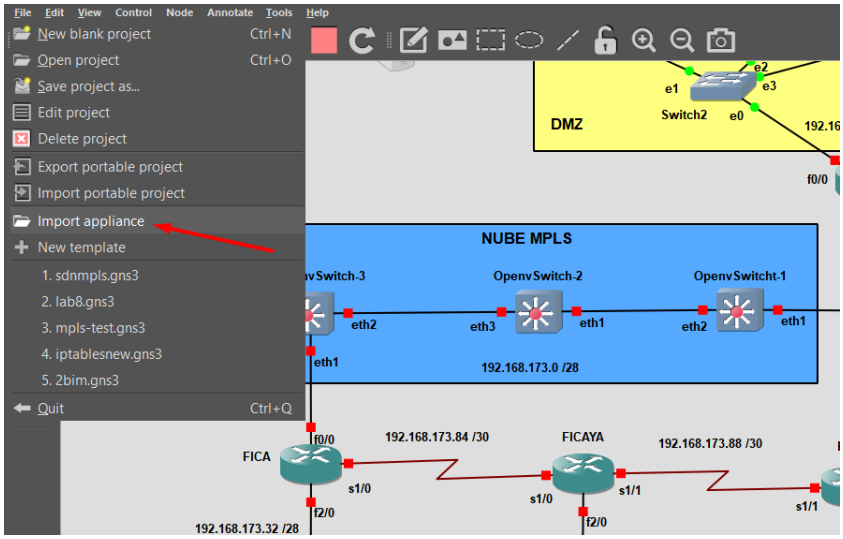
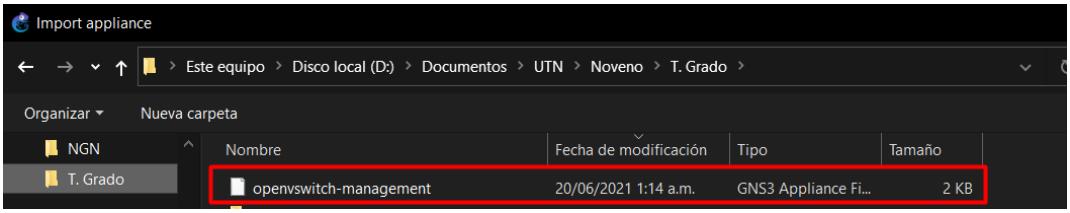


Figura 132

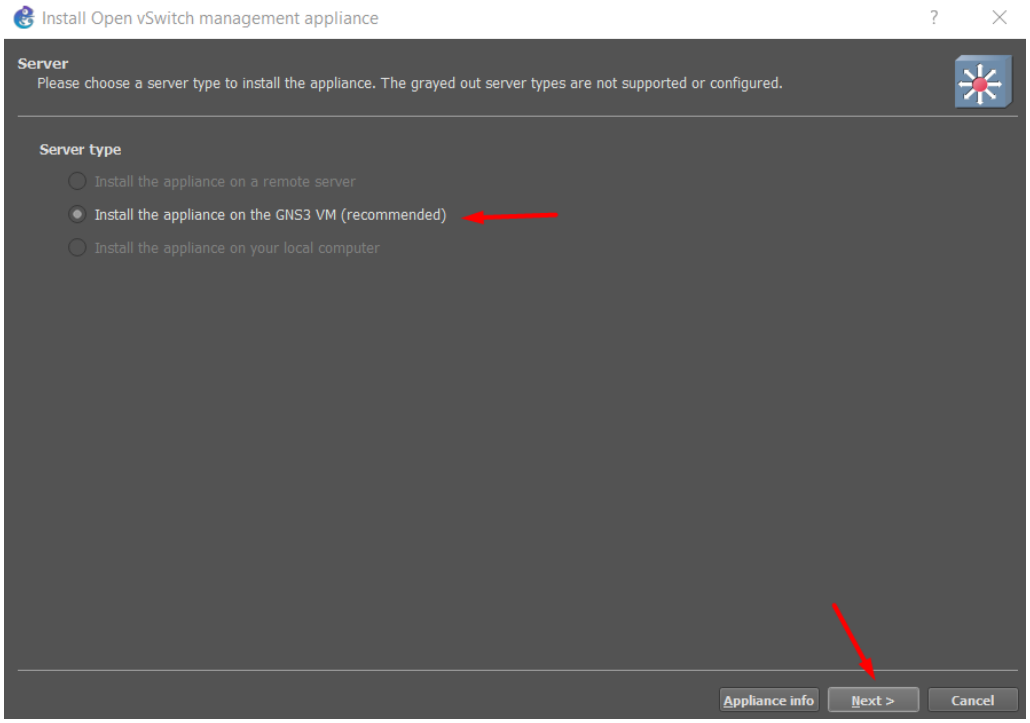
Selección de la aplicación Open vSwitch descargada



Continuando con el proceso, la Figura 133 indica el tipo de servidor que se debe escoger, en este caso es el propio de GNS3 como se recomienda, luego se da en Next para continuar con el proceso.

Figura 133

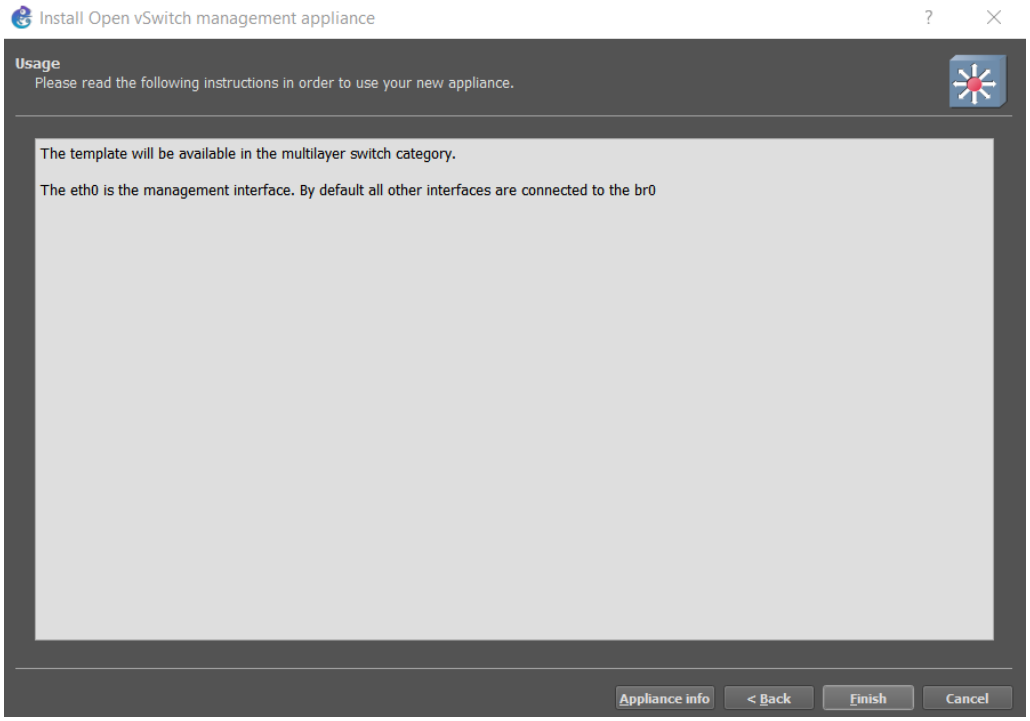
Selección del servidor para operar la aplicación



La Figura 134 muestra que la aplicación ha sido añadida exitosamente, es decir está lista para que se pueda hacer uso de ella.

Figura 134

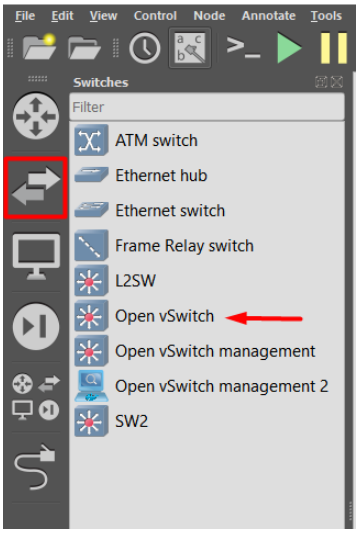
Aplicación añadida con éxito en GNS3



Para hacer uso de la aplicación, en la Figura 135 muestra la sección de switches y en la fecha roja se remarca que la aplicación se ha añadido exitosamente y está lista para ser usada, únicamente se debe arrastrar hacia la pantalla principal y establecer la topología que se desee.

Figura 135

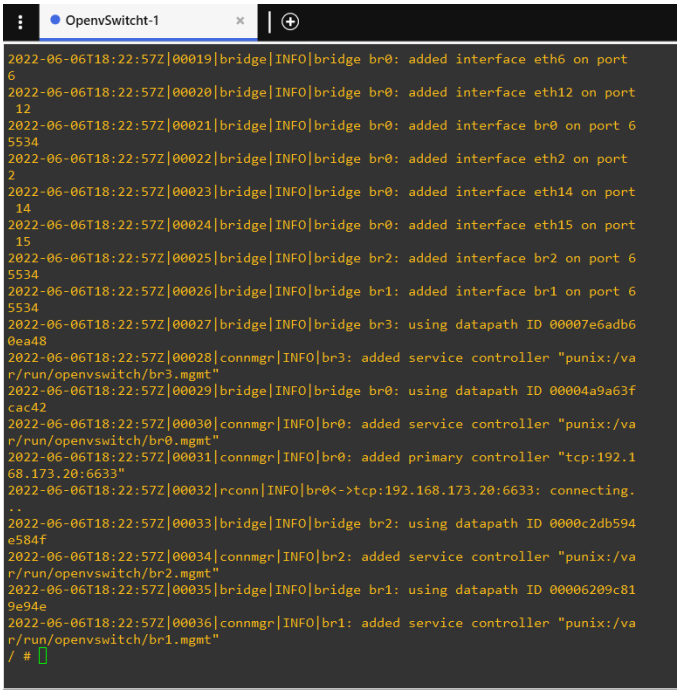
Open vSwitch en la sección de Switches de GNS3



Finalmente se ejecuta Open vSwitch, y como se aprecia en la Figura 136 se tiene la consola lista para realizar las configuraciones necesarias para el funcionamiento de la topología de red diseñada.

