ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES

"IMPLEMENTACIÓN DE REDES DEFINIDAS POR SOFTWARE (SDN) SOBRE REDES IEEE 802.11 MEDIANTE MININET WI-FI."

TRABAJO DE TITULACIÓN:

Tipo: **PROPUESTA TECNOLÓGICA**

Para optar al Grado Académico de:

INGENIERA EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES

AUTORA: LISETH PAULINA CEVALLOS SÁNCHEZ

TUTOR: ING. VINICIO RAMOS VALENCIA Msc.

RIOBAMBA – ECUADOR

2018

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

ESCUELA DE INGENIERÍA EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES

El Tribunal de Titulación certifica que: el trabajo de tipo **PROPUESTA TECNOLOGÍCA** con tema **IMPLEMENTACIÓN DE REDES DEFINIDAS POR SOFTWARE (SDN) SOBRE REDES IEEE 802.11 MEDIANTE MININET WI-FI**, de responsabilidad de la señorita Liseth Paulina Cevallos Sánchez, ha sido minuciosamente revisado por los Miembros del Tribunal, quedando autorizada su presentación.

NOMBRE	**FIRMA**	**FECHA**
Dr. Julio Santillán Castillo **VICEDECANO DE LA FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**	.....	.....
Ing. Franklin Moreno **DIRECTOR DE LA ESCUELA DE INGENIERÍA EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES**	.....	.....
Ing. Vinicio Ramos Valencia **DIRECTOR TRABAJO DE TITULACIÓN**	.....	.....
Ing. Alberto Arellano Aucancela **MIEMBRO DEL TRIBUNAL**	.....	.....

Yo, Liseth Paulina Cevallos Sánchez, declaró ser la autora del presente trabajo de Titulación: **"IMPLEMENTACIÓN DE REDES DEFINIDAS POR SOFTWARE (SDN) SOBRE REDES IEEE 802.11 MEDIANTE MININET WI-FI",** previo a la obtención del título de INGENIERÍA EN ELECTRÓNICA, TELECOMUNICACIONES Y REDES, haciéndome responsable por las ideas, criterios, doctrinas y resultados expuestos en este trabajo, y declarando que el patrimonio intelectual de la misma pertenece a la Escuela Superior Politécnica de Chimborazo.

…….

Liseth Paulina Cevallos Sánchez

DEDICATORIA

La presente Tesis está dedicada a mi abuelita Lilia Vallejo y a mi madre Miryam Sánchez, por brindarme su apoyo y consejos para hacer de mí una mejor persona.

A mis hermanas Katherine y Polet, a mis tíos y primos, por su compañía y palabras de aliento a largo de toda mi carrera.

A Dios, quien supo guiarme por el buen camino darme fuerzas para seguir siempre adelante.

"Camina con la certeza de saber, que cada paso que das, está más cerca de alcanzar aquello que una vez solo fue un sueño."

Liseth P. Cevallos Sánchez

AGRADECIMIENTO

Agradezco a mi madre Miryam su apoyo incondicional, a mi abuelita Lilia sus consejos y paciencia me ayudaron para lograr seguir adelante para terminar mi carrera. A mis hermanas y a todas las personas que de una u otra manera siempre me alentaron a seguir adelante

A mis amigas especialmente a Alexandra, Génova, Gaby y Evelyn por todo su apoyo, por acompañarme en cada paso a lo largo de todos estos años, por sus consejos y confianza.

De manera especial quiero agradecer al Ing. Vinicio Ramos, al Ing. Alberto Arellano por sus valiosos aportes en el desarrollo y culminación de mi carrera estudiantil.

Liseth P. Cevallos Sánchez

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE GRÁFICOS

RESUMEN

El principal objetivo del presente trabajo fue la implementación de redes definidas por software (SDN) sobre redes IEEE 802.11 mediante Mininet Wi-Fi. Se recurrió a herramientas empleadas por las redes SDN para implementar enlaces inalámbricos y movilidad de hosts; se definieron conceptos básicos como: arquitectura SDN, protocolo OpenFlow, Mininet Wi-Fi y su eje principal el controlador. El controlador de SDN empleado fue OpenDayLight de Código Abierto; el cual ayuda a programar los caminos que seguirán los flujos de datos a lo largo de la red de una manera eficiente. La red fue simulada mediante el software Mininet Wi-Fi, emulador de Redes Inalámbricas Definidas por Software (SDWN). Los escenarios de pruebas fueron programados en lenguaje Python, a través de la consola de comandos de Mininet Wi-Fi. Se empleó la herramienta Wireshark para llevar a cabo capturas del tráfico en diversas interfaces, y conjuntamente con la herramienta iperf3 se envían flujos de datos TCP y UDP para obtener ancho de banda, perdida de datagramas, jitter en los enlaces de la red. Al realizar las pruebas respectivas para el escenario con los modelos de movilidad, se obtuvo que, en los modelos ReferencePoint y RandomWalk se generó un jitter de 5,64ms y 5,7ms, respectivamente, y valores de 0.73% y 0% en la pérdida de datagramas, debido a que RandomWalk crea patrones de movilidad más realistas y flexibles, mientras que ReferencePoint crea patrones de comportamiento grupal. Se concluye que al emplear un nuevo modelo de redes fue posible la implementación de soluciones eficientes para mejorar las redes inalámbricas; al optimizar un mayor despliegue de servicios y mejorar el rendimiento de las aplicaciones en toda la red.

Palabras clave: <TECNOLOGÍA Y CIENCIAS DE LA INGENIERÍA>, <REDES DE COMPUTADORES>, <REDES DEFINIDAS POR SOFTWARE (SDN)>, <IMPLEMENTACION DE REDES SDWN>, <COMUNICACIONES INALÁMBRICAS>, <MININET-WIFI (SOFTWARE)>, <OPENDAYLIGHT (SOFTWARE)>, <WIRESHARK (SOFTWARE)>

SUMMARY

The main objective of this work was the implementation of software defined networks (SDN) over IEEE 802.11 networks through Mininet Wi-Fi. It was resorted to tools used by SDN networks to carry out wireless links and station mobility; basic concepts were defined such as: SDN architecture, OpenFlow protocol, Mininet Wi-Fi and its main axis, the controller. The SDN driver was OpenDayLight Open Source which helps to reprogram paths that will follow the data flows throughout the network in an efficient way. The network was simulated by using the software Mininet Wi-Fi, emulator of Wireless Networks Defined by software (SDWN). The test scenarios were programmed in Python language, through the command console of Mininet Wi-Fi. The Wireshark tool was used to carry out traffic captures in various interfaces, and together with the iperf3 tool, TCP and UDP data streams were sent to obtain bandwidth, datagram loss, and jitter in the network links. When performing the respective tests for the scenario with the mobility models, it was obtained that in the ReferencePoint and RandomWalk models a jitter of 5.64ms and 5.7ms was generated, respectively, and values of 0.73% and 0% in the loss of datagrams, because RandowWalk creates patterns of group behavior. It is concluded that, by using a new network model, it was possible to implement efficient solutions to improve wireless networks; by optimizing a greater deployment of services and improving the performance of applications throughout the network.

Keywords: <TECHNOLOGY AND SCIENCE OF ENGINEERING>, <NETWORKS OF COMPUTERS>, <NETWORKS DEFINED BY SOFTWARE (SDN)>, <IMPLEMENTATION OF SDWN NETWORKS>, <COMMUNICATIONS WIRELESS>, <MININET-WIFI (SOFTWARE)>, <OPEANDAYLIGHT (SOFTWARE)>, <WIRESHARK (SOFTWARE)>

INTRODUCCIÓN

En la actualidad la densidad de usuarios conectados a Internet a nivel mundial va creciendo de manera constante, así, en el año de 1990 el tráfico de Internet era de 1 TB/Mes mientras que para el 2015 fue de 28,06 TB/seg, en años posteriores se provee un crecimiento exponencial. Así, según el Ministerio de Telecomunicaciones y de la Sociedad de la Información (Mintel) el año 2016, en Ecuador el 92.11% de la población nacional usaron Wi-Fi en sus dispositivos móviles. (ACOSTA, 2017, pp. 2-4) (INEC, 2016, pp. 6 -11).

La instancia de un número mayor de dispositivos conectados a Internet, a través Wi –Fi como: ordenadores portátiles, tablets, u otros dispositivos móviles (smartphones); incrementan la demanda masiva de información, nuevos requerimientos en calidad de servicios (QoS), un alto nivel de disponibilidad y un mayor ancho de banda ha provocado que la infraestructura de red sea insuficiente para satisfacer las nuevas demandas.

Los nuevos requerimientos de las redes obligan a una solución eficiente y viable para su diseño, arquitectura y gestión, una mejora en diversos aspectos; con lleva como resultado al desarrollo de una nueva tecnología de red (ESPAÑA, 2016, pp. 8-10).

Las Redes Definidas por Software (SDN) aportan ventajas frente a las redes tradicionales tales como: potencial para revolucionar todas las redes, la reducción de costo tanto en equipamiento como operacional, añaden mayor flexibilidad al momento de controlar toda la red y permite acelerar la introducción de nuevos servicios (NÚÑEZ, 2015, pp. 7-32).

Las SDN no solo se limitan a la investigación en redes cableadas, de igual manera pueden llevarse a cabo la experimentación en el segmento de mayor crecimiento de las telecomunicaciones: las redes inalámbricas, comúnmente se conoce como: Redes Inalámbricas Definidas por Software (SDWN). Para su despliegue experimental, los simuladores, emuladores y testbest son herramientas que ayudan a validar, evaluar y experimentar los conceptos SDWN, más estos presentan algunos inconvenientes. Así, dentro de las herramientas de investigación de SDWN encontramos el emulador Mininet Wi-Fi; software empleado para la virtualización de estaciones y puntos de acceso inalámbricos, permitiendo la experiencia de SDWN más realistas. (FONTES, y otros, 2017,pp. 5-10)

ANTECEDENTES

El modelo de red tradicional manejado en la actualidad, considera a toda la red como un conjunto de elementos independientes, relacionados entre sí; donde las políticas de configuración son establecidas manualmente en cada uno. Se exterioriza una relación entre el funcionamiento de los servicios prestados y la persona encargada de las configuraciones, donde la gestión de la red presenta una alta complejidad y falta de flexibilidad (LEYTON, 2016, pp. 9-20).

Las redes vienen trabajando de manera eficaz desde hace muchos años, pero debido a la demanda de nuevos servicios, el incremento de los dispositivos conectados a Internet, el acceso a los servicios de la nube, entre otros; por lo cual, la capacidad de satisfacer los diferentes requerimientos se ve afectado de manera significativa (ALBÁN, y otros, 2015, p. 21).

Dentro de este contexto se definen las Redes Definidas por Software (SDN), presentadas como un modelo emergente orientado a facilitar la implementación de servicios de red de manera dinámica y escalable; centralizando la gestión y apartando el plano de los datos con los de control, introduciendo un elemento lógico centralizado denominado Controlador como eje fundamental de su funcionamiento.

La Open Networking Foundation (ONF) líder en el desarrollo de las SDN, lleva a cabo la normalización de protocolos para su estandarización; uno de sus aportes en cuanto a redes inalámbricas Wi-Fi es el software Mininet Wi-Fi (OPEN NETWORKING FOUNDATION, 2010), desarrollo como bifurcación de Mininet para simular redes SDN inalámbricas (SDWN).

Donde, se lleva a cabo la emulación de estaciones inalámbricas y puntos de acceso inalámbricos virtuales, manteniendo cada una de las capacidades de las SDN originales, por tanto, su uso añade un valor significativo en el campo de las redes inalámbricas.

En el país desde hace algún tiempo se ha venido trabajando en el desarrollo de redes definidas por software, aportando ideas innovadoras en este campo de investigación, así podemos mencionar: Escuela Superior Politécnica de Chimborazo (YÁNEZ, y otros, 2015, pp. 9-12), Universidad Central del Ecuador (ESPAÑA, 2016, pp. 8-10), Escuela Politécnica Nacional (MOSCOSO, 2016, pp. 28-30) (CHICO, 2013, pp. 30-40), y muchas otras.

FORMULACION DEL PROBLEMA

¿La implementación de SDN sobre redes IEEE 802.11 permitirá un mejor análisis y emulación de la movilidad de las estaciones inalámbricas de una red?