Figura 136

Consola principal de Open vSwitch



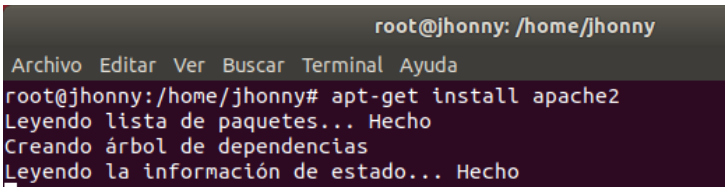
Anexo 3: Instalación servidor WEB y FTP

Anexo 3.1: Servidor WEB

Para instalar la plataforma de Apache, se abre la consola y se accede como usuario root, luego por medio del comando **`apt-get install apache2`** se procede con su instalación, así como muestra la Figura 137.

Figura 137

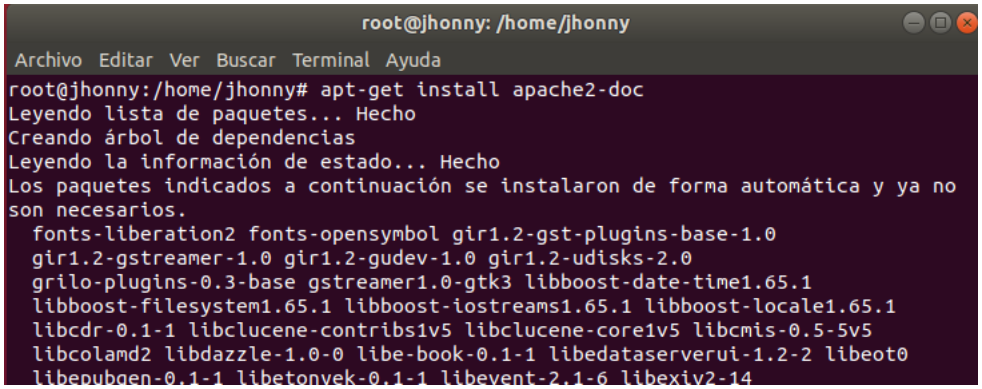
Instalación Servidor Web Apache



Una opción adicional es instalar la documentación de Apache, la cual contiene notas de las versiones, guías de usuario, manual de referencia, etc. o también se puede acceder desde la dirección https://httpd.apache.org/docs/2.4/es/. En la Figura 138 se muestra el proceso de instalación a través del comando: **`apt-get install apache2-doc`**

Figura 138

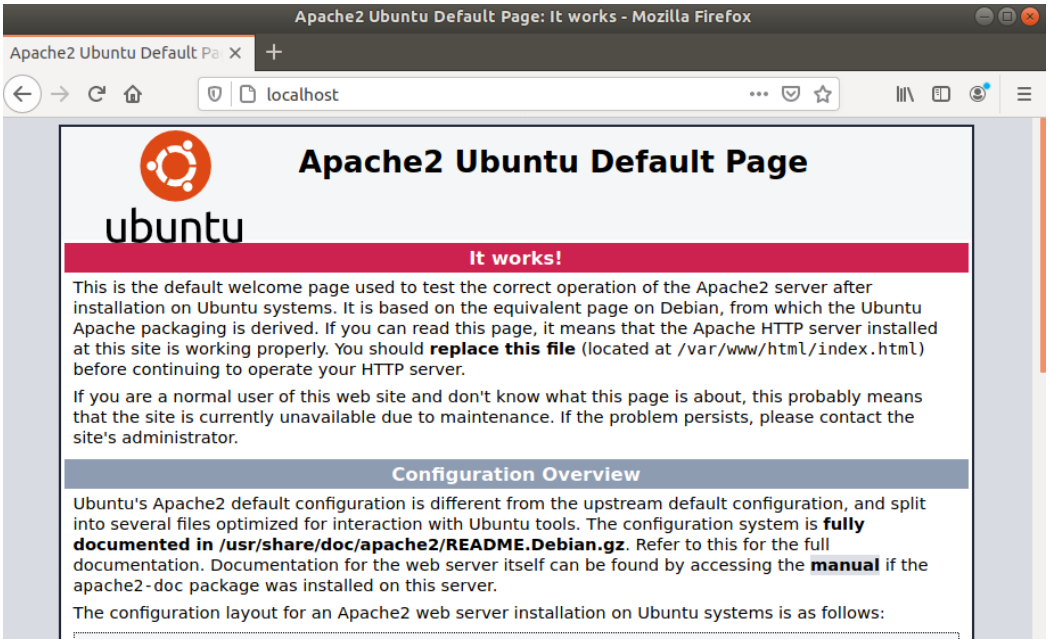
Instalación de la Documentación de Apache



Para determinar que la instalación se realizó correctamente, entrando en un navegador se escribe en la barra de navegación la dirección *127.0.0.*1 o *localhost*, se accede a la página por default de Apache y como se observa en la Figura 139 el servidor ya se encuentra operativo.

Figura 139

Página dafault de Apache



Si se desea acceder a la documentación instalada se escribe en la barra de navegación *127.0.0.1/manual* o *localhost/manual* y como se observa en la Figura 140, la documentación también se encuentra disponible.

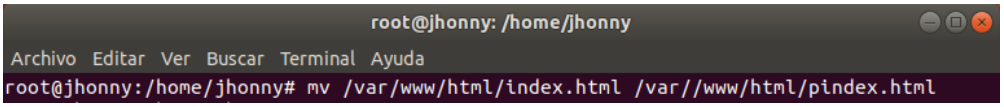
Figura 140

Documentación de Apache

Ahora es momento de configurar la página web que se desea diseñar, como primer paso se cambia el nombre del archivo **index.html** original para tenerlo de respaldo, este proceso se realiza mediante el comando ***mv /var/www/html/index.htm /var/www/html/pindex.html,*** esto se muestra en la Figura 141.

Figura 141

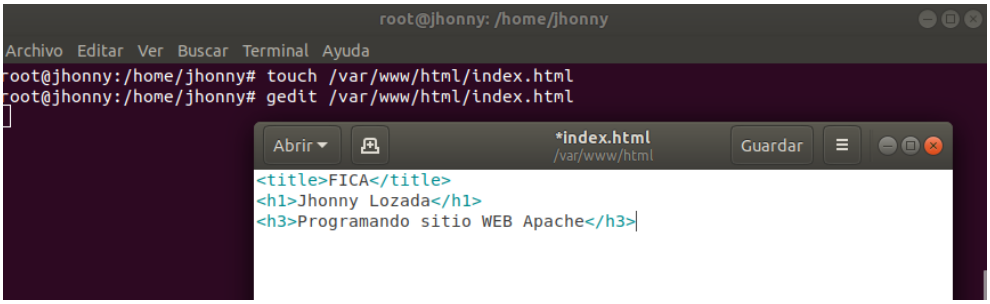
Cambio de nombre del archivo original Index de Apache para respaldo



Se prosigue con el proceso y ahora se crea un nuevo archivo **index.html,** la Figura 142 proporciona el proceso a seguir con los siguientes comandos: ***touch /var/www/html/index.htm*** para crear el archivo y ***gedit /var/www/html/index.htm*** para editar el archivo y configurar el sitio Web a través de lenguaje de programación HTML5.

Figura 142

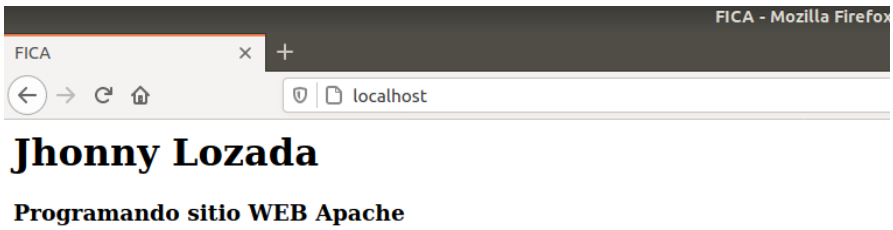
Creación y edición del archivo index.html



Para comprobar que la configuración ha sido aplicada correctamente y el sitio web funciona correctamente, es suficiente con recargar el sitio web y verificar que aparezca lo establecido mediante HTML5, esto se muestra en la Figura 143.

Figura 143

Página Web editada

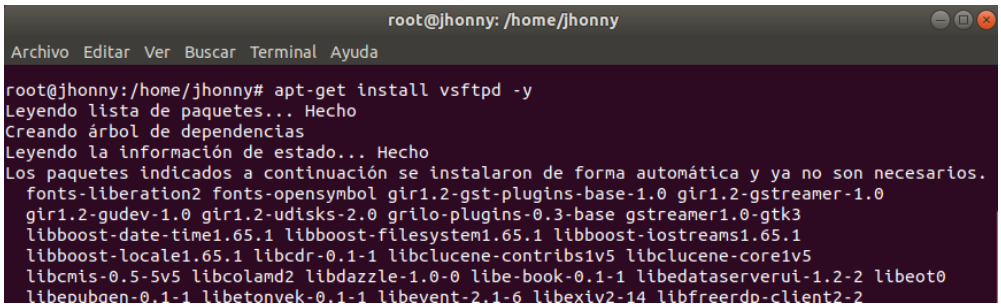


Anexo 3.2: Servidor FTP

Para la instalación del servidor FTP se lo realiza mediante la herramienta **Vsftpd,** la cual es una de las más potentes y completas que existe en distribuciones de Linux, en la Figura 144 se muestra su instalación a través del comando: `apt-get install vsftpd -y`

Figura 144

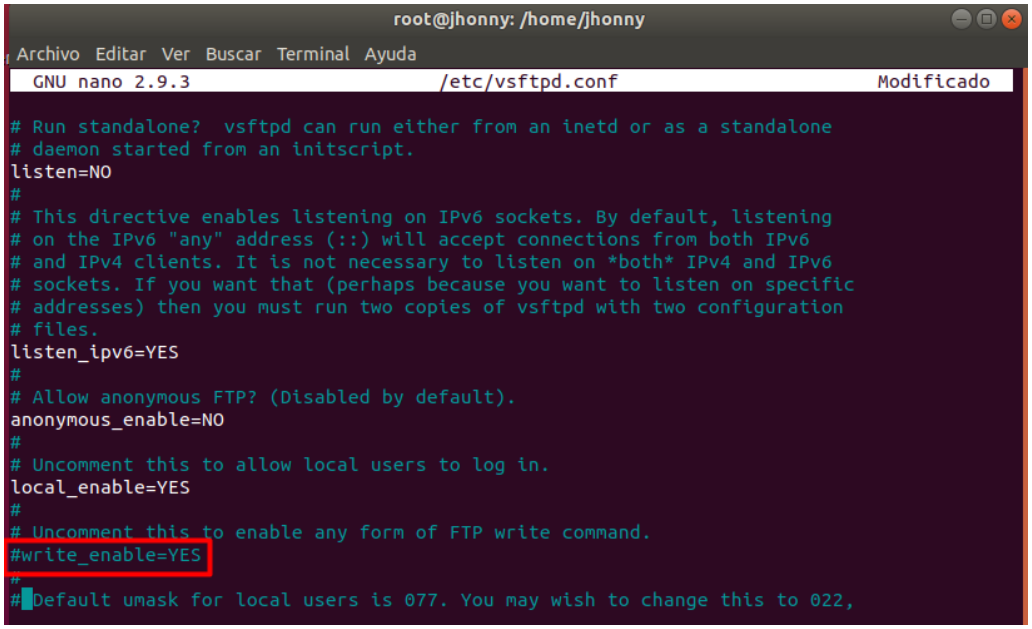
Instalación de Vsftpd para el servidor FTP



Finalizada la instalación se procede a editar el archivo de la configuración Vsftpd, se lo realiza a través del comando **nano /etc/vsftpd.conf,** tal y como se muestra en la Figura 145, se desmarca la línea **write_enable=YES** que se remarca en el recuadro rojo guardando los cambios al momento de salir de la edición.

Figura 145

Escritura habilitada en Vsftpd



Con los cambios realizados, se reinicia el servicio con el comando **service vsftpd restart** para iniciar con los cambios implementados, y luego a través del comando **service**

vsftpd status se verifica que el servicio esté corriendo normalmente como se muestra en la Figura 146.

Figura 146

Vsftpd corriendo sin errores

Para hacer uso del servicio FTP, se descarga el programa FileZilla a través de la dirección https://filezilla-project.org, luego hay que seleccionar la opción Cliente que se señala con una flecha roja en la Figura 147, hay que remarcar que es una de las herramientas más completas para servicio FTP.

Figura 147

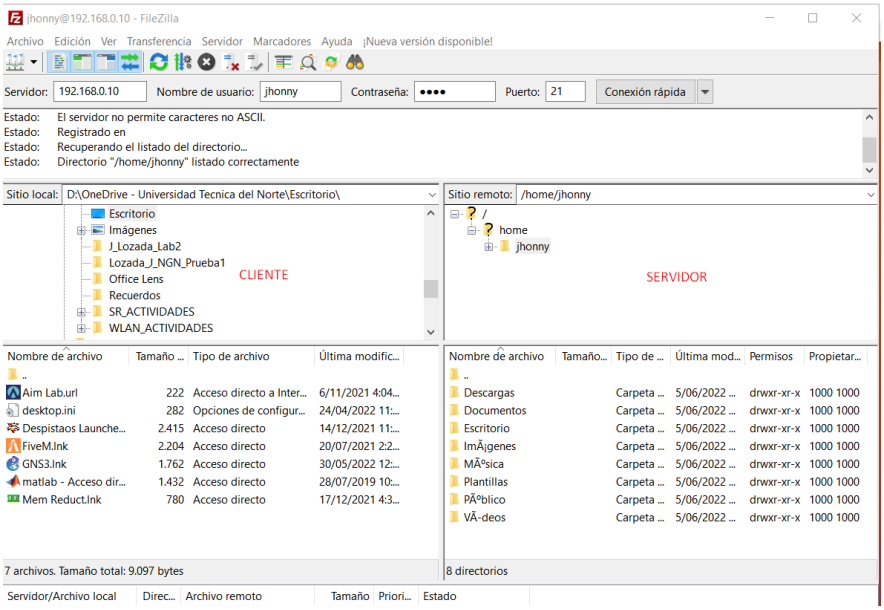
Sitio oficial de FileZilla

Para terminar, una vez el programa es instalado para conectarse al servicio es necesario ingresar las credenciales del servidor, estas son: dirección IP, nombre de usuario, contraseña y

el puerto 21 de FTP así como se muestra en la Figura 148, con esto el servicio FTP estaría listo y operativo.

Figura 148

Servicio FTP operativo estableciendo conexión Cliente-Servidor



Anexo 4: Instalación servidor Streaming

Para realizar la instalación de Streama se necesita de Java 8 o posterior, la Figura 149 muestra la instalación a través del comando `apt-get install openjdk-8-jdk`

Figura 149

Instalación de Java para Streama

```
root@ubuntu:/home/jhonny# apt install openjdk-8-jdk
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  ca-certificates-java fonts-dejavu-extra java-common libgif7 libice-dev
  libpthread-stubs0-dev libsm-dev libx11-dev libx11-doc libxau-dev libxcb1-dev
  libxdmcp-dev libxt-dev openjdk-8-jdk-headless openjdk-8-jre
  openjdk-8-jre-headless x11proto-core-dev x11proto-input-dev x11proto-kb-dev
  xorg-sgml-doctools xtrans-dev
Paquetes sugeridos:
  default-jre libice-doc libsm-doc libxcb-doc libxt-doc openjdk-8-demo
  openjdk-8-source visualvm icedtea-8-plugin fonts-ipafont-gothic
  fonts-ipafont-mincho fonts-wqy-microhei fonts-wqy-zenhei fonts-indic
Se instalarán los siguientes paquetes NUEVOS:
  ca-certificates-java fonts-dejavu-extra java-common libgif7 libice-dev
  libpthread-stubs0-dev libsm-dev libx11-dev libx11-doc libxau-dev libxcb1-dev
  libxdmcp-dev libxt-dev openjdk-8-jdk openjdk-8-jdk-headless openjdk-8-jre
  openjdk-8-jre-headless x11proto-core-dev x11proto-input-dev x11proto-kb-dev
  xorg-sgml-doctools xtrans-dev
```

Lo siguiente es crear la carpeta que contendrá los archivos que serán necesarios más adelante, como se observa en la Figura 150 con el comando **mkdir** se crea un directorio, en este caso se crean dos con los siguientes comandos: **mkdir /data** y **mkdir /data/Streama**

Figura 150

Creación de directorios

```
root@ubuntu:/home/jhonny# mkdir /data
root@ubuntu:/home/jhonny# mkdir /data/streama
```

Continuando el proceso con el comando **cd /data/streama** se ingresa al directorio que se acabó de crear, luego se descarga Strema desde el repositorio de Github con la sentencia **wget https://github.com/streamaserver/streama/releases/download/v1.6.0-RC7/streama-1.6.0-RC7.war** mostrado en la Figura 151.

Figura 151

Descarga de Streama desde el repositorio de Github

```
root@ubuntu:/data/streama# wget https://github.com/streamaserver/streama/releases/download/v1.6.0-RC7/streama-1.6.0-RC7.war
--2022-05-08 20:00:19--  https://github.com/streamaserver/streama/releases/download/v1.6.0-RC7/streama-1.6.0-RC7.war
Resolviendo github.com (github.com)... 140.82.112.4
Conectando con github.com (github.com)[140.82.112.4]:443... conectado.
Petición HTTP enviada, esperando respuesta... 302 Found
Ubicación: https://objects.githubusercontent.com/github-production-release-asset-2e65be/39890823/d9e19c5a-64bd-11e8-9e89-9b
%2F20220509%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20220509T010020Z&X-Amz-Expires=300&X-Amz-Signature=b8b38412080b822c1
=0&key_id=0&repo_id=39890823&response-content-disposition=attachment%3B%20filename%3Dstreama-1.6.0-RC7.war&response-content
--2022-05-08 20:00:20--  https://objects.githubusercontent.com/github-production-release-asset-2e65be/39890823/d9e19c5a-64b
NJYAX4CSVEH53A%2F20220509%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20220509T010020Z&X-Amz-Expires=300&X-Amz-Signature=b8b
=host&actor_id=0&key_id=0&repo_id=39890823&response-content-disposition=attachment%3B%20filename%3Dstreama-1.6.0-RC7.war&re
Resolviendo objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.
Conectando con objects.githubusercontent.com (objects.githubusercontent.com)[185.199.108.133]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 94360806 (90M) [application/octet-stream]
Grabando a: “streama-1.6.0-RC7.war”

streama-1.6.0-RC7.war                          19%[====================>
```

Con el archivo descargado se debe convertir en ejecutable, esto se hace mediante el comando **chmod +x streama-1.6.0-RC7.war**, finalmente se ejecuta la aplicación con el comando **./streama-1.6.0-RC7.war**; todo este proceso lo proporciona la Figura 152.

Figura 152

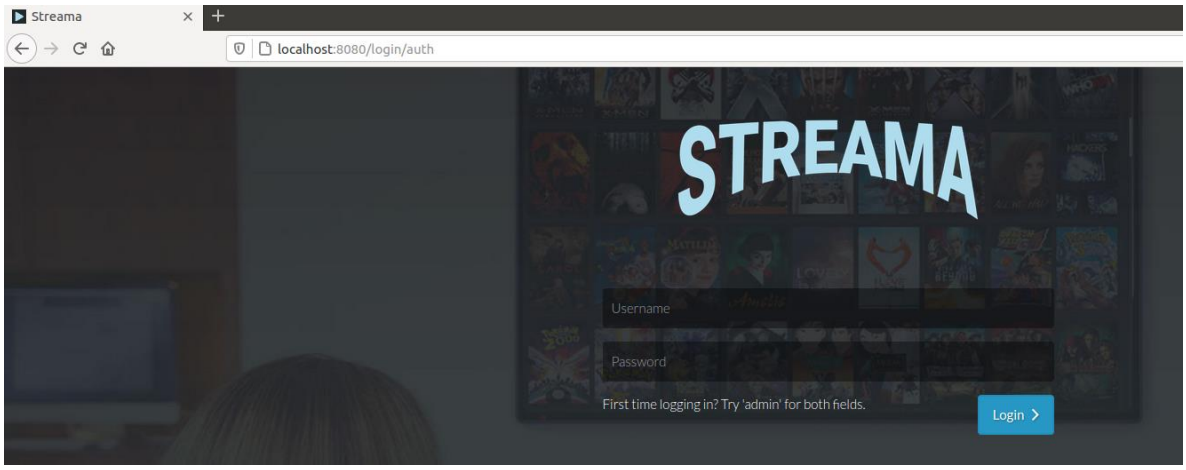
Ejecución de la aplicación Streama

```
root@ubuntu:/data/streama# chmod +x streama-1.6.0-RC7.war
root@ubuntu:/data/streama# ./streama-1.6.0-RC7.war
```

Una vez la aplicación se encuentra en ejecución, en la barra de navegación se escribe la URL http://dirección-IP:8080, en la Figura 153 se muestra la pantalla de inicio de Streama, y para ingresar se usa las credenciales admin (username y contraseña).