SISTEMATIZACIÓN DEL PROBLEMA

- ¿Cuál es el estado actual del estudio de las Redes Definidas por Software sobre redes IEEE 802.11?
- ¿Cuáles serán las ventajas de la implementación de las redes SDN con estaciones y puntos de acceso inalámbricos?
- ¿Cuáles serán las diferencias entre los tipos de movilidad definidos en Mininet Wi-Fi?
- ¿Cuáles serán los parámetros de mayor incidencia en las SDN Wireless?

JUSTIFICACION DEL TRABAJO DE TITULACIÓN

Justificación Teórica

Las Redes Definidas por Software (SDN) surgen de la necesidad de implementar una mejora viable a los inconvenientes presentados en la implementación de servicios de red de manera: dinámica, económica y escalable; facilitando al administrador de red la gestión de los servicios a bajo nivel.

En las SDN se efectúan la gestión de los dispositivos de red de manera distinta a la habitual; tanto el plano de control, como el plano de datos son tratados de manera independiente uno del otro, es decir, la separación del hardware y software, donde el control se lleva a cabo por dispositivos denominado controladores (ARTECHE, 2014, pp. 8.20).

En la actualidad encontramos varias tecnologías las cuales aportan múltiples beneficios en la gestión de las redes, pero sin duda una de las mejores es la SDN, las cuales brindan una mejora significativa en la gestión de los dispositivos de red (MARÍN, 2016, pp. 30-40) .

La posibilidad de emular redes inalámbricas Wi-Fi con la ayuda del más reciente aporte en el campo de las SDN, por parte de un grupo de desarrolladores (FONTES , y otros, 2015, pp 2-6) de la

Universidad de Campinas; la herramienta Mininet Wi-Fi facilta la emulación de redes inalámbricas virtuales, por consiguiente, el presente documento detalla la información necesaria acerca de la investigación redes inalámbricas, enfatizando su emulación en entornos virtuales. Creando una base para investigaciones futuras en lo referente a redes SWDN.

Justificación Aplicativa

Para la emulación de las SWDN se maneja la herramienta Mininet Wi-Fi, la cual incorpora modelos de movilidad para estaciones inalámbricos; de igual manera se facilita la visualización de los escenarios a través de MiniEdit, interfaz gráfica de Mininet y con scripts programados en Python, usando estas herramientas se simularon los diferentes escenarios. El eje central de la red será el controlador OpenDayLight, encargado de optimizar los servicios de la red y maximizar su funcionamiento.

Para un análisis completo de toda la red, la captura de tráfico a través de Wireshark; aporta la información necesaria para conocer cada tipo de movilidad permitidos y emulados en Mininet Wi-Fi. Para fines de investigación y estudio, toda la información será documentada, sirviendo de refuerzo para el desarrollo de una red SDN sobre redes IEEE 802.11, en entornos virtuales.

OBJETIVOS

Objetivo General

Implementar redes definidas por software (SDN) sobre redes IEEE 802.11 mediante Mininet Wi-Fi.

Objetivos Específicos

- Analizar la arquitectura de las SDN inalámbricas, protocolo OpenFlow, software Mininet y Mininet Wi-Fi, controlador SDN.
- Simular los elementos de la red mediante el entorno de simulación Mininet Wi-Fi y el manejo de la interfaz gráfica de usuario MiniEdit.
- Comparar los diferentes tipos de movilidad de estaciones inalámbricas y puntos de acceso inalámbricos programados en scripts de Python.
- Evaluar los diferentes tipos de movilidad en los escenarios emulados en Mininet WiFi

CAPITULO I

1. MARCO TEÓRICO

1.1. Estándar IEEE 802.11

La especificación IEEE 802.11 (ISO/IEC 8802-11) estándar internacional que define características de una red de área local inalámbrica (WLAN), Wi-Fi (Wireless Fidelity) nombre de la certificación otorgada por la Wi-Fi Alliance para nombrar las conexiones inalámbricas, para garantizar la compatibilidad entre dispositivos que utilizan el estándar (WI-FI ALLIANCE, 2017).

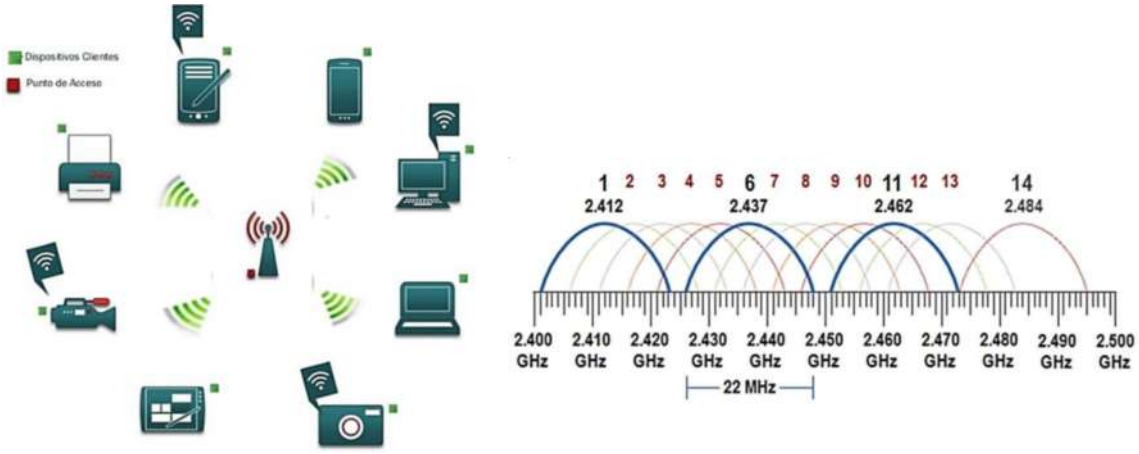
El IEEE adopta el estándar 802.11 en 1997 y se convierte en el primer estándar WLAN. De acuerdo a la IEEE, 802.11 principalmente controla las Capas 1 y 2 de la pila de referencia OSI, las cuales son la capa física y la capa de enlace de datos.

Las redes inalámbricas Wi-Fi o Wireless permite montar redes locales sin emplear ningún tipo de cableado, utilizando infrarrojos u ondas de radio o frecuencias no normalizadas, es decir, en el espectro libre de frecuencias (INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, 2008).

1.1.1 Componentes

Para crear una red sin cables muy sencilla el marco ideal es disponer de un punto de acceso o router inalámbrico que conecte varios equipos y, a la vez, brinde servicio de Internet a través de una línea ADSL o cualquier tipo de conexión.

La Figura 1-1 (a) describe el esquema básico de una red Wi-Fi, la cual normalmente se compone de un Punto de Acceso (AP), dispositivo o estación base que dispone de una antena de emisión y recepción creando un área de cobertura para los usuarios, y los dispositivos clientes, entre los cuales podemos encontrar computadores, portátiles, teléfonos móviles los cuales son aprovechan la tecnología Wi-Fi.



(a) (b)

Figura 1-1 (a) Esquema Wi-Fi, **(b)** Canales Wi-Fi en la banda de frecuencia 2.4GHz.

Realizado por. – CEVALLOS, Liseth, 2017,

Fuente: http://www.sincables.ec/2015/11/canales-wi-fi-como-configurarlos-i/

Se define una distancia de 100m para una conexión Wi-Fi, sin embargo, con lo nuevos estándares la distancia teórica se incrementa algunos metros para brindar mayor cobertura (LOPEZ, 2014, pp. 4-20). Cada uno de los dispositivos clientes suelen operar en varios canales en bandas de frecuencias de 2,4 GHz y 5 GHz, bajo ninguna licencia para su uso. La banda de 2.4GHz es la más concurrida, constando de un total de 14 canales para Wi-Fi (REID, y otros, 2003, pp. 5-10).

En América, solo los canales 1 al 11 son compatibles. Dentro del espectro de frecuencias, la superposición de los canales provoca una reducción en el número de canales utilizables; debido a que los puntos de acceso transmiten en un canal determinado, pero la señal se propaga sobre cerca de cuatro canales llegando a 20MHz y 22MHz de ancho de banda (GALEON, 2010). Así, se recomienda que los canales inalámbricos sean el 1, 6 y 11 para evitar la superposición, Figura 1-1(b).

1.1.2 Estándares

Existen varios estándares de redes inalámbricas Wi-Fi. La Tabla 1-1 proporciona un resumen de las versiones más comunes de este estándar, así como una breve descripción de cada una de ellas (IEEE WORKING GROUP, 2017).

Tabla 1-1 Versiones estándar IEEE 802.11.

Características	Versiones						
	802.11 legacy	**802.11 (a)**	**802.11 (b)**	**802.11 (g)**	**802.11 (n)**	**802.11 (ac)**	**802.11 (ad)**
Año	1997	1999	1999	2003	2009	2013	2014
Banda (GHz)	2.4 -2.5	5	2.4	2.4	2.4 o 5	2.4 y 5.5	2.4 , 5 y 60
Modulación	DSSS, FHSS	OFDM	DSSS	DSSS, OFDM	OFDM	OFDM	OFDM
Ancho de banda de canal (MHz)	20	20	20	20	20 o 40	20, 40,80 y 60	2,160
Velocidad Máxima (Mbps)	2	54	11	54	600	1300	7000
Alcance exterior (m)	100	120	140	140	250	-	-
Alcance interior (m)	20	35	35	38	70	35	3.3
Compatibilidad con versiones anteriores	-	No	No	802.11 b	802.11 b/g	802.11 b/g/n	802.11 b/g/n/ac

Realizado por. - CEVALLOS, Liseth. 2017.

Fuente.- (LOPEZ, 2014, pp. 4-20)

1.2. Redes Definidas por Software (SDN)

En la actualidad, las redes deben responder y ajustarse de forma dinámica bajo políticas actualizadas, para reducir el trabajo manual y los costos asociados a la gestión y administración de las mismas. Se requiere implementar y ejecutar nuevas aplicaciones dentro y por encima de las redes; desafío afrontado por las SDN (ALBÁN, y otros, 2015, p. 21).

Las SDN se remontan a las décadas en las que el Internet entraba en auge, donde los primeros pasos se basaban en la transferencia de archivos, y el correo electrónico era una las aplicaciones más atrayentes. Así, los primeros esfuerzos relacionados a dicha tecnología fueron: ActiveNetworking, Open Signaling, DCAN y Ethane.

Dentro de la red, los dispositivos de conectividad están integrados por dos componentes importantes: el plano de control y el plano de datos. El plano de los datos se encarga de la transmisión de los datos, mientras el plano de control determina el camino para asegurar la transferencia de dichos datos (NETWORK WORLD, 2013).

Debido al constante crecimiento de los requerimientos de la red, se pensó en separar los planos de control y de datos, dando paso a una interfaz abierta entre los planos y el control lógico de la red (LEYTON, 2016, pp. 9-20).

1.2.1. Definición

Las Redes Definidas por Software comprenden una arquitectura de red, donde se aborda de manera diferente la creación de redes; permiten una naturaleza dinámica y un alto consumo de ancho de banda para las aplicaciones modernas (MAGAZCITUM, 2013).

Esta arquitectura de red permite, separar el plano de control del plano de datos empleando controladores basados en software, responsables de gestionar el flujo de datos entre el controlador y los conmutadores (switch). Los conmutadores comparten toda su información de la red al controlador y pasan a ser unidades de conmutación de tráfico (OPEN NETWORKING FOUNDATION, 2010).

El elemento del plano de control, el controlador y los elementos del plano de datos, los switchs están normalmente separados, incluso físicamente, su comunicación se lleva por medio de un protocolo estándar. De esta manera, el controlador percibe a la red como una sola unidad.

La ONF (Open Networking Foundation) es líder en el avance de las SDN y la estandarización de los elementos de la arquitectura SDN como el protocolo OpenFlow. OpenFlow es la interfaz diseñada específicamente para las redes SDN.

Por consiguiente, esta tecnología virtualiza la red independizándola de la infraestructura física de la que disponga, permitiendo mejoras en aspectos tales como la seguridad y monitorización de toda la red (YÁNEZ, y otros, 2015, pp. 9-12).

1.2.2. Características y Ventajas

Las SDN ayudan a gestionar la naturaleza dinámica de las aplicaciones actuales y para ello, el elemento fundamental para la construcción de soluciones SDN es el protocolo OpenFlow. La Figura 2-1 enumera algunas de las características principales de la arquitectura SDN (CITRIX SYSTEMS, 2017).