Figura 153

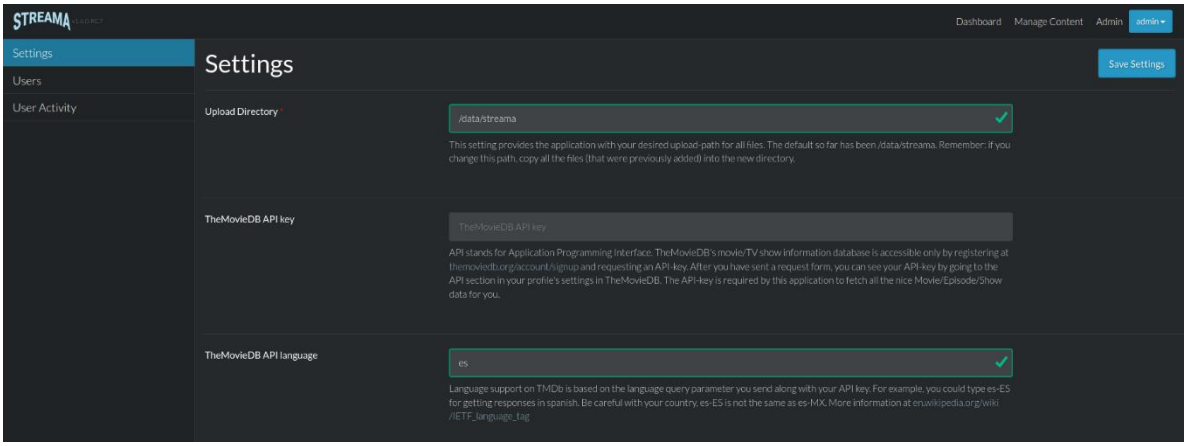
Página de inicio de Streama



Al ingresar directamente se pasa directamente al apartado de configuración, aquí simplemente se coloca la dirección del directorio en donde se encuentra la aplicación y el idioma de la API como se muestra en la Figura 154.

Figura 154

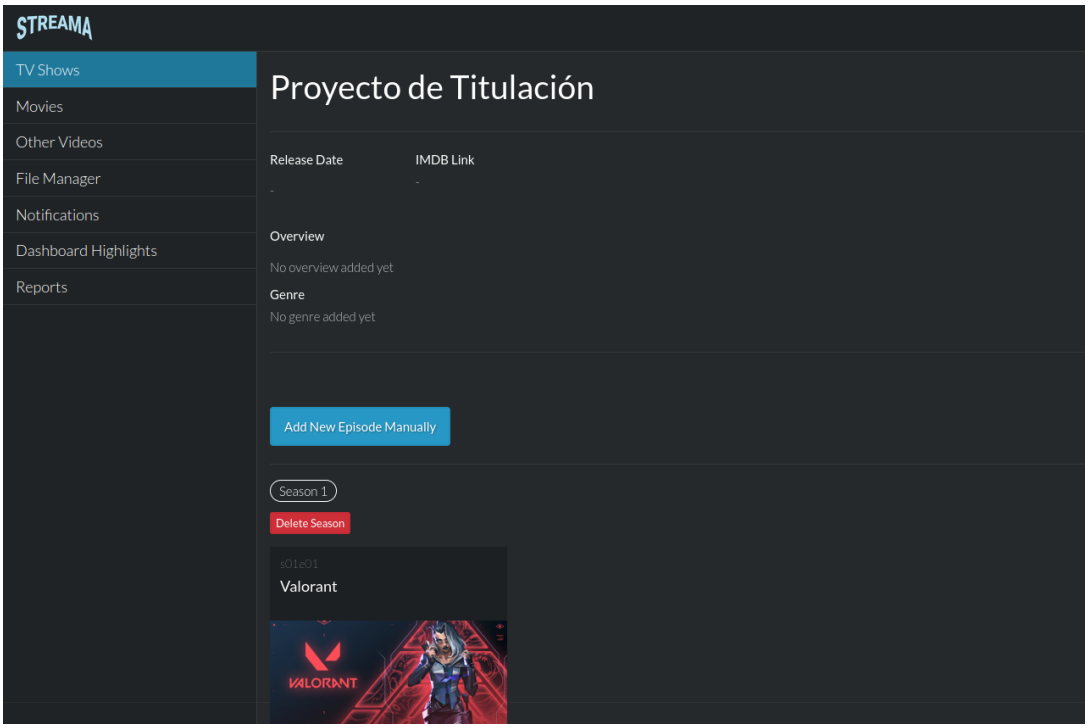
Configuración de Streama



Para terminar, con las configuraciones establecidas ya es posible añadir contenido en la plataforma para que los usuarios puedan disfrutarlo, así como se muestra en la Figura 155.

Figura 155

Carga de contenido en Streama

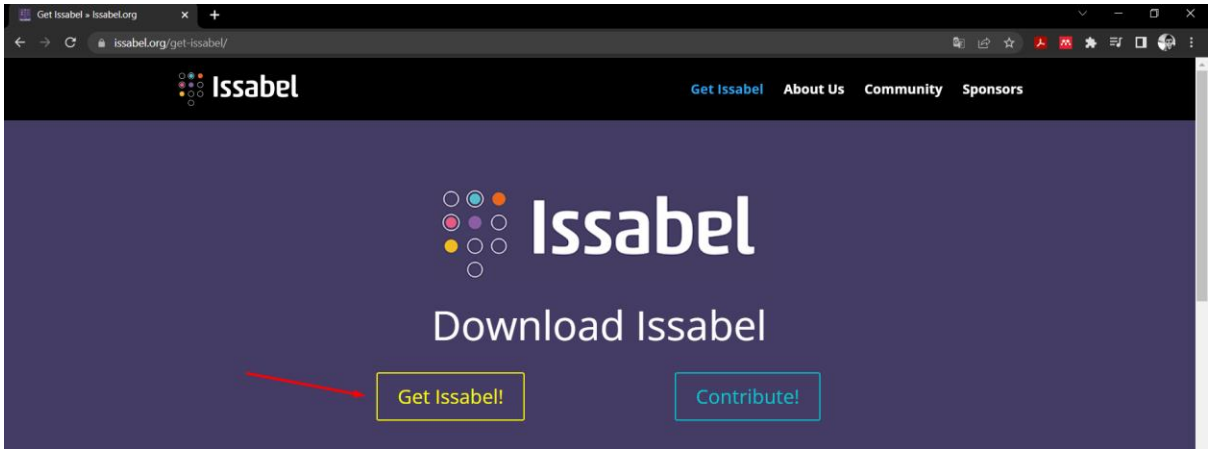


Anexo 5: Instalación servidor VoIP

Lo primero a realizar es ingresar a la URL https://www.issabel.org/get-issabel/, en la Figura 156 se muestra el sitio oficial de Issabel, lo siguiente es seleccionar la opción **Get Issabel** señalada con la flecha roja, en donde inmediatamente empezará la descarga del archivo ISO.

Figura 156

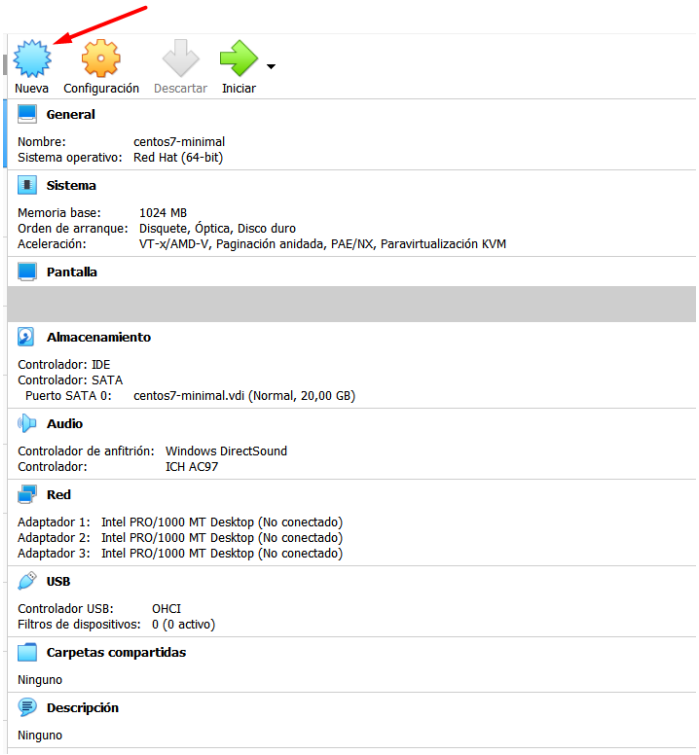
Sitio Oficial de Issabel



VMware y VirtualBox son de las plataformas más utilizadas para crear máquinas virtuales (VM), en este caso se usa VirtualBox como se observa en la Figura 157. El primer paso es dar a la opción **nueva** señalada con la flecha roja para crear la VM.

Figura 157

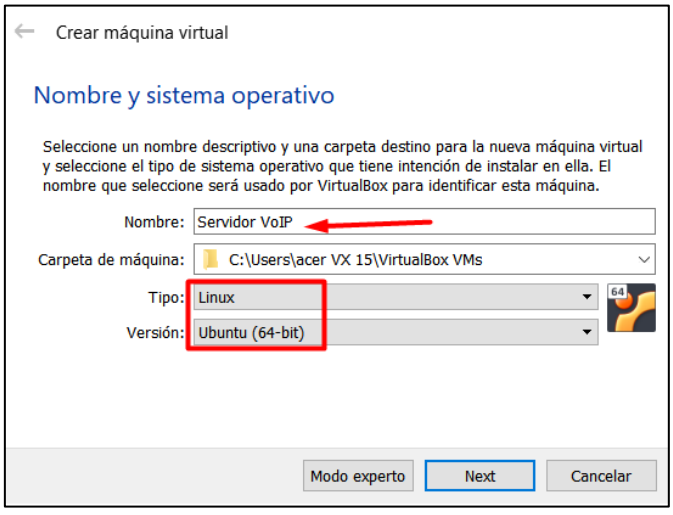
Creación de una VM en VirtualBox



Para crear la VM se le asigna un nombre, la ruta donde se va a alojar, el tipo de Sistema Operativo y por último la versión, así como se observa en la Figura 158.

Figura 158

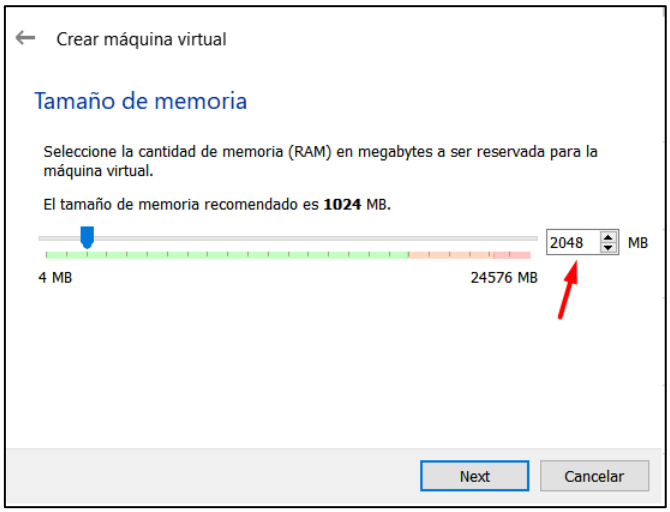
Nombre y Sistema Operativo para la VM



Se designa la cantidad de memoria RAM para la VM como muestra la Figura 159, se debe tener en cuenta la capacidad de RAM de la máquina física para que no llegue a afectar su funcionamiento.

Figura 159

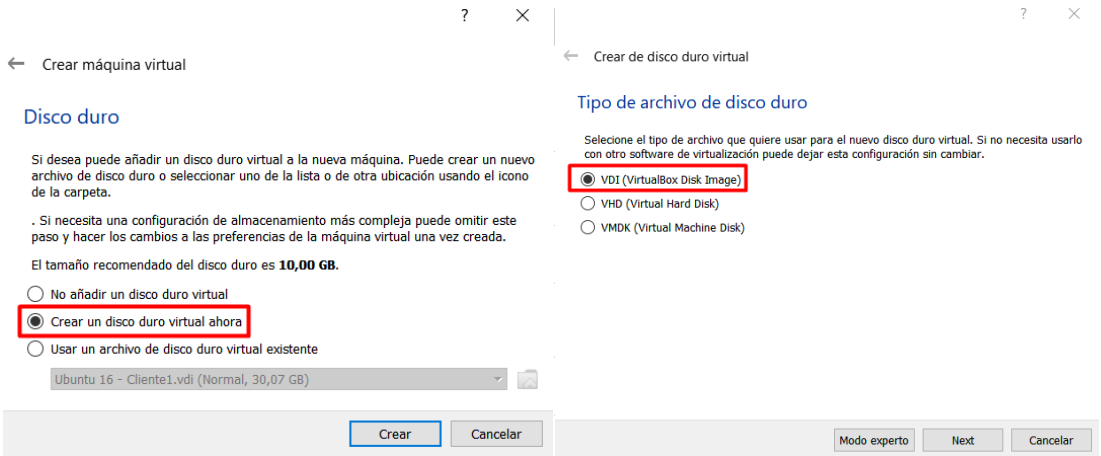
Designación de memoria RAM para la VM



Siguiendo con el proceso, primero se selecciona crear un disco duro y luego se escoge el tipo VDI que es el recomendado, así como lo resaltan los recuadros de color rojo en la Figura 160.

Figura 160

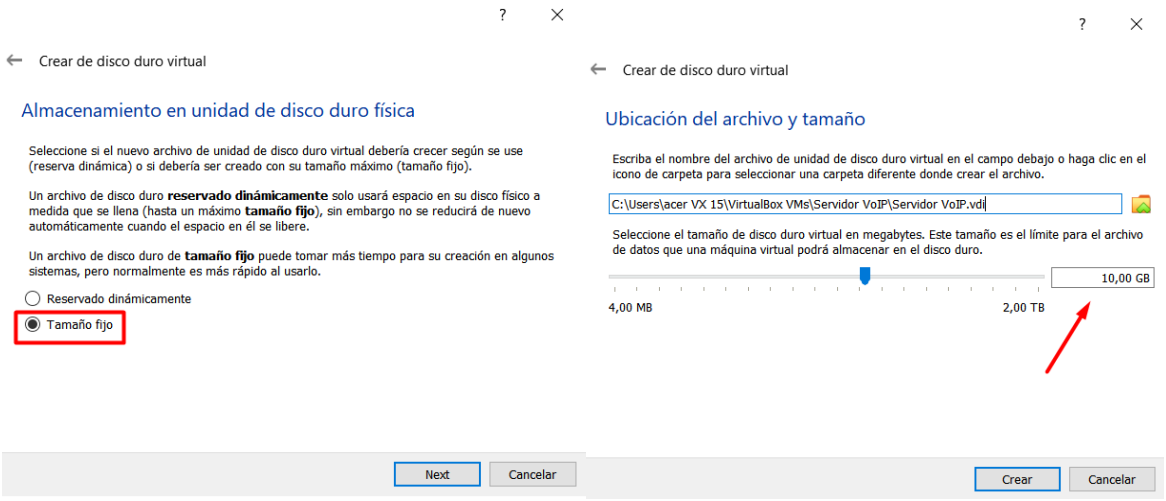
Creación del disco duro de la VM



Para poder determinar el tamaño del disco duro de la VM se debe seleccionar tamaño fijo y luego se establece la ubicación de este, así como la Figura 161 muestra. Hay que aclarar que depende de la capacidad de la máquina física el tamaño que se pueda designar a la VM.

Figura 161

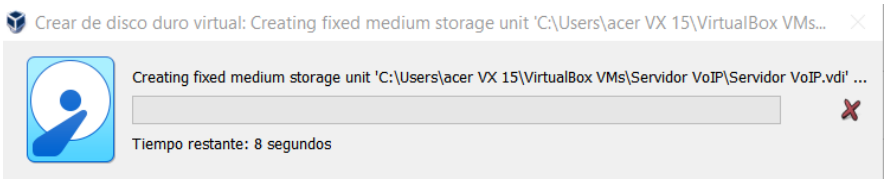
Ubicación y tamaño del disco duro para la VM



Finalmente, la VM se crea como la Figura 162 lo muestra, es normal que dependiendo de las características de la máquina física tarde más o menos tiempo en establecerse.

Figura 162

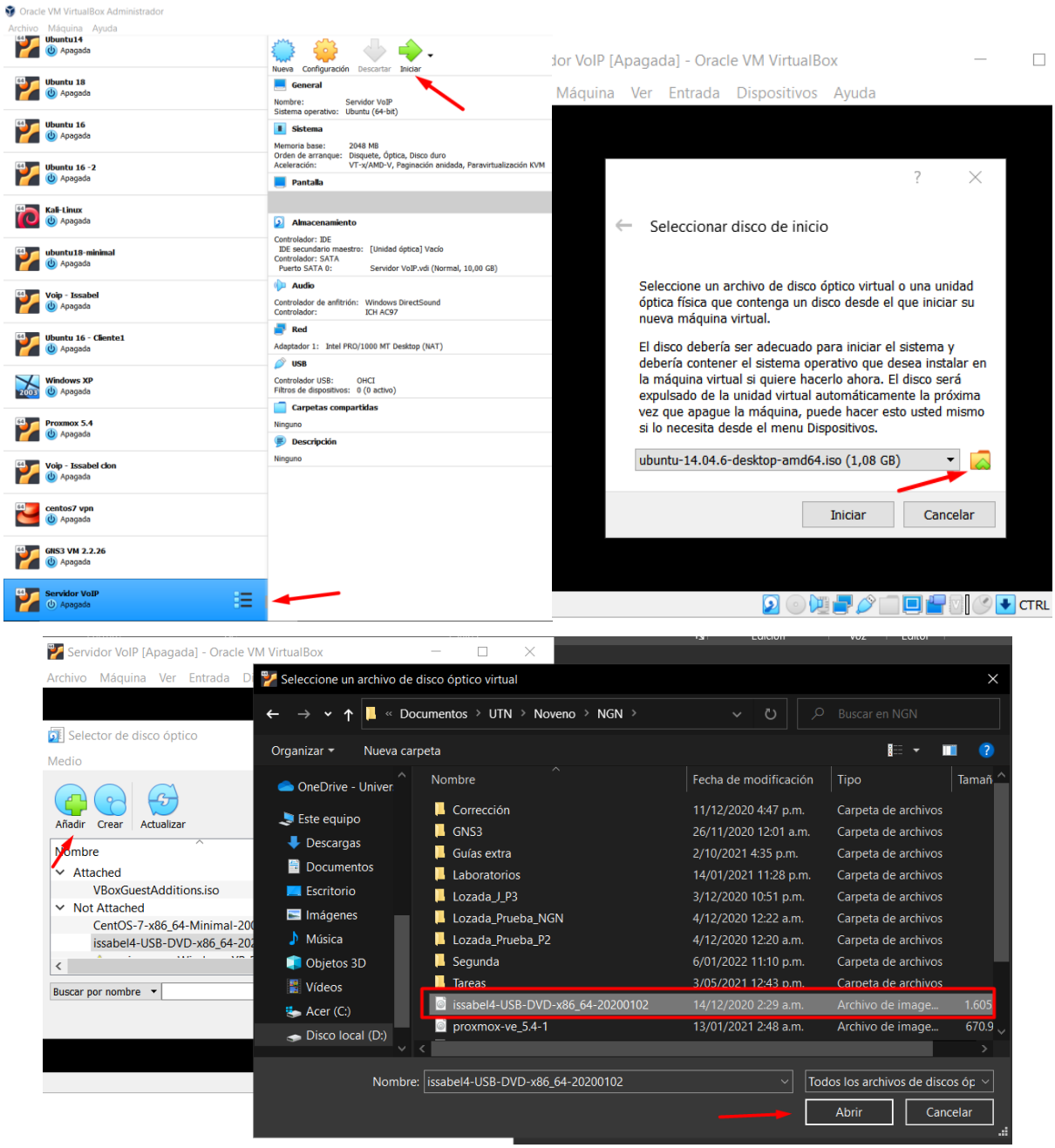
Establecimiento de la VM creada



Seleccionando la VM creada, se arranca y lo primero que solicitará es el disco de inicio, se da click en el símbolo de carpeta y se desplegará explorador de archivos de la máquina física, luego se escoge el ISO de Issabel que fue descargado al principio, todo este proceso lo proporciona la Figura 163.