Figura 2-1 Características de las SDN.

Realizado por. – CEVALLOS, Liseth, 2017.

1.2.3. Arquitectura

En la Figura 3-1 se representa la arquitectura de SDN compuesta por tres capas: Infraestructura, Control y Aplicaciones, accesibles desde interfaces de programación de aplicaciones (API´s) abiertas: Southbound "Hacia el Sur" y Northbound "Hacia el Norte" (ESPAÑA, 2016, pp. 8-10).

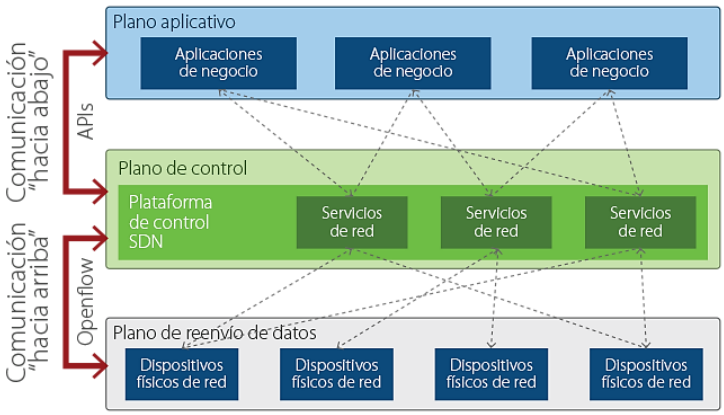


Figura 3-1 Arquitectura SDN.

Fuente. – http://www.magazcitum.com.mx/?p=2353, 2013. Redes complejas hechas fácilmente.

Capa de Infraestructura.- Hace relación a los dispositivos de red como: router y switches, tanto físicos (hardware) y/o virtuales (software) los cuales proporcionan la conmutación de paquetes y de reenvío.

Capa de Control.- Desde aquí opera el controlador, el cual tiene el control de las políticas de reenvió de paquetes; toma decisiones en base a la topología completa de la red y proporciona API´s abiertas tanto para la parte superior e inferior (LEYTON, 2016, pp. 9-20).

Capa de Aplicaciones.- Consta de las aplicaciones solicitadas por los usuarios finales, como por ejemplo: Aplicaciones de voz, video, servicios de red. Logra que las configuraciones resulten simples.

La interfaz hacia el Sur o Southbound: permite la comunicación del controlador SDN con los dispositivos de red de la capa de Infraestructura. El protocolo OpenFlow es su elemento principal. De igual manera, la interfaz hacia el norte Northbound permite la comunicación entre el controlador hacia las aplicaciones; permitiendo incluir funciones básicas de red, como ruteo, seguridad para la administración de los servicios de red (MAGAZCITUM, 2013).

1.3. SDN vs Redes Tradicionales

La complejidad de las redes actuales imposibilita aplicar un conjunto coherente de seguridades, calidad de servicio y otras políticas dentro de las mismas. Los servidores de aplicaciones o de datos, la virtualización de equipos, los servicios en la nube, entre otros, conducen a que las empresas decidan examinar su infraestructura de manera inmediata. La Figura 4-1 presenta de forma simplificada cada una de las arquitecturas, tanto de la red tradicional como de las redes SDN, (NÚÑEZ, 2015, pp. 7-32).

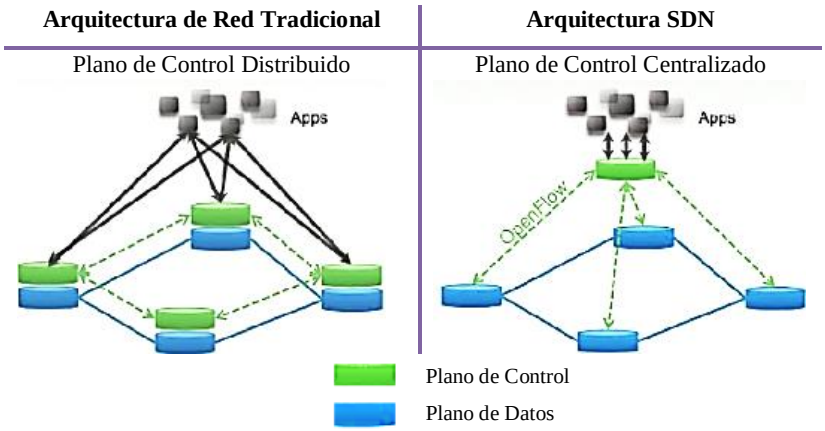


Figura 4-1 Comparativa de las SDN y redes tradicionales.

Fuente. – https://alln-extcloud-storage.cisco.com/ciscoblogs/silver1-550x304.png, 2013. SDN 101: What It Is, Why It Matters, and How to Do It

Así, las Redes definidas por Software, se presentan como una solución estandarizada para cumplir las necesidades de las empresas y los usuarios de la red, dando paso a la transformación de la arquitectura de la red convencional. La Tabla 2-1 identifica alguna de las diferencias entre la red tradicional y las redes SDN (SAVU, y otros, 2014, pp. 7-37).

Tabla 2-1 Diferencias entre red tradicional y SDN.

Red Tradicional	SDN
- Envió de paquetes basados en coincidencia de tablas	- Tablas de enrutamiento abiertas
- Configuraciones en cada dispositivo	- Configuración centralizada
- Incapaz de ser escalable	- Escalabilidad
- Difícil de aplicar políticas	- Dinámica
- Tablas de flujo en cada dispositivo	- Tabla de enrutamiento abiertas
- Nuevas aplicaciones en mayor tiempo	- Nuevas aplicaciones en menos tiempo-

Fuente.- (ESPAÑA, 2016, pp. 8-10)**,** Diferencias entre red tradicional y SDN**.**

1.4. Protocolo OpenFlow

OpenFlow surge a raíz del proyecto de investigación: "OpenFlow: Enabling Innovation in Campus Networks" (2008), en la Universidad de Stanford. Su primera versión OpenFlow (versión 1.1) se lanzó en febrero de 2011 y la segunda (versión 1.2) fue supervisada por la ONF; organización que controla las especificaciones de las SDN. Actualmente la versión en vigor es la 1.4.0 (ALBÁN, y otros, 2015, p. 21).

OpenFlow se define como un protocolo emergente de comunicación abierto, es la primera interfaz de comunicaciones estándar definida entre los planos de control y datos de las SDN, tanto física como virtual. Al centralizar el control de los dispositivos de la capa de infraestructura, simplifica la administración de las redes y potencia la capacidad de programación permitida por las SDN.

El protocolo OpenFlow se implementa entre los dispositivos de la capa de infraestructura de red y el controlador SDN. OpenFlow lleva a cabo el concepto de flujos para identificar el tráfico de la red de acuerdo a varios parámetros como: el patrón de uso de la red, las aplicaciones y los requerimientos específicos de cada una de ellas (RAMOS, Catalina, 2012, pp. 4-6).

Los flujos anteriormente mencionados se pueden entender como una serie de paquetes con encapsulación TCP, van de una dirección MAC o IP a otra; o como paquetes de una determinada VLAN.

Este protocolo puede ser desplegado en las redes existentes de diferentes empresas tomando en cuenta que los dispositivos de red, tanto físicos y virtuales deben poder soportar el protocolo. Así también, la fácil migración a esta nueva tecnología es debido a la imparcialidad en cuanto a los diferentes proveedores de equipo de red (MOSCOSO, 2016, pp. 28-30).

1.4.1. Arquitectura

La arquitectura de OpenFlow consta de tres conceptos básicos: (1) la red se construye con conmutadores compatibles con OpenFlow que componen el plano de datos; (2) el plano de control consta de uno o varios controladores de OpenFlow; (3) el protocolo OpenFlow propiamente dicho, detallados en la Figura 5-1 (WOLFGANG, y otros, 2014: pp. 5-10).

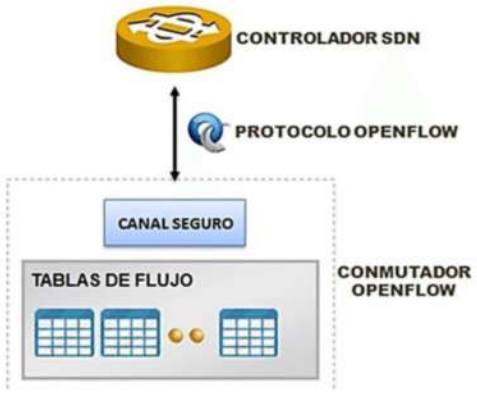


Figura 5-1 Arquitectura OpenFlow.

Fuente. – https://www.researchgate.net/profile/Yanko_Antonio_Muro, 2016.

Un conmutador OpenFlow es un dispositivo de conectividad creado o modificado para conectarse con el controlador, para recibir reglas las cuales definirán los patrones de flujo de tráfico mediante el protocolo OpenFlow.

El protocolo OpenFlow toma decisiones sobre cómo deben ser reenviados los paquetes de los switches al o los controladores. Una aplicación de gestión se ejecuta en las interfaces del controlador que une todos los conmutadores a la red, facilitando la configuración de diversos caminos de reenvío los cuales dispondrán de todo el ancho de banda.

El controlador es la unidad central o núcleo, mueve la funcionalidad del plano de control de la red OpenFlow. Otorga una interfaz para crear, modificar y controlar los flujos de tablas de un conmutador. Corre sobre un servidor de red, puede ser parte de un conjunto de conmutadores OpenFlow, uno por switch o uno para cada conjunto (YÁNEZ, y otros, 2015, pp. 9-12).

1.4.2. Conmutador Openflow

El conmutador OpenFlow trabaja con las denominadas tablas de flujo, los switch y router Ethernet modernos contiene dichas tablas de flujos. Cada tabla de flujo es diferente según el fabricante del dispositivo de red, sin embargo, se han identificado un conjunto de funciones comunes entre ellos (ALBÁN, y otros, 2015, p. 21).

A cada entrada de la tabla de flujos se la relaciona a una determinada acción, que puede ser: (1) enviar el paquete a un determinado puerto para su enrutamiento en la red; (2) encapsular y enviar paquetes al controlador de un determinado flujo de datos; y (3) eliminar los paquetes de un flujo

de datos principalmente para prevenir ataque en la seguridad de la red (CENTENO, y otros, 2014, pp 3-5).

Un conmutador OpenFlow consiste en una tabla de flujos la cual contiene las entradas de flujo, para desempeñar la búsqueda de paquetes; un canal seguro, que permite conectar el dispositivo de conectividad al servidor controlador y las tablas de grupo compuestas por entradas de grupo; los grupos permiten identificar que el mismo conjunto de acciones deben ser llevadas a cabo por múltiples flujos (ESPAÑA, 2016, pp. 8-10) . Los componentes se detallan en la Figura 6-1.

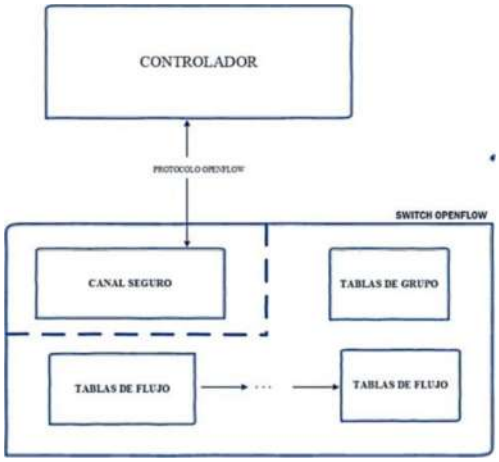


Figura 6-1 Arquitectura switch OpenFlow.

Realizado por. –CEVALLOS, L. 2017.

Además, los conmutadores OpenFlow pueden ser clasificados de acuerdo a la implementación del protocolo. Así, en el conmutador OpenFlow dedicado: el protocolo ya viene instalado en el dispositivo y este solo se limita a enviar el tráfico ya definido por el controlador. En el conmutador OpenFlow habilitado: se añade actualizaciones de firmware para que el dispositivo pueda ser utilizado dentro de una red SDN (CHICO, 2013, pp. 30-40).

1.4.2.1. Tablas de Flujo

Conjunto de registros instalados en los conmutadores, indican a cada dispositivo que hacer con el tráfico de la red. El conmutador consulta su tabla de flujos antes de procesar el tráfico conforme a su contenido. En caso de no encontrar ninguna coincidencia, se hace una consulta con el controlador quien determina la acción a seguir.

Alguna de las ventajas de dichas tablas es que permiten, debido a la toma de decisiones centralizadas aplicar QoS (calidad de servicios), rutas de menor latencia, entre otras (VELASCO ,

2016,p. 4-15) . Una entrada de una tabla de flujos OpenFlow posee los campos detallados en la Figura 7-1 (MOSCOSO, 2016, pp. 28-30).

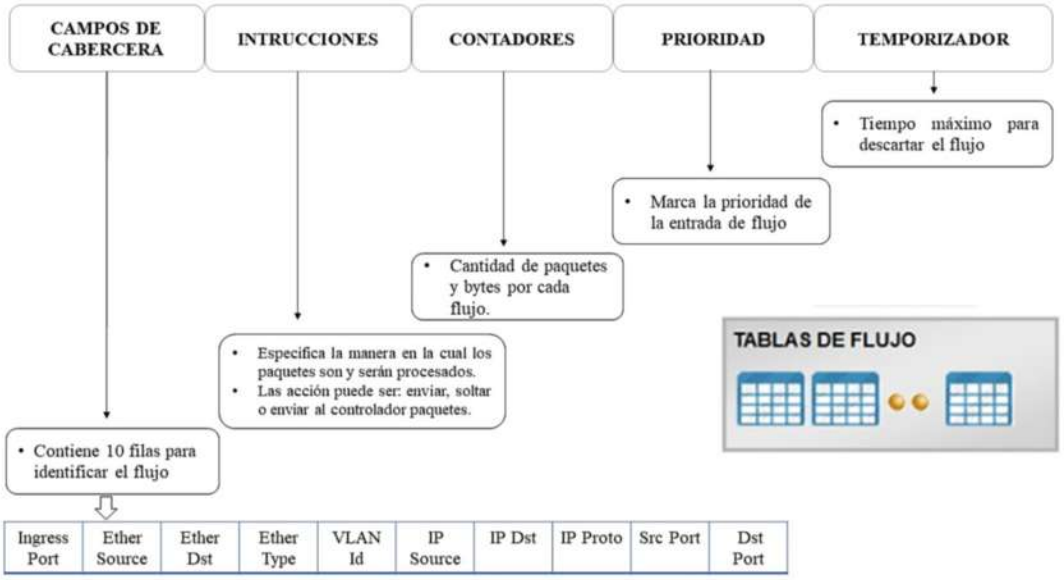


Figura 7-1 Campos de una Tabla de Flujos.

Realizado por. –CEVALLOS, L. 2017.

Fuente.- (OPEN NETWORKING FOUNDATION, 2015, pp. 4-100)

Cuando se recibe un paquete, el switch comienza una búsqueda en la primera tabla de flujo; dichas tablas se enumeran desde cero, la búsqueda será ejecutada también en otras tablas de flujo. De ser el caso de no encontrar coincidencia se actualiza el número de la tabla de flujo y salta a otra tabla, hasta encontrar una coincidencia. Al no encontrar coincidencia se devuelve al controlador

Cada flujo de entrada es analizado en la tabla, se busca el paquete entrante con la mayor prioridad, se aplican las distintas instrucciones, para finalizar se envía el dato a la siguiente con el conjunto de acciones (MARÍN, 2016, pp. 30-40).

1.4.3. Tipos de comunicación OpenFlow

Openflow lleva a cabo una comunicación entre el controlador y los conmutadores OpenFlow para transmitir acciones e información. Existen tres tipos de comunicaciones soportadas por el protocolo. La comunicación del controlador al conmutador, la comunicación Asíncrona y la Sincrónica.

Comunicación controlador – conmutador. - iniciado por el controlador, no siempre espera una respuesta del conmutador. Se efectúan procesos como: configuración del switch, gestión de las tablas de flujo, estados de las tablas y la capacidad del conmutador en cualquier momento.

Comunicación Asíncrona.- son enviados desde el conmutador al controlador. Denotan un cambio en el switch o en el estado de la red, denominado:evento. Un evento packet-in (paquete dentro) es un mecanismo del switch para enviar un paquete al controlador, si este no tiene coincidencia de entrada en el mismo.

Comunicación Síncrona.- enviada sin ser solicita ya sea por el conmutador o el controlador, sin importar su dirección. Se utiliza para resolver problemas de conexión entre el conmutador y controlador (YÁNEZ, y otros, 2015, pp. 9-12) (RAJASRI, y otros, 2011, pp. 23-26).

1.5. Controlador SDN

El controlador es una entidad de software, es el responsable de tomar las decisiones con respecto a las entradas en una tabla de flujo de acuerdo a reglas y políticas de enrutamiento. Puede ser ejecutado en diversas plataformas computacionales o ser ejecutado como una máquina virtual.

El control lógico centralizado pretende disponer de un controlador SDN con un alcance global y la unión de todos los dispositivos de red; los cuales comparten información y trabajan conjuntamente como una solo entidad (VELASCO , 2016,p. 4-15).

Un controlador SDN debe manejar un gran número de conmutadores, por lo cual el concepto de escalabilidad al momento de seleccionar un controlador es de suma importancia. Dependiendo de las necesidades actuales y futuras de la red, un solo controlador debería ser capaz de administrar al menos 100 switches sin problemas de rendimiento.

Dada la amplia diversidad de controladores SDN estos pueden tener soluciones: (1) Comerciales tales como Big Switch, Cisco, Juniper; y (2) Código Abierto: NOX, Beacon, OpenDayLight entre otros.

La diferencia entre los controladores está en el lenguaje de programación y la plataforma en la que trabaje, más la funcionalidad es la misma ya que todos soportan trasmitir y recibir paquetes OpenFlow. (NÚÑEZ, 2015, pp. 7-32).

1.5.1. Controlador OpenDayLight (ODL)

Proyecto de software de código abierto bajo la Fundación Linux con el objetivo de promover la adopción e innovación de las SDN, buscando convertirse en una plataforma abierta tanto para usuarios finales, clientes y empresas, para obtener nuevas soluciones de manera conjunta.

OpenDayLight al ser una plataforma abierta, dispone de la opción de optar por tecnologías de diversos fabricantes, tales como: Cisco, Citrix, IBM, Juniper Networks, RedHat y VMWare algunos fundadores del proyecto (SDX CENTRAL , 2016).

Una de las ventajas más sobresalientes de ODL reside en el soporte para el protocolo OpenFlow, sin embargo, también puede soportar otros estándares SDN abiertos. Proporciona la capacidad de desplegar una variedad de entornos de red y soporta un marco de control modular (ALVAREZ, 2015, pp- 21-49).

El controlador ODL expone las API´s abiertas (northbound) para ser utilizadas por las aplicaciones. Dichas aplicaciones utilizan al controlador para recopilar información sobre la red, ejecutar algoritmos para análisis, entre otras. El desarrollo de ODL está impulsado por una gran comunidad a nivel mundial que actualiza la plataforma cada seis meses y la adapta continuamente (LINUX FOUNDATION, 2016).

OpenDayLight está en constante cambio desde sus inicios, así, se dispone de sus diversas versiones de software cada una con características particulares, como se observa en la Figura 8-1.

Figura 8-1 Versiones de Software de OpenDayLight.

Realizado por. - CEVALLOS, L, 2017.

Fuente.- (LINUX FOUNDATION, 2016)

El primer lanzamiento de código de software para ODL es Hydrogen (febrero,2014). Cuenta con tres ediciones diferentes para ayudar a los usuarios a comenzar: la edición base, la edición de virtualización y la edición de los proveedores de servicios. Cuenta con 1.3 millones líneas de código (ALVAREZ, 2015, pp- 21-49).

El segundo código para el Controlador ODL es Helium (octubre,2014) . Con una nueva interfaz de usuario y un proceso de instalación más sencillo y personalizable, gracias al uso de Apache Karaf. Esta versión tiene integración con OpenStack (proyecto de computación en la nube para proporcionar una infraestructura como servicio).

En junio de 2015 se lanzó la tercera versión de software, Lithium (Junio,2015). Donde el OpenDayLight reposiciona el Controlador OpendayLight como la Plataforma OpenDayLight. La próxima versión como se muestra es Beryllium (febrero,2016), resuelve los desafíos clave de la red relacionados a la entrega de servicios automatizados, optimiza recursos de NFV, mejora la GUI (interfaz gráfica de usuario). Para inicios del 2017, la quinta versión Boron (septiembre,2016), da un enfoque práctico en la ingeniería de redes a gran escala (SDX CENTRAL , 2016).

Sus más recientes versiones Carbon (mayo, 2017) y Nitrogen (noviembre, 2017) ofrecen mejoras en las necesidades de IoT. En el Capítulo II se delimitan las características de selección del Controlador OpenDayLight para la realización del presente trabajo de titulación.

1.6. Redes Inalámbricas Definidas por Software (SDWN)

Las SDN son un paradigma de red en constante crecimiento, permiten a los administradores de red especificar el comportamiento de toda la red de forma centralizada y lógica, a través de la plataforma controladora. Con las mismas bases de SDN, las Redes Inalámbricas Definidas por Software (SDWN), tienen como objetivo proporcionar un control centralizado de puntos de acceso (AP), que cumplen las instrucciones recibidas y siguen siendo responsables del envío y recepción del tráfico en los enlaces inalámbricos (DUARTE, y otros, 2015).

SDWN es una rama de investigación emergente en el campo de las SDN, las cuales mejoran la flexibilidad, la capacidad de administración y la facilidad del uso de entornos de redes inalámbricas.

Cabe recordar, que el camino hacia el despliegue de las SDWN no está exento de retos en su implementación, hoy en día, su implementación es llevada a cabo para investigaciones

académicas dentro de campus universitarios en su mayoría. Para probar su funcionamiento y mejorarla (FONTES, y otros, 2017,pp. 5-10).

1.6.1. Factores de calidad de servicio

Hoy en día, las redes inalámbricas deben soportar las mismas aplicaciones que corren en las redes cableadas, así, dichas redes deben garantizar cierto nivel de calidad de servicio. La calidad de servicio garantiza el rendimiento extremo a extremo de los servicios electrónicos, tal como lo percibe el usuario final. Algunos parámetros de la calidad de servicio a considerar son:

1.6.1.1. Pérdida de paquetes

Se debe principalmente cuando los paquetes enviados no llegan a su destino, debido a la congestión en equipos o segmentos de red (MOSCOSO, 2016, pp. 28-30).

1.6.1.2. Retardo

Medido en unidades de tiempo, hace referencia al tiempo total transcurrido desde que el paquete origen se transmite a destino (WI-FI ALLIANCE, 2017).

1.6.1.3. Latencia

Hace referencia al tiempo que le toma a un nodo enviar un mensaje más el tiempo que le toma al otro nodo receptarlo, se incluye también el tiempo de todo el recorrido y los dispositivos intermedios (Análisis de la QoS en redes inalámbricas, 2013, pp. 86 -96).

1.7. Herramientas de Software

1.7.1. Mininet

Emulador de red que permite la creación de escenarios de redes virtuales. Se ejecuta en dispositivos finales, conmutadores, routers que posean Linux Kernel. Apoyado por la interfaz de línea de comandos (CLI) permitiendo crear, gestionar y compartir los escenarios creados (HENAO, 2015, pp- 9-60).

Su virtualización es ligera, funciona correctamente con topologías personalizadas y posee una API basada en Python incluida. Sus conmutadores soportan OpenFlow para un enrutamiento altamente flexible. Apoya la investigación, desarrollo, aprendizaje, pruebas, entre otras actividades para beneficiar una red experimental completa solamente en una portátil o PC, siendo de gran ayuda para los investigadores dentro del campo de las SDN (MININET TEAM, 2017) (DUARTE, y otros, 2015) .

Un host Mininet se comporta como una máquina real, ejecuta programas y envía paquetes en una interfaz Ethernet real emulada. Se reenvían paquetes por API´s de Mininet, programados para manejar el protocolo OpenFlow.

Algunas de sus características principales son: arranque más rápido, segundos en lugar de minutos; fácil conexión con redes reales; rendimiento interactivo; rápido y reconfigurable (CHICO, 2013, pp. 30-40). Una de sus limitaciones es que recurre a un solo núcleo Linux para todos los hosts virtuales, en el futuro podrá soportar otros sistemas operativos (ESPAÑA, 2016, pp. 8-10).