Figura 163

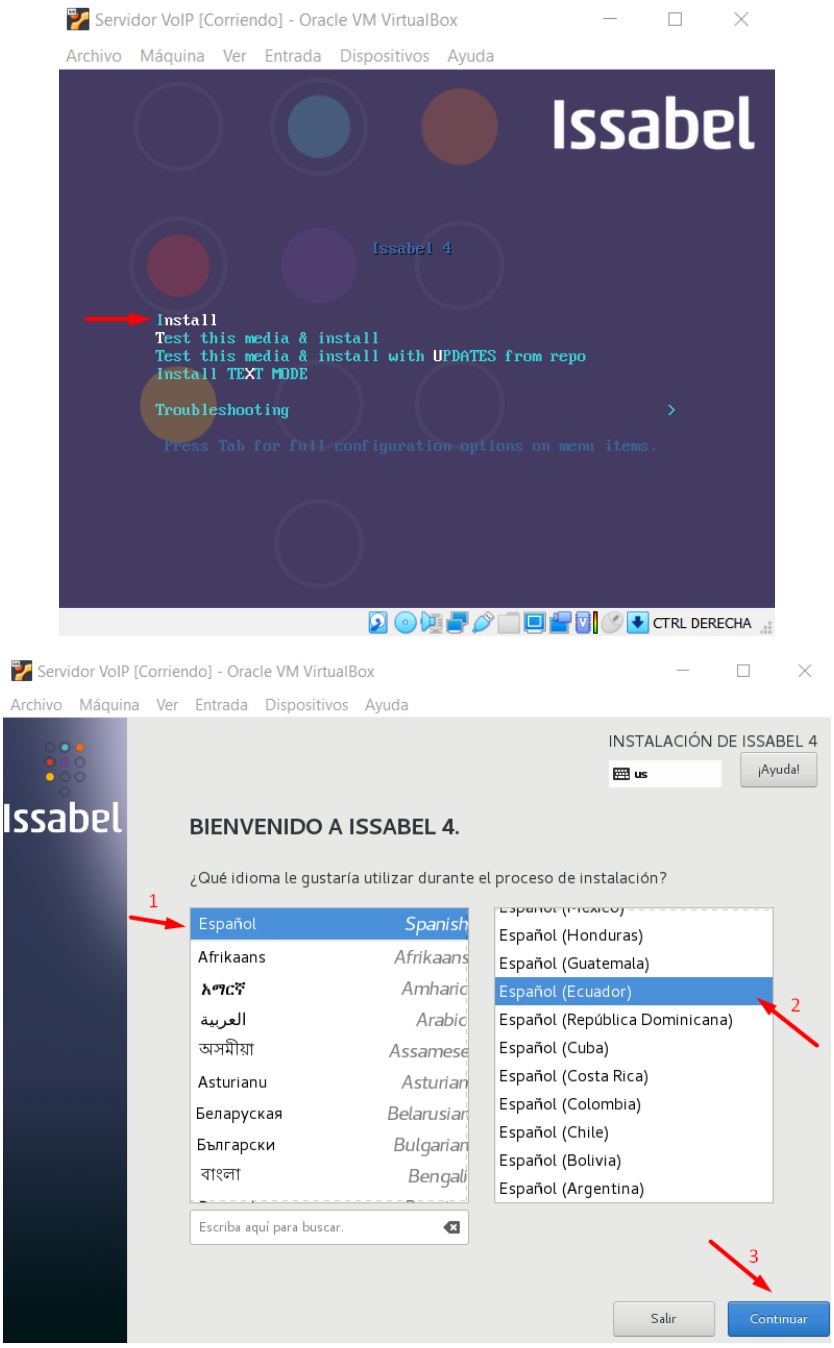
Selección del disco inicio con el ISO de Issabel



Al iniciar con el disco de arranque, se selecciona instalación para comenzar el proceso, y lo primero que se selecciona es el idioma para el proceso de instalación, en la Figura 164 muestra el proceso que se sigue.

Figura 164

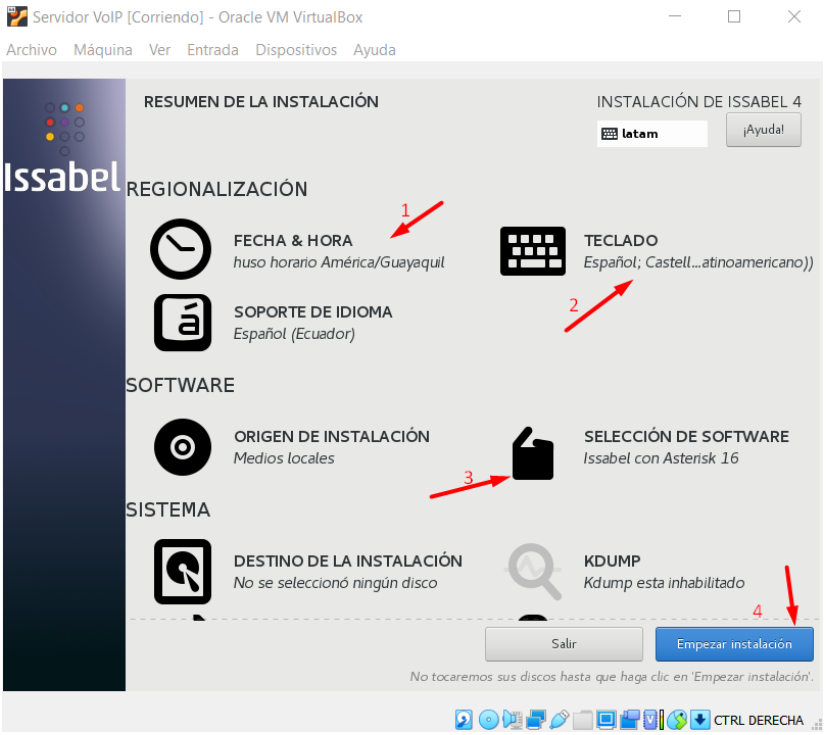
Instalación de Issabel



Ahora viene una parte esencial en el proceso de instalación, la Figura 165 proporciona el procedimiento a seguir; primero se selecciona la zona horaria, segundo el idioma del teclado, tercero en software se debe seleccionar Asterisk 16 y finalmente se empieza con la instalación.

Figura 165

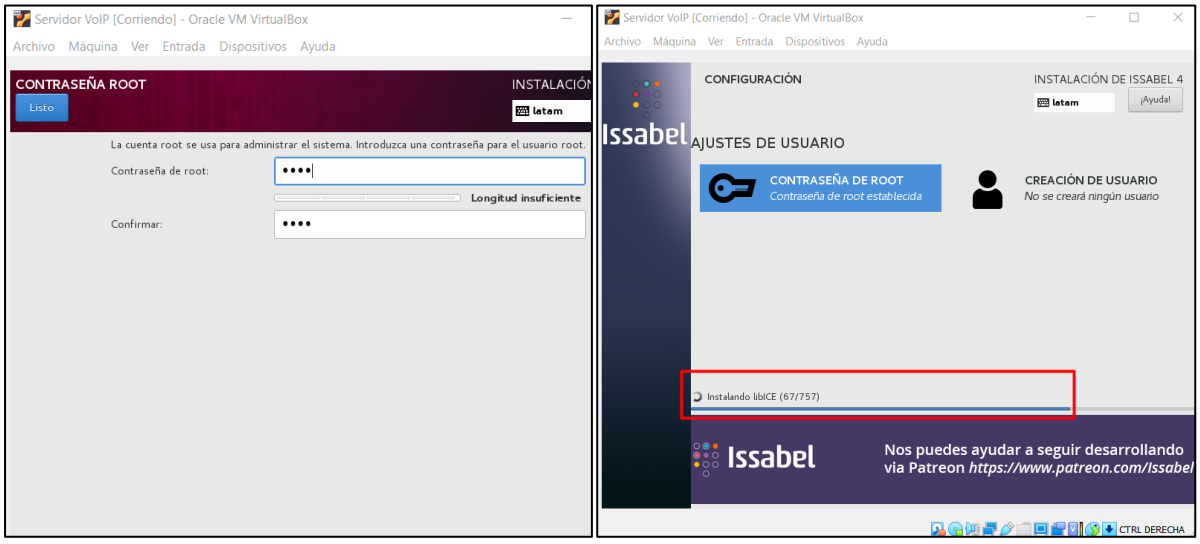
Establecimiento de parámetros de instalación



Para terminar, se establece la contraseña de root mientras de ejecuta el proceso de instalación, así como se observa en la Figura 166.

Figura 166

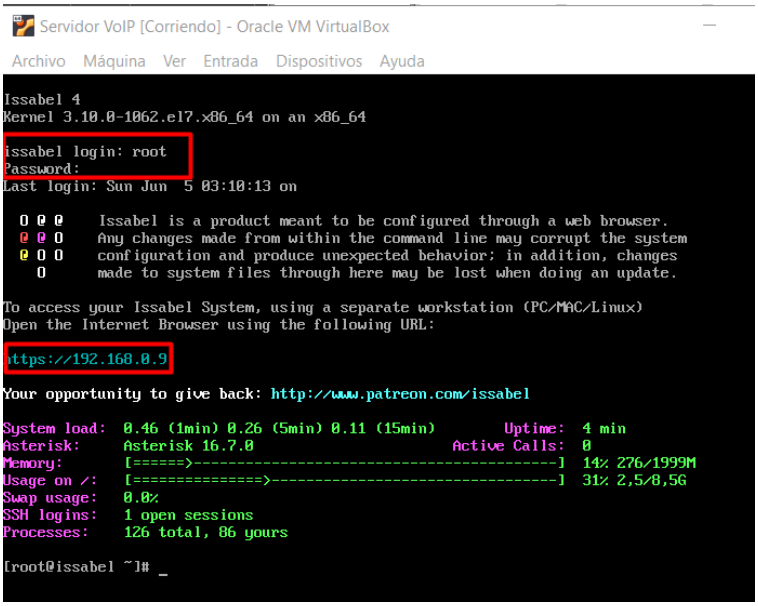
Establecimiento de la contraseña de root



Una vez finalizada la instalación, se procede a Iniciar Issabel mostrando su interfaz en la Figura 167, en donde al iniciar con la contraseña de root aparece la URL con la que se debe ingresar en el navegador.