1.7.2. MiniEdit

Editor de interfaz gráfica de usuario (GUI) para Mininet. Herramienta experimental para extender la funcionalidad de Mininet. Es capaz de crear y ejecutar diferentes simulaciones de red, de igual manera, se configuran los diferentes elementos de red de manera simplificada y guarda las diferentes topologías (MARTÍNEZ, 2015, pp. 55-60).

MiniEdit tiene una interfaz de usuario sencilla presenta: un área de trabajo, herramientas de diseño; iconos de las herramientas tales como: equipos host, conmutadores Openflow habilitados, conmutador Ethernet de aprendizaje, router básico, netlink para crear enlaces entre los dispositivos de red y la herramienta fundamental un controlador. En la Figura 9-1 se observa la interfaz de MiniEdit.

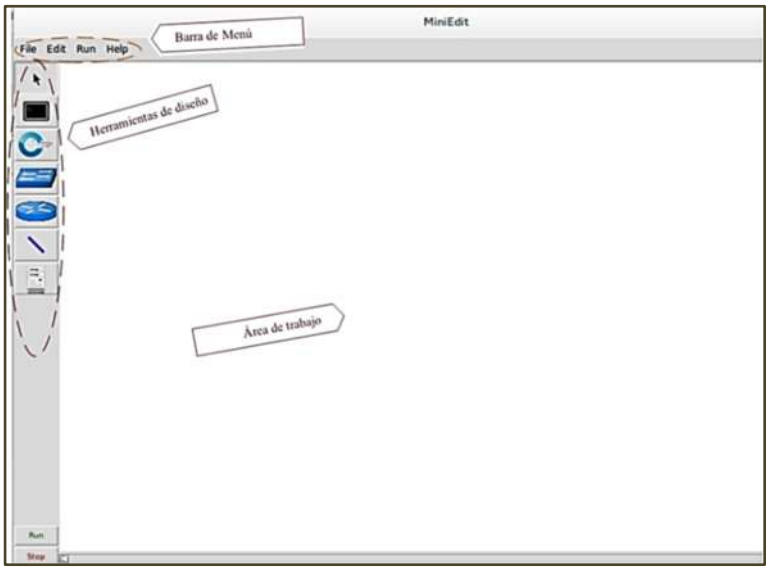


Figura 9-1 Interfaz de MiniEdit.

Realizado por. –CEVALLOS, L. 2017.

1.8. Mininet Wi-Fi

Herramienta del emulador de SDN Mininet; la cual extiende las funcionalidades propias de Mininet, añadiendo estaciones virtuales Wi-Fi (STAs) y puntos de acceso (APs) basados en los controladores inalámbricos estándar de Linux y el controlador de simulación inalámbrico mac80211/SoftMac, el cual soporta la mayoría de las características proporcionadas por las NIC (tarjetas inalámbricas). Los desarrolladores añaden clases para apoyar la incorporación de los dispositivos inalámbricos en diferentes escenarios de red y emular los atributos de una estación móvil como su posición y movimiento en relación a los puntos de acceso.

Mininet Wi-Fi ejecuta código real con tráfico real, soportando topologías arbitrarias y entornos dinámicos en tiempo real (FONTES, y otros, 2017, pp. 9-25).

1.8.1. Arquitectura y Componentes

Los principales componentes que forman parte de Mininet Wi-Fi se observan en la Figura 10-1. Donde Linux usa módulos para proporcionar la comunicación entre los dispositivos y el software; dinámicamente cargados por el usuario. Dichos módulos se encuentran tanto en el espacio de usuario como en el espacio del Kernel (Núcleo).

En el espacio del Kernel, el módulo *mac80211_hwsim* es un simulador responsable de crear interfaces virtuales de la conexión Wi-Fi, importantes tanto para las estaciones y los puntos de acceso. Se crean N número de interfaces cuando se inserta en el Kernel, donde N es especificado por el usuario. Las interfaces reciben el nombre de interfaces inalámbricas regulares *wlanX* (X es un número especificado por el usuario). También crea una interfaz maestra llamada hwsim0. Esta interfaz se utiliza para escuchar cualquier comunicación entre las interfaces inalámbricas simuladas (FONTES , y otros, 2015, pp 2-6).

Continuando en el espacio del Kernel, MLME (Entidad de gestión se subcapa MAC) se lleva a cabo en el lado de las estaciones, mientras en el espacio del usuario el hostapd es responsable de esta tarea en el lado del AP, representado por (ap1).

De igual manera, Mininet Wi-Fi dispone de algunas utilidades como son *iw* y *iwconfig* para la configuración de la interfaz y para obtener información de las interfaces inalámbricas. Conjuntamente *wpa_suplicant* y *Hostapd* se utilizan para soportar WPA (Wi-Fi Protected Access), entre otras cosas. Igual de importante el TC (Control Tráfico) en el espacio de usuario

es utilizado para configurar el programador de paquetes del Kernel de Linux, responsable de controlar la latencia, el retardo, entre otras.

Las estaciones (STA) son dispositivos conectados a un AP a través de autenticación y asociación. Cada estación tiene una tarjeta inalámbrica (staX-wlan0, X representa el número de cada STA). Los puntos de acceso son dispositivos que gestionan estaciones asociadas. Están virtualizados a través del daemon hostapd. Permite al usuario configurar características como: ssid, canal, contraseña, criptografía, entre otras.

Tanto las STA y los AP emplean *cfg80211* para la comunicación con el controlador de dispositivos inalámbrico, un API de configuración de Linux 802.11 que proporciona comunicación entre STAs y *mac80211*. Este framework se comunica directamente con el controlador del dispositivo Wi-Fi a través de un socket netlink (más específico *nl80211*) que se utiliza para configurar el dispositivo y para la comunicación entre el espacio del Kernel y el usuario (FONTES, y otros, 2017, pp. 9-25) (MARTÍNEZ , 2015, pp. 5-11.).

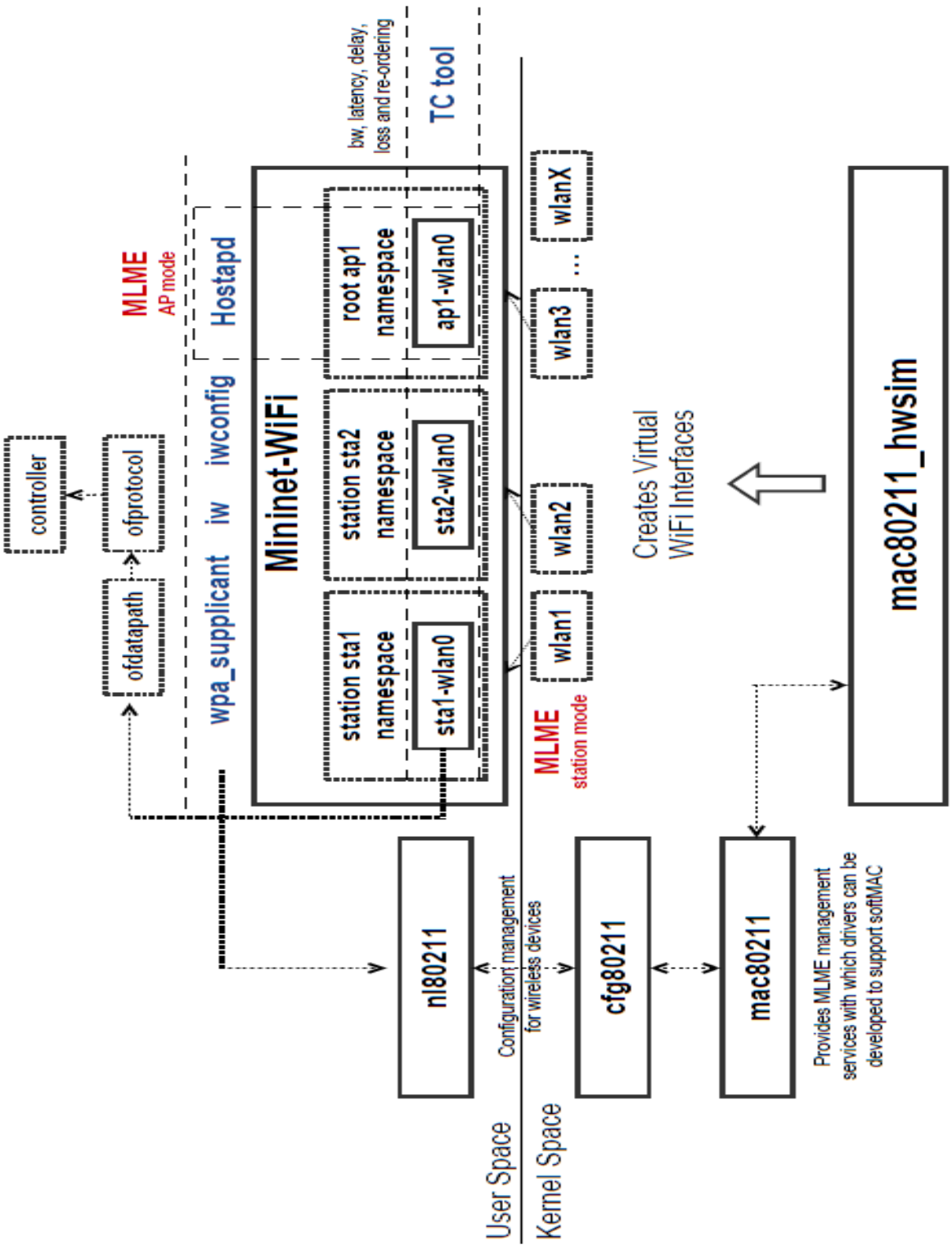


Figura 10-1 Componentes de Mininet Wi-Fi

Fuente.- (FONTES, y otros, 2017, pp. 9-25).

1.8.2. Movilidad

En el contexto de redes de datos se define a la movilidad como la capacidad de una red para acomodar hosts móviles de una parte de la red a otra. Por citar un ejemplo, un usuario conectado a la red desde su tablet moviéndose desde su oficina hasta la sala de reuniones ubicado en un edificio ajunto como se observa en la Figura 11-1, y aun así ser capaz de tener conexión a Internet; conectándose a la red a través de puntos de acceso Wi-Fi cercanos en toda su trayectoria (MARTÍNEZ , 2015, pp. 5-11.).

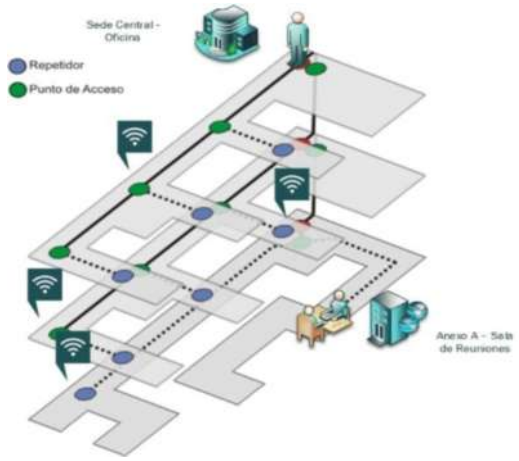


Figura 11-1 Movilidad

Realizado por. - CEVALLOS, L. 2017.

Para determinar patrones de movimiento se lleva a cabo la definición de modelos de movilidad, los cuales representan el movimiento de un nodo basado en su velocidad y su dirección; en algunos modelos también se considera la aceleración del nodo.

Los modelos de movilidad son desarrollados utilizando trazas prácticas y algunas teóricas que intentan ser realistas, para entender los patrones de movimiento. Sin embargo, dichas trazas son limitadas, dado que la mayoría solo son aplicables en escenarios específicos. Por ejemplo, la traza obtenida en un campus universitario no puede ser aplicable en un escenario como una ciudad donde encontraremos un mayor número de puntos de acceso (DUARTE, y otros, 2015).

Hoy en día, el esfuerzo por obtener trazas con mayor precisión en diversos ambientes va en constante aumento, como resultado, se desarrollan modelos que han demostrado ser una buena aproximación de la caminata humana en ambientes de red al aire libre.