Figura 167

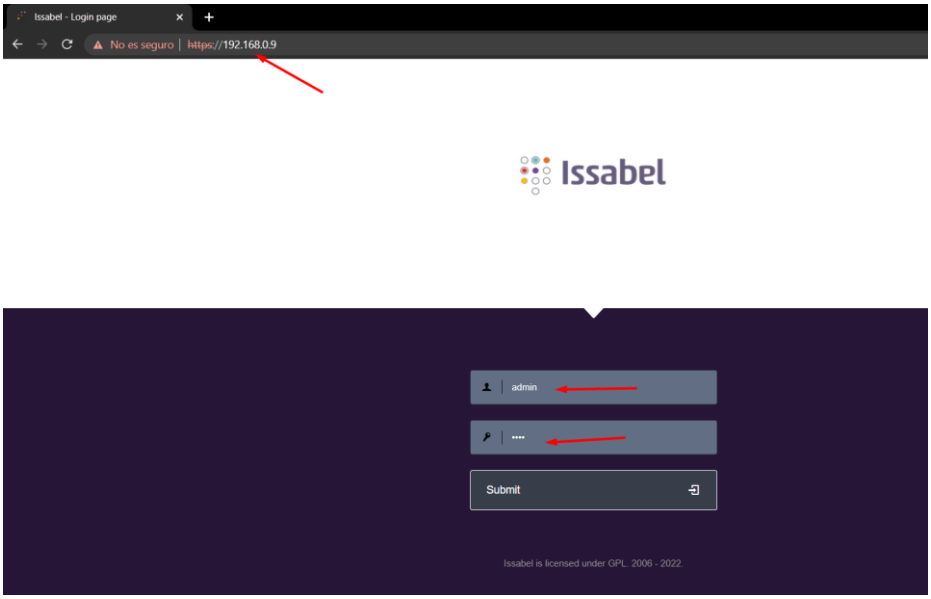
Issabel en ejecución



Se ingresa la URL proporcionada en Issabel en la barra de navegación, la cual nos lleva a la página de inicio de Issabel mostrada en la Figura 168, en usuario de coloca admin y el password es la contraseña de root.

Figura 168

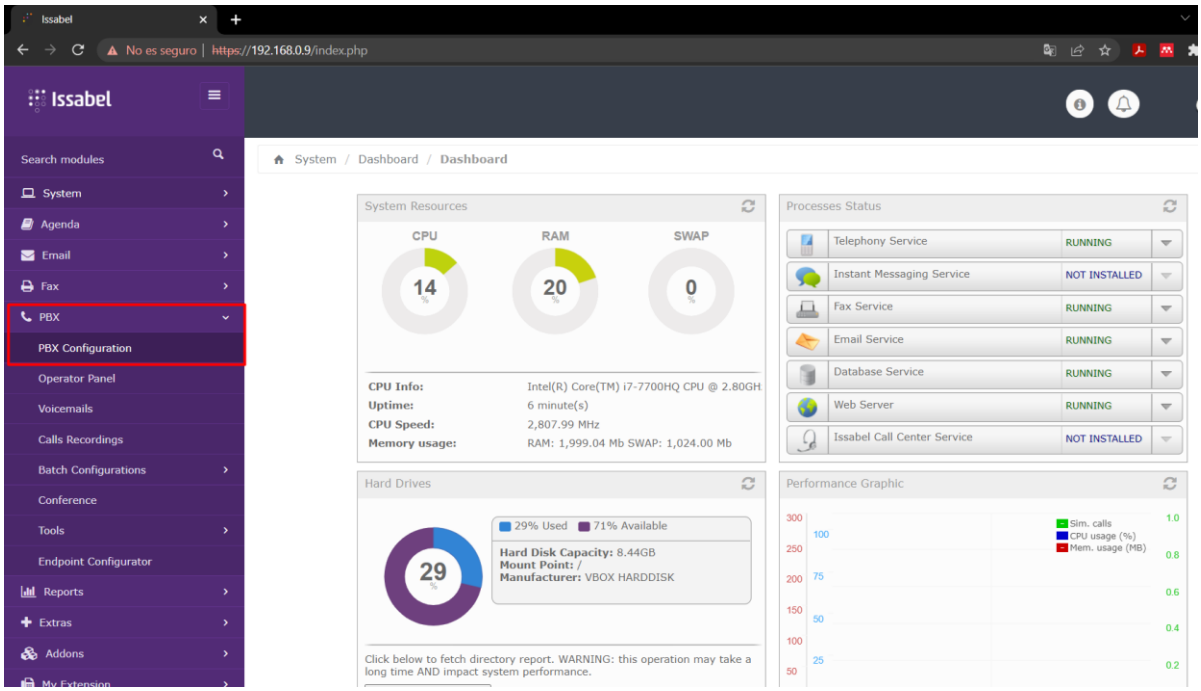
Página de inicio de Issabel



Al logearse, en la Figura 169 se muestra la interfaz para realizar las respectivas configuraciones, además el estado actual de la CPU y RAM del sistema. Para crear las extensiones se selecciona el apartado PBX y se escoge la opción Configuración PBX resaltado en el recuadro rojo.

Figura 169

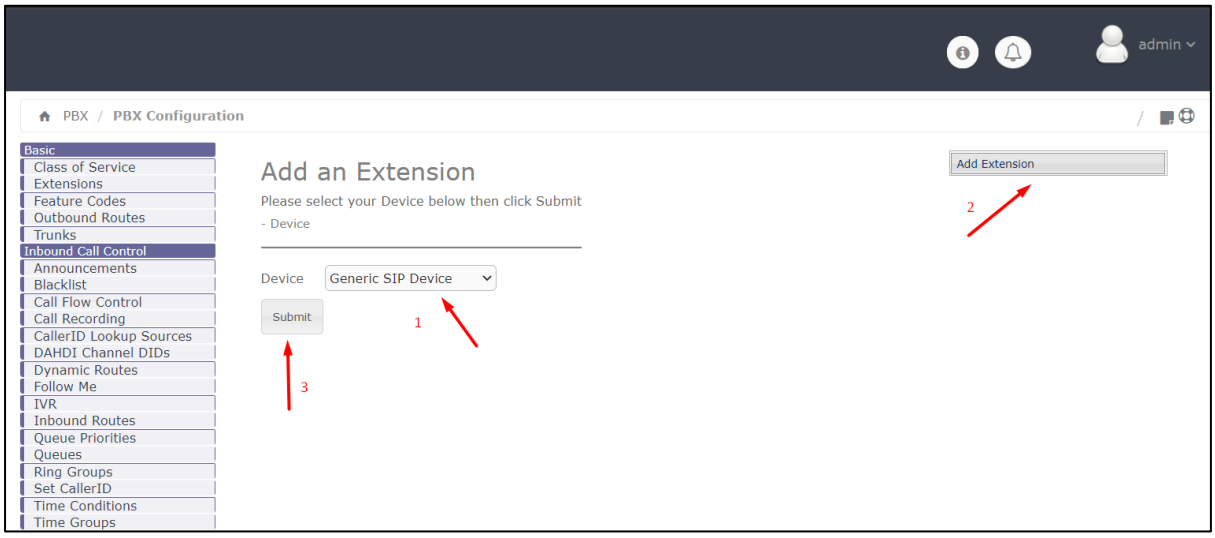
Interfaz de configuraciones de Issabel



Dentro de la configuración de PBX, en la Figura 170 se detalla el proceso a seguir; primero se selecciona el tipo SIP, segundo añadir extensión y tercero submit para aceptar la creación de esta.

Figura 170

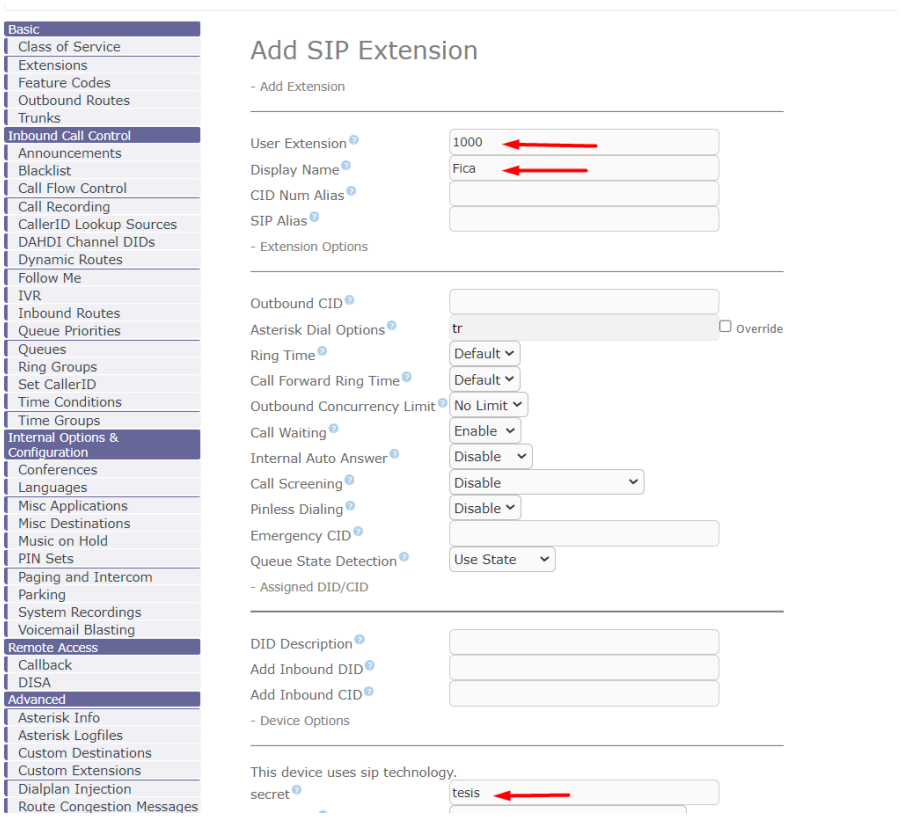
Creación de una extensión SIP



Para establecer una extensión hay que tomar tres parámetros, estos se muestran en la Figura 171 y son: el número de extensión para el usuario, el nombre con el que aparecerá y finalmente es obligatorio establecer una contraseña.