Dentro de las SDWN, Mininet Wi-Fi soporta modelos de movilidad para diferentes situaciones, utilizando *net.StarMobility.* Los modelos de movilidad varían diferentes parámetros y así logran movimientos aleatorios, utilizando los métodos: *RandomWalk, TruncatedLevyWalk, RandomDirection, RandomWayPoint, GaussMarkov, ReferencePoint, TimeVariantCommunity*.

Los cuales pueden estar clasificados según se observa en las Figura 12-1 (FONTES, y otros, 2017, pp. 9-25).

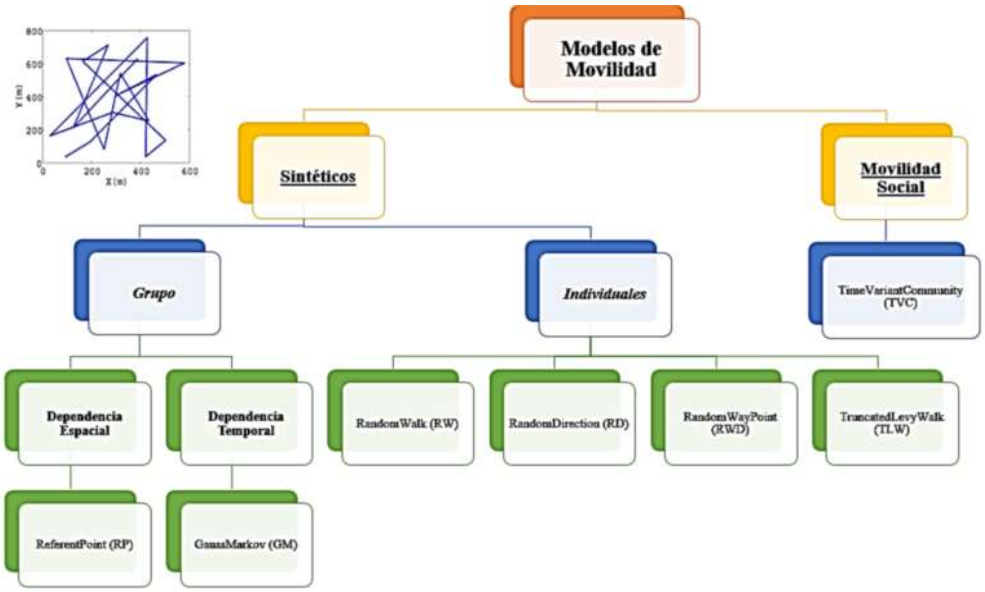


Figura 12-1 Clasificación de los modelos de movilidad en Mininet Wi-Fi.

Realizado por. - CEVALLOS, L. 2017.

Los modelos sintéticos son modelos analíticos desarrollados a partir de un análisis analítico de escenarios específicos. Se modela el movimiento al relacionarlo con patrones de comportamiento individuales (modelos individuales) y con los patrones de movimiento grupal (modelos de grupo). (FONTES, y otros, 2017, pp. 9-25)

Los modelos sintéticos individuales son:

- *RandomWalk.* – se basa en la selección aleatoria de dirección y velocidad, cada nodo se mueve hacia una nueva posición aleatoria. Cuando un nodo llega a su destino, se le asigna una nueva dirección desde rangos predefinidos.
- *RandomDirection.* – los nodos son capaces de llegar al borde de la simulación; cuando un nodo alcanza el límite del área simulada, se detiene durante un periodo y elige una dirección nueva al aplicar un ángulo entre 0° y 180° y reanuda su movimiento.
- *RandomWayPoint.* – considera un tiempo de pausa aleatorio entre los cambios de dirección y velocidad
- *TruncatedLevyWalk.* – se basa en la captura del movimiento en diversos escenarios al aire libre.

En los modelos grupales, la dependencia grupal se refiere a que el movimiento del nodo estará influenciado por los nodos que los rodean; la dependencia temporal hará referencia a que el movimiento real del nodo está influenciado por el movimiento anterior. (Reference Point Group Mobility and Random Waypoint Models in Performance Evaluation of MANET Routing Protocols, 2008, pp. 1-8)

En los modelos de grupo, tendremos:

- *Dependencia Espacial- ReferentPoint.*- simula el comportamiento de un grupo, cada nodo pertenece a un grupo, dicho nodo seguirá un centro lógico (líder grupo) quien dictamina el comportamiento del movimiento del grupo.

- *Dependencia Temporal – GaussMarkov.*- se basa en un modelo gaussiano por lo que elimina, paradas repentinas y giros bruscos experimentales. Usa un parámetro de sintonía para variar el grado de aleatoriedad en el patrón de movilidad

En muchas aplicaciones, los nodos inalámbricos son transportados por humanos, animales o vehículos donde cada uno posee un patrón de movimiento propio. Las investigaciones actuales han desarrollado modelos basados en la movilidad humana, para de cierta manera, tener un patrón aproximado. Así, todos estos patrones son agrupados como modelos de Movilidad Social.

El modelo de Movilidad Social en Mininet Wi-fi es:

- *TimeVariantCommunity.*- se basa en las comunidades (área que un usuario visita con alta probabilidad), las cuales se utilizan para modelar las preferencias de visitas y los periodos de tiempo en la misma ubicación. (Mobility Models for Delay Tolerant Network: A Survey, 2014, pp. 121-130)

1.9. Python

Lenguaje de programación creado por Guido van Rossum a principios de los años 90. Es un lenguaje similar a Perl, C y Java; pero con una sintaxis simple y un código legible. Se trata de un lenguaje interpretado o de script, de alto nivel, orientado objetos, multiplataforma, con tipado dinámico y fuertemente tipado (GONZÁLEZ, 2010, pp. 7-12).

Lenguaje de sintaxis simple, clara y sencilla; el tipado dinámico, la gran cantidad de librerías disponibles y la potencia del lenguaje, hacen que desarrollar una aplicación en Python sea rápido y sumamente fácil. Además de ser uno de los lenguajes para comenzar a programar (CHRISTENSSON, 2010).

En Python, el código fuente se traduce a un pseudo código de maquina intermedio llamado bytecode la primera vez que se ejecuta, guardados como programas compilados *.pyc o .pyo* (bytecode optimizado), logrando ser utilizados como módulos de programación referenciados por otros programas de Python.

Para ejecutar código Python existen dos formas. En la primera se escriben líneas de código en el intérprete y se obtiene una respuesta del mismo para cada línea, o bien la segunda en donde se escribe el código de un programa en un archivo de texto y se ejecuta. Es ejecutable en sistemas operativos: Windows, Mac, Unix y de igual manera en usada para .NET y Mininet. (PYTHON TUTORIALS, 2017)

1.9.1. Elementos del Lenguaje

Al igual que en otros lenguajes de programación, Python consta de una serie de elementos que alimenta su estructura. Los cuales se describen en la Tabla 3-1, los cuales ayudaran en la programación de scripts los cuales ayudaran para la creación de los diversos escenarios de prueba.

Tabla 3-1 Elementos del lenguaje Python

Elementos	***Descripción***	***Sintaxis***
Variables	Es un espacio para almacenar datos modificables, en la memoria de un ordenador.	nombre_variable = valor_variable
Comentarios	Notas en el código, para comprenderlo mejor. Pueden ser: de una sola línea o multi-línea.	• Solo una línea: # comentario • Multi-línea ”””” Comentario De varias líneas ””””
Listas	Es una variable que permite almacenar varios datos, los cuales pueden ser modificar sus datos una vez creados.	mi_lista= (‘ cadena de texto’, 20 ,’otro dato’, 25)

Realizado por. - CEVALLOS, L. 2017
Fuente.- (BAHIT, 2013, pp. 10 - 40)

En Python cada uno de los archivos .py son denominados módulos. Cada módulo, a la vez, forma parte de paquetes. Un paquete, es una carpeta la cual puede contener diversos archivos .py; para que una carpeta pueda ser considera un paquete debe contener un archivo de inicio llamado __init__.py, el cual no requiere contener ninguna instrucción (GONZÁLEZ, 2010, pp. 7-12).

Los módulos pueden ser importados y utilizados por otros módulos. Para importar un módulos se lleva a cabo la instrucción *import*, seguida del nombre del paquete más el nombre del módulo,

ejemplo: *import paquete.modulo1*; dicha sentencia permite hacer uso del código contenido en el módulo.

Para acceder a cualquier elemento del módulo importado, se realizada mediante el *namespace*, seguido de un punto (.) y el nombre del elemento deseado. Un namespace, es el nombre indicado después de la palabra import. Al momento de importar algunos elementos de un módulo, se utiliza la instrucción *from* seguida del *namespace* más la instrucción *import* seguida del elemento deseado, Figura 13-1 (BAHIT, 2013, pp. 10 - 40) (GONZÁLEZ, 2010, pp. 7-12).

```
#!/usr/bin/python

from mininet.net import Mininet
from mininet.node import Controller,OVSKernelSwitch
from mininet.link import TCLink
from mininet.cli import CLI
from mininet.log import setLogLevel

def topology():

    net = Mininet( controller=Controller, link=TCLink, switch=OVSKernelSwitch )

    print "*** Creating nodes"
    ap1 = net.addBaseStation( 'ap1', ssid= 'ssid-ap1', mode= 'g', channel= '1', position='10,30,0', range='20' )
    ap2 = net.addBaseStation( 'ap2', ssid= 'ssid-ap2', mode= 'g', channel= '6', position='50,30,0', range='20' )
    sta1 = net.addStation( 'sta1', mac='00:00:00:00:00:01', ip='10.0.0.1/8', position='10,20,0' )
    sta2 = net.addStation( 'sta2', mac='00:00:00:00:00:02', ip='10.0.0.2/8', position='50,20,0' )
    c1 = net.addController( 'c1', controller=Controller )
```

Figura 13-1 Ejemplo de código Python

Realizado por. - CEVALLOS, L. 2017.

CAPITULO II

2. MARCO METODOLÓGICO

2.1. Métodos

El presente trabajo de titulación hace referencia a los métodos:

- *Experimental*

Implementado al realizar pruebas de rendimiento y conectividad de la red en los escenarios planteados y obtener resultados que se analizan a través de métricas de red.

- *Deductivo*

Las herramientas de software que fueron empleadas en la implementación de las topologías en Mininet Wi-Fi y el controlador OpenDayLight y su posterior análisis permitieron obtener un nuevo aporte a las futuras investigaciones dentro del campo de las SDWN.

2.2. Técnicas

- *Observación*

Se lleva a cabo un registro del comportamiento de la red en cada uno de los escenarios planteados, donde podemos medir el rendimiento a través de *iperf3*; para obtener parámetros como retardos, perdida de paquetes, ancho de banda en los enlaces.

2.3. Instrumentos de investigación y recolección de datos

Se logró obtener información de vital importancia a través de la investigación bibliográfica y documental de: libros, revistas científicas, papers, publicaciones en líneas y videos online; lo cual permitió una correcta compresión del tema, ayudando al mejor perfeccionamiento de la presente investigación.

2.4. Población y muestra

La población de estudio son los administradores de redes que utilizan las tecnologías actuales, y la muestra corresponde a los escenarios implementados a través de Mininet Wi- Fi. Se desarrolló dos escenarios. El primero consta de 5 puntos de acceso, 8 estaciones fijas, 3 hosts y un controlador remoto. En el segundo se delimitan 6 puntos de acceso, 12 estaciones móviles y un controlador remoto. Se limita el número de estaciones fijas para llevar a cabo una mejor visualización de los movimientos en la cuadricula de simulación.

2.5. Desarrollo del proyecto

En la Figura 1-2, se describe el procedimiento a seguir para el desarrollo de la investigación

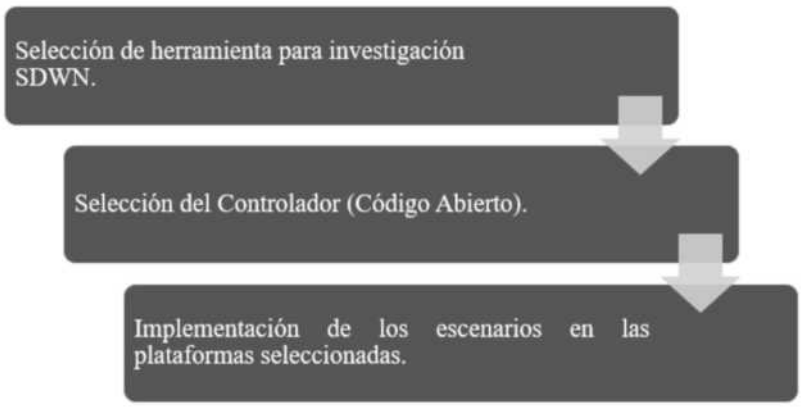


Figura 1-2 Proceso para el desarrollo del proyecto.

Realizado por. - CEVALLOS, L. 2017.

Se inicia, con el estudio de las herramientas más populares para poder realizar la virtualización de Redes Inalámbricas Definidas por Software (SDWN). Así, antes de detallar la ruta de investigación propuesta basada en el emulador Mininet Wi-Fi, vamos a examinar plataformas de investigación disponibles: simuladores, emuladores y testbest. Para la selección del controlador, eje central de las SDN se llevó a cabo la comparación de diversos controladores de código abierto en cuanto a sus características. Una vez seleccionados tanto el Controlador como la plataforma de simulación, se efectuó la implementación de escenarios tanto para estaciones móviles como fijas; demostrando la movilidad de estas en tiempo real. El análisis de tráfico en diferentes interfaces ayudara a la mejor comprensión de las SDWN en un entorno virtualizado

2.6. Selección de herramienta para investigación SDWN

2.6.1. Comparación de plataformas

Para la selección de la plataforma adecuada, se llevó a cabo una comparación de simuladores, emuladores y testbeds los cuales apoyan la investigación en el campo de las SDW. Se establecen sus características principales para mejorar la experimentación realista de las SDWN. La cuantificación exacta de cada característica dependerá del modelo implementado en cada herramienta, la Tabla 1-2 ilustra las principales fortalezas y limitaciones como una guía inicial para elegir la mejor herramienta.

Tabla 1-2 Ranking de Simuladores, emuladores y testbeds

Característica	***Simuladores***	***Emuladores***	***Testbeds***
Fidelidad	1	2	3
Aplicaciones Reales	1	3	3
Escalabilidad	3	2	1
Realismo del trafico	1	3	3
Realismo en el tiempo de ejecución	3	2	3
Sustentabilidad	3	3	1
Flexibilidad	3	3	1
Total	**15**	**18**	**15**

Realizado por: Cevallos, L. 2018.
Fuente. - (FONTES, y otros, 2017,pp. 5-10).

(Importancia: 1: Baja, 2: Media, 3: Alta)

La mejor herramienta para llevar a cabo la investigación son los emuladores los cuales proveen de flexibilidad y la posibilidad de experimentar con tráfico de red más realista. Se complementa la selección de Mininet Wi-Fi, con simuladores y emuladores conocidos como DCE/ns -3 y OMNet ++, y, de igual manera con los testbeds EMULAB Y R2lab t; como se indican en las Tablas 2-2 y 3-2.

Tabla 2-2 Comparación de Mininet Wi-Fi con emulador y simulador para Wireless.

Software	*Tipo*	*Tipo de Fuente*	*Lenguaje de programación*	*Protocolos soportados*
Mininet Wi-Fi	Emulador	Open	Python	IEEE 802.11, 802.3, OpenFlow
DCE/ns-3	Emulador	Open	C++, Python	IEEE 803.11, LTE, OpenFlow
OMNet ++	Simulador	Open	C++	IEEE 803.11, OpenFlow

Fuente. - (FONTES, y otros, 2017,pp. 5-10).

Tabla 3-2 Comparación en Mininet Wi-Fi y testbeds.

Software	*Disponibilidad*	*Soporte de Movilidad*	*Protocolos soportados*
Mininet Wi-Fi	Público	Si	IEEE 802.11, 802.3, OpenFlow
EMULAB	Público	No	IEEE 802.11,
R2lab	Público	No	IEEE 802.11, LTE

Fuente. - (FONTES, y otros, 2017,pp. 5-10).

Tanto el simulador OMNet ++ como el emulador DCE/ns-3 presenta algunas limitaciones. OMNet ++ tiene su propia implementación de switch/AP y no soporta controladores de terceros. Por su parte, DCE/ns -3 no admite la ejecución de aplicaciones no modificadas o sin modificación del kernel.

Al abordar los vacíos en las diversas herramientas de experimentación SDWN, se selecciona el emulador Mininet Wi-Fi como la mejor alternativa para la presente investigación, ya que permite la ejecución de código real sin modificación en kernel o aplicaciones, soporta las tecnologías de código abierto (controladores de terceros). Sin dejar el aporte con los modelos de movilidad, modelos de propagación de canales inalámbricos, la capacidad de integrarse con dispositivos reales, características favorables para el desarrollo del proyecto. Para detalles de instalación de Mininet Wi-Fi ver Anexo A.

2.6.2. Selección del Controlador (Código Abierto)

2.6.2.1. Comparación de controladores

En el mercado existen varios controladores, desarrollados por comunidades y otros por empresas reconocidas a nivel mundial. La diferencia principal menciona anteriormente radica en el lenguaje

de programación que utilizan, pero en si ejecutan las mismas funciones. Entre algunos controladores citados continuamente se mencionan: NOX, POX, OpenDayLight (ODL), Floodlight, Beacon, entre otros.

Características de selección para Controladores OpenFlow

La selección de un controlador es vital, ya que este es el eje central para todo el diseño de las SWDN. Presentado una serie de características al momento de la selección del controlador indicado para el diseño de la red. Estas son:

- *Soporte de Openflow.* - Versiones de OpenFlow que el controlador soporta, y su posibilidad de migrar a nuevas versiones.
- *Virtualización.* - Permite a los administradores de la red crear dinámicamente las redes virtuales basados en políticas, para satisfacer una amplia gama de requerimientos.
- *Funcionalidad de la red.* - El controlador tomara decisiones de enrutamiento basados en múltiples campos de la cabecera OpenFlow, para descubrir multiples caminos desde el origen del flujo hasta su destino.
- *Provee REST API.* - Expone los servicios del controlador a las aplicaciones de gestión.
- *Escalabilidad.* - Se toma en consideración el número de conmutadores que un controlador puede soportar. Y su capacidad de asegurar el impacto de sobrecarga de difusión de red, reduciendo al mínimo la proliferación de las entradas de la tabla de flujo.
- *Rendimiento.* - Referente al tiempo de conformación de flujo y el número de flujos por segundo que puede establecer el controlador.
- *Programación de la red.* - Al contar con interfaces para la programación del controlador, proporciona a estas varias funcionalidades.
- *Soporte de plataforma.* - Los controladores corren sobre sistemas operativos, por consecuente este debe ser multiplataforma permitiendo una mayor flexibilidad e independencia al momento de su implementación.
- *Procesamiento.* - Conocer si puede soportar procesos multiples o no, pues esto repercutirá en la escalabilidad de los núcleos de la CPU. (CENTENO, y otros, 2014, pp 3-5).

Tabla 4-2 Comparativa entre Controladores OpenFlow.

	Características							
Controladores	*NOX*	*CALIF.*	*POX*	*CALIF.*	*OPENDAYLIGHT*	*CALIF.*	*FLOODLIGHT*	*CALIF.*
Soporte OpenFlow (versión)	OF v 1.0	1	OF v 1.0	1	OF v 1.0	1	OF v 1.0	1
Virtualización	Mininet Open Vswitch	2	Mininet Open Vswitch	2	Mininet Open Vswitch	2	Mininet Open Vswitch	2
Funcionalidad de la red	Nivel bajo	1	Nivel bajo	1	Nivel alto	3	Nivel medio	2
Provee REST API	No	0	No	0	Si	3	Si	3
Escalabilidad	Baja	1	Media	2	Alta	3	Media	2
Rendimiento	Nivel Bajo	1	Nivel Bajo		Nivel alto	3	Nivel medio	2
Programación de la red	Python +	1	Python +	1	Web	3	Web	3
Plataformas (S.O)	Linux	1	Windows Linux Mac OS	3	Windows Linux Mac OS	3	Windows Linux Mac OS	3
Multiprocesos	Si	3	No	1	Si	3	Si	*3*
TOTAL:		11		11		24		*21*

Realizado por.- Cevallos, L.2018

Fuente.- (ALBÁN, y otros, 2015, p. 21) (SDX CENTRAL , 2016)

(**Importancia**: 1:Bajo; 2:Medio; 3:Alto)

Por los resultados adquiridos en la Tabla 4-2, se ha resuelto implementar el Controlador OpenDayLight como eje vital para la virtualización de escenarios SDWN. Para detalles de su instalación ver Anexo B. Se han delimitado varias opciones de controladores los más utilizados en investigaciones en el país como Nox, Pox o Floodlight, y tomando en cuenta el constante cambio del controlador ODL, se ha apostado por él, especialmente por tener una alta funcionalidad de la red.

2.6.3. Implementación de los escenarios en las plataformas seleccionadas

Con el fin de simular las SDWN es necesario como se mencionó anteriormente dos elementos básicos, un emulador de redes y un controlador. En el presente proyecto se decidió el implementar el software de emulación Mininet Wi-Fi y el controlador OpenDayLight.

El proyecto se desarrolló con dos computadores con sistema operativo Windows 10, usando una máquina virtual que contiene Linux, y que a su vez tiene instalado Mininet Wi-Fi. La máquina virtual se ha cargado en VirtualBox. Para más detalles revisar ver Anexo C.

Mininet Wi-Fi trabaja con los estándares IEEE 802.11 a/b/g y n delimitando su cobertura y su velocidad de transmisión máxima con movilidad y sin movilidad como se detalla en la Tabla 5-2. Los valores mencionados en la Tabla 5-2 corresponden cuando no se especifica el equipo a utilizar en la simulación. Refiriendo a que Mininet Wi-Fi dispone de 3 tipos de puntos de acceso D-Link AirPlus DI-524, TL WR740 y WRT 120N de Cisco. Se seleccionó el punto de acceso WRT 120N para llevar a cabo la programación de la topología en scripts de Python.

Tabla 5-2 Configuración de Mininet Wi-Fi para estándares IEEE 802.11

Estándar	**Rango en interiores (m)**	**Velocidad de transmisión máxima (Mbps)**	
		Con movilidad	*Sin movilidad*
802.11a	33	11	20
802.11b	50	3	6
802.11g	33	11	20
802.11n	70	600	48

Realizado por. - Cevallos, L.2018

Fuente.- (FONTES, y otros, 2017, pp. 9-25)

2.6.3.1. Creación de topologías en Mininet Wi-Fi

La topología es la forma física o virtual con la que está diseñada una red; la interconexión de cada uno de los dispositivos que la conforman como: conmutadores, puntos de acceso, estaciones o nodos entre otros.

En Mininet Wi-Fi, se permiten al igual que en Mininet 4 modelos de topologías mencionados en la Tabla 6-2. Así, para llevar a cabo todas las topologías se parte del comando: **$ sudo mn –wifi**, el cual por defecto crea un punto de acceso con dos estaciones conectadas al mismo.

Tabla 6-2 Topologías de Mininet Wi-Fi

Topología	Comando	Descripción
Single	$ sudo mn --wifi --topo single, N	N: indica el número de STAs conectadas a un único AP
Lineal	$ sudo mn --wifi --topo lineal, N	N: indica el número de APs, y cada AP tiene una única STA.
Tree	$ sudo mn –wifi --topo single, N,M	N: Niveles. M: número de STAs conectadas a cada AP.
Custom (Personalizada)	Se puede implementar topologías tan complejas como se desee, creando scripts en lenguaje Python	

Realizado por. - Cevallos, Liseth.2018.

El investigador selecciona el lenguaje Python, adecuado para administrar redes y con gran acogida al momento de programar diversas aplicaciones de red, para el desarrollo de las simulaciones del trabajo de titulación

Para el presente trabajo de titulación se realiza una topología personalizada (custom), en la cual, se realizaron los escenarios tanto con estaciones fijas, como también la implementación de movilidad en dichas estaciones. Se crea una carpeta en el directorio **~/mininet-wifi/** el cual contendrá scripts Python para cada escenario.

2.6.3.2. Escenarios de simulación

Los escenarios de simulación a emplearse se muestran en las Figuras 2-2 y 2-3. Existen 2 posibles escenarios:

Escenario 1: Topología personalizada en scripts Python con estaciones fijas inalámbricas usando un controlador remoto.