Figura 171

Configuración extensión SIP



Si todo se ha realizado correctamente, la extensión se añadirá como se observa en el recuadro rojo de la Figura 172.

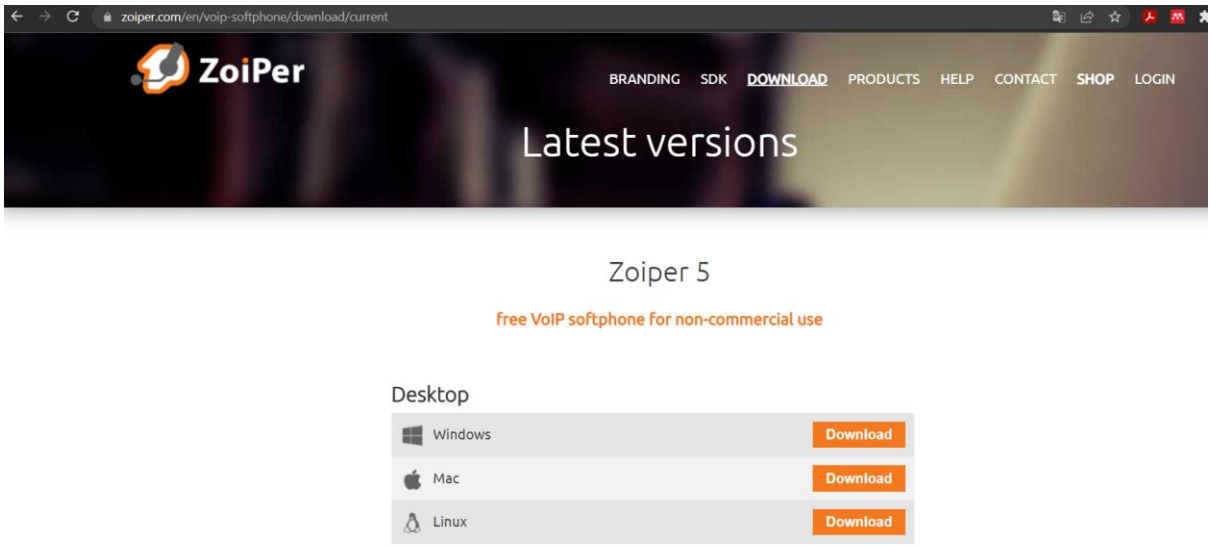
Figura 172

Extensión SIP establecida con éxito

Para comprobar que el servidor Issabel opera correctamente, es necesario hacer uso de un softphone, en este caso se usa Zoiper en su versión 5 que se descarga desde la URL https://www.zoiper.com/en/voip-softphone/download/current, en la Figura 173 se muestra la página oficial para descargar la aplicación Zoiper.

Figura 173

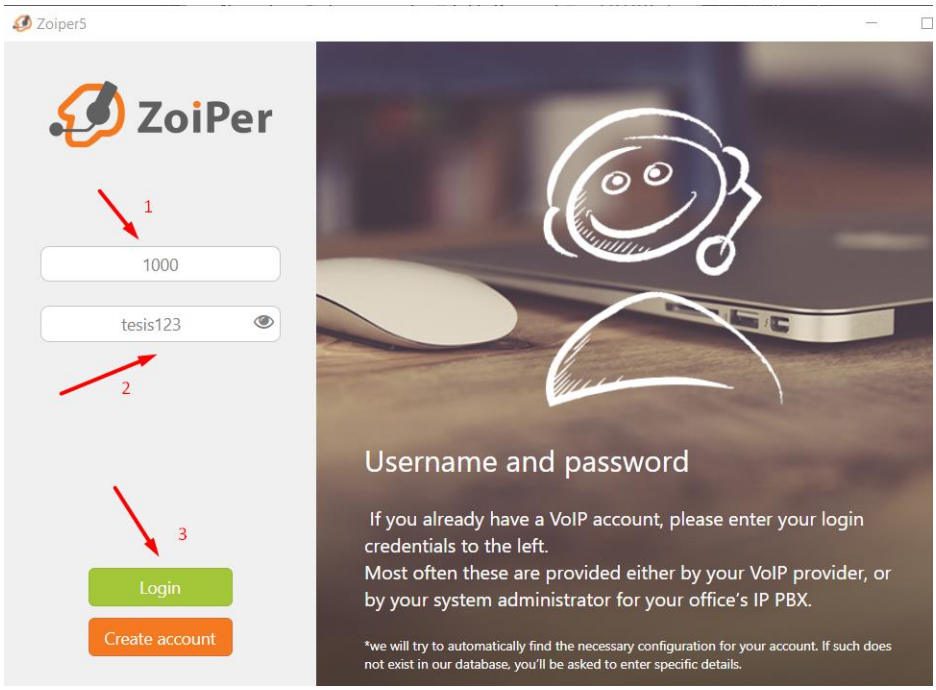
Página oficial de descarga de Zoiper



Una vez instalada la aplicación, se mostrará una interfaz como la Figura 174, además en ella se detalla el procedimiento a seguir; primero se debe digitar el número de extensión, segundo la contraseña de la extensión y tercero se da click en Login.

Figura 174

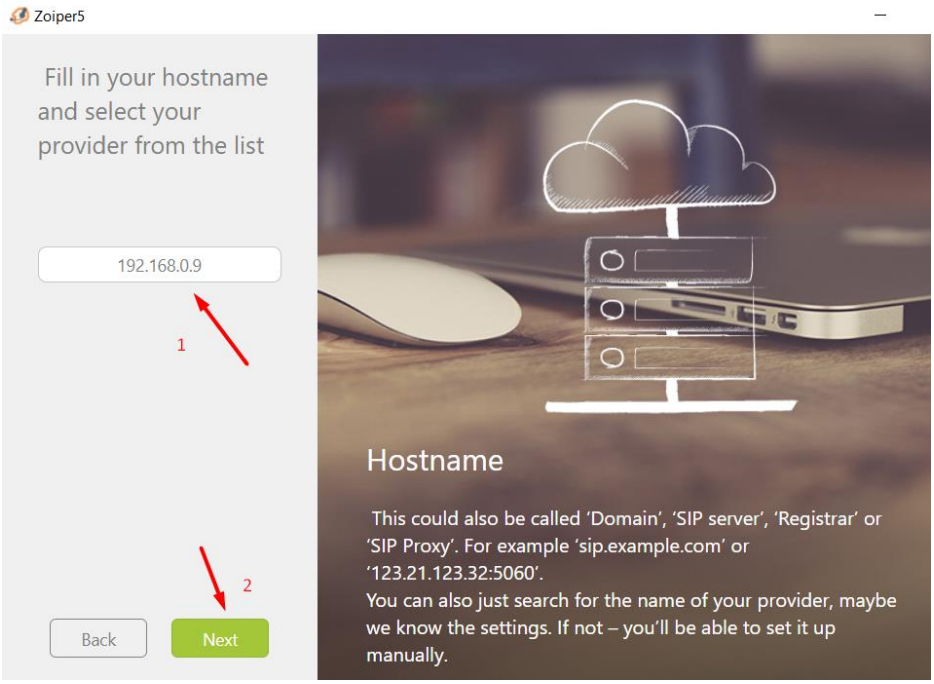
Ingreso de credenciales de la extensión SIP



Lo siguiente a ingresar es la dirección IP de servidor y luego darle a Next, igual que lo muestra la Figura 175.

Figura 175

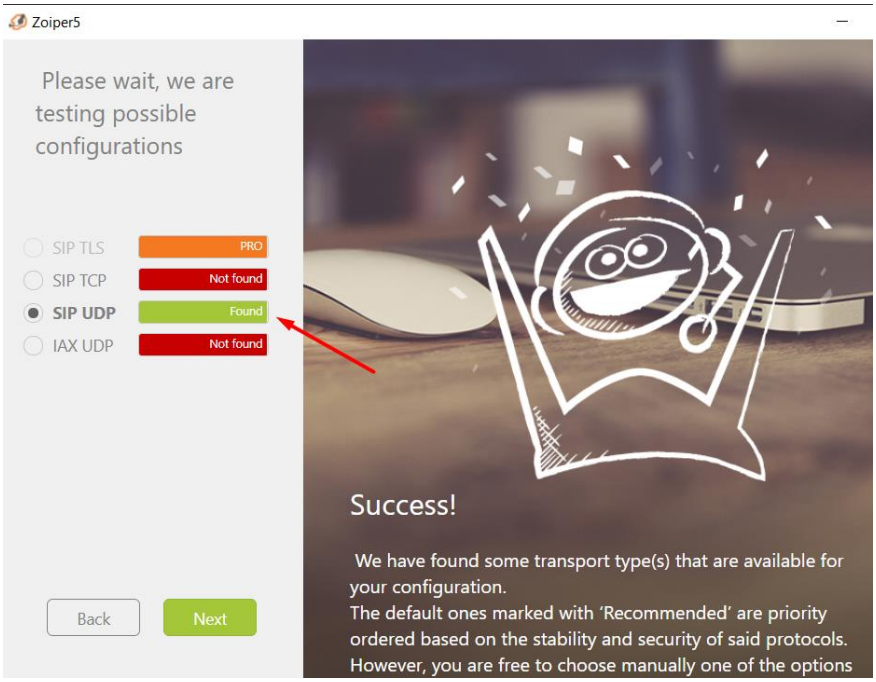
Ingreso de la dirección IP del servidor Issabel



Con todo lo anterior establecido, si todas las configuraciones se han realizado sin errores, Zoiper añadirá con éxito la extensión SIP como lo muestra la Figura 176.

Figura 176

Extensión SIP registrada con éxito en Zoiper



Para culminar con este proceso, en la Figura 177 se observa la extensión SIP con un tick verde indicando que se encuentra registrada y lista para llamar o recibir llamadas de otras extensiones.

Figura 177

Extensión enlazada y lista para comunicarse con otras extensiones