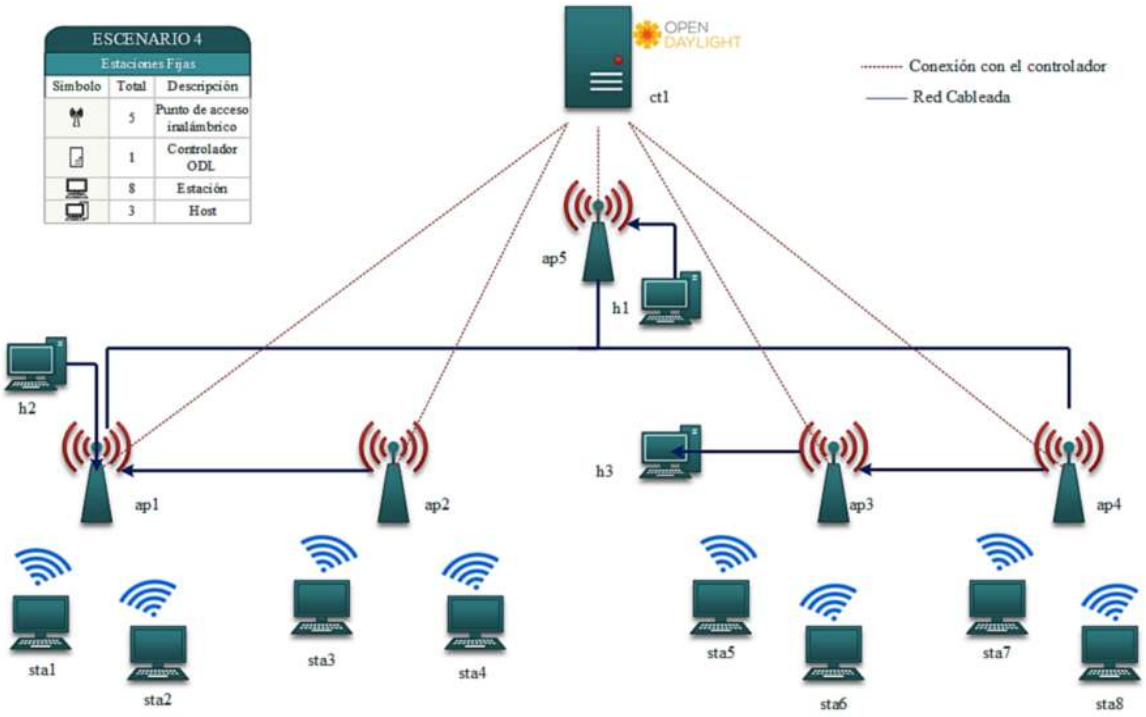


Figura 2-2 Topología Escenario 1.

Realizado por. - CEVALLOS, L. 2018.

Tabla 7-2 Direcciones IP y canales inalámbricos. Escenario 1

Estación	Dirección IP	Punto de Acceso	Canal
sta1	192.168.0.11/24	ap1	1
sta2	192.168.0.2/24	ap2	6
sta3	192.168.0.3/24	ap3	11
sta4	192.168.0.4/24	ap4	1
sta5	192.168.0.5/24	ap5	6
sta6	192.168.0.6/24		
sta7	192.168.0.7/24	**Hosts**	**Dirección IP**
sta8	192.168.0.8/24	h1	10.0.0.2/8
Controlador		h2	10.0.0.3/8
ct1	192.168.1.2/24	h3	10.0.0.4/8

Realizado por. - Cevallos, Liseth.2018.

Escenario 2: Topología personalizada en scripts Python con estaciones móviles inalámbricas usando un controlador remoto y modelos de movilidad.

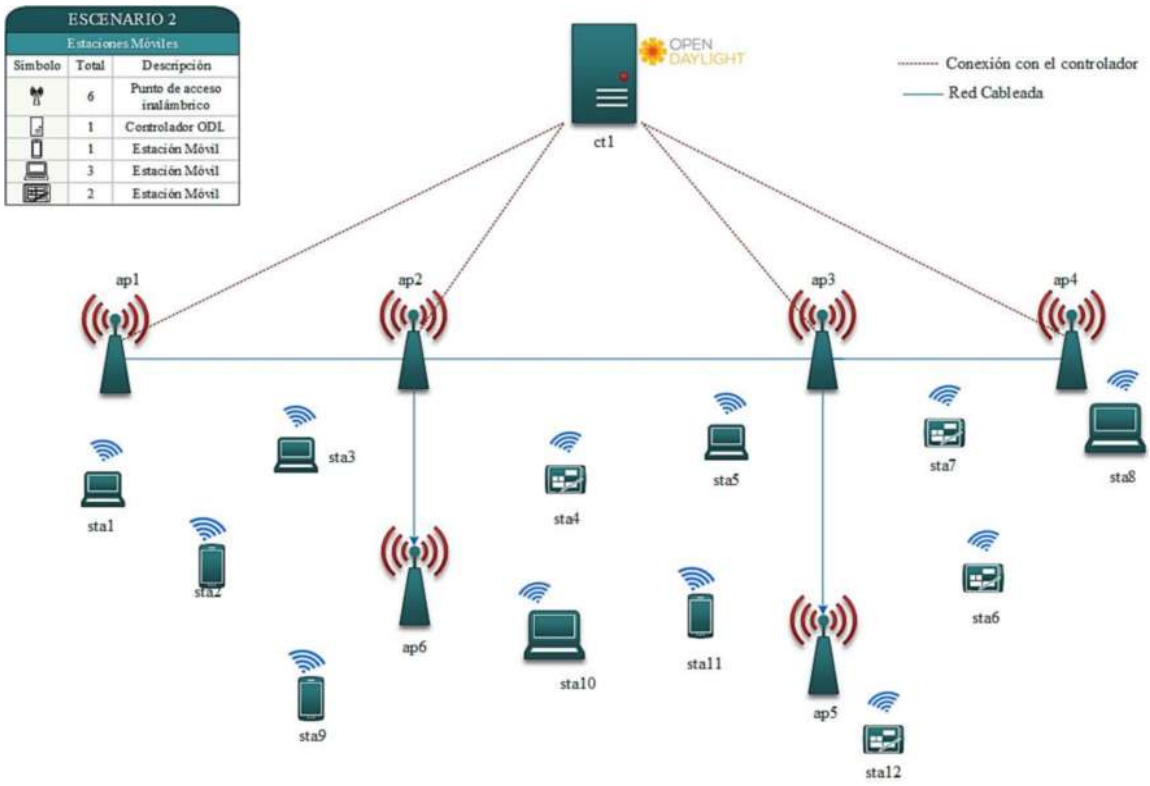


Figura 3-2 Topología Escenario 2

Realizado por. - CEVALLOS, L. 2018.

Tabla 8-2 Direcciones IP y canales inalámbricos. Escenario 2.

Estación	Dirección IP	Punto de Acceso	Canal
sta1	192.168.0.11/24	ap1	1
sta2	192.168.0.2/24	ap2	6
sta3	192.168.0.3/24	ap3	1
sta4	192.168.0.4/24	ap4	11
sta5	192.168.0.5/24	ap5	6
sta6	192.168.0.7/24	ap11	11
sta7	192.168.0.8/24		
sta8	192.168.0.9/24	**Controlador**	**Dirección IP**
sta9	192.168.0.10/24	ct1	192.168.1.2/24
sta10	192.168.0.12/24		
sta11	192.168.0.13/24		
sta12	192.168.0.14/24		

Realizado por. - CEVALLOS, L. 2018.

Para recrear los escenarios mencionados se establece el número de puerto 6633 para la comunicación entre controlador y los dispositivos de red, y el direccionamiento IP se determina en las Tablas 7-2 y 8-2 para cada escenario.

2.6.3.3. Código en scripts Python

Para el desarrollo de los scripts de Python para su posterior implementación en Mininet Wi-Fi, se podría seguir el siguiente proceso como se pauta en la Figura 2-2. Para mayor detalle Anexo D.

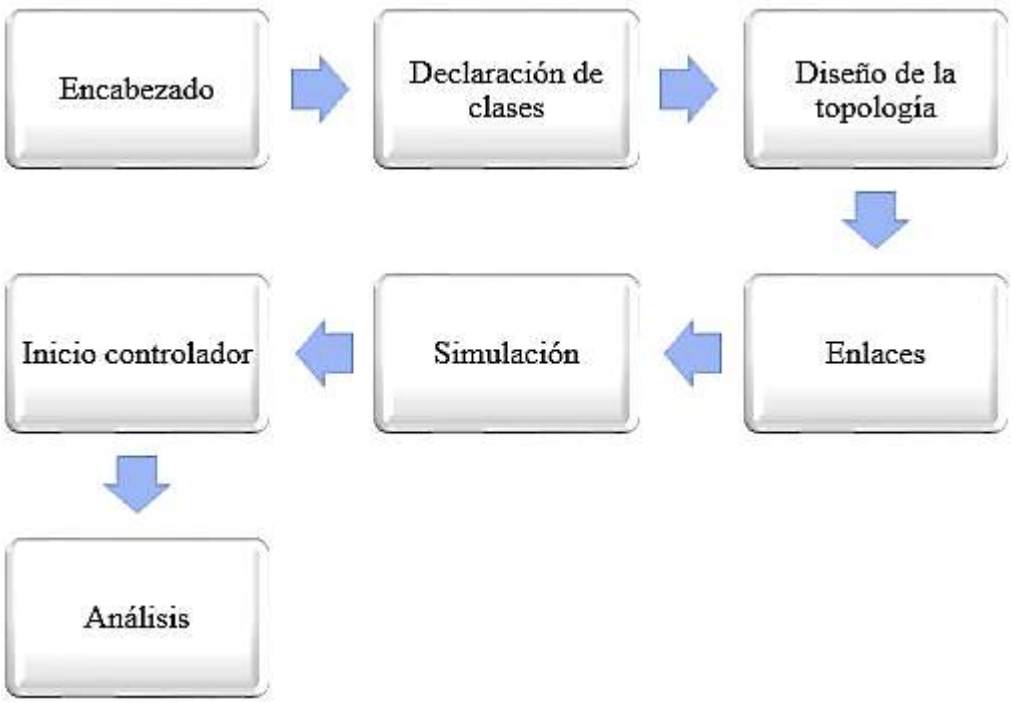


Figura 4-2 Diagrama de Flujo para el diseño de scripts Python.

Realizado por.- Cevallos, L.2018

El código completo de los escenarios se encuentra en los Anexos E y F, respectivamente.

2.6.3.4. Comandos CLI Mininet-Wifi

Mininet Wi-Fi nos permite la ejecución de comandos a través de la interfaz de línea de comandos, los cuales son de ayuda para obtener de forma resumida mayor información de la topología implementada. Como se muestra en la Tabla 9-2.

Para ejecutar comandos con permisos de root, se antepone al comando **sudo**. Una vez se cierre Mininet Wi-Fi se recomienda limpiar la topología anterior para evitar complicaciones.

Tabla 9-2 Comandos para Mininet Wi-Fi.

$comando en Shell #comando como root > comando dentro de Mininet Wi-fi	
Comando	**Descripción**
>net	Como están conectados los dispositivos.
>nodes	Muestra nodos de la topología.
>dump	Información de todos los nodos.
>pingall	Conectividad entre todos los nodos
>EOF	Detiene los dispositivos de la red y la cierra
> [node] ifconfig	Muestra información de las interfaces de nodo indicado
>[node1] ping [node1]	Envía un ping desde en node1 al node2.
>links	Enlaces de toda la red
>xterm [node]	Abre un terminar del nodo especificado
>position [node]	Posición del nodo indicado
>distance [node1] [node2]	Distancia entre dos nodos
$sudo wireshark &	Inicia el analizador de trafico Wireshark
$sudo mn –c	Limpia los residuos de redes creadas.

Realizado por.- Cevallos, L.2018.

En la presente investigación se utilizan topologías personalizadas por lo que al momento de programar las mismas se creara una carpeta en la carpeta de Mininet Wi-Fi, para poder ingresar de manera más rápida a la misma. Para ingresar a los scripts de Python se ingresa el comando **$sudo python nombre_archivo.py** para ejecutar el código del script. De igual manera, Mininet Wi-Fi también permite ejecutar comandos al momento de la ejecución de escenarios de movilidad. Dichos comandos son presentados en la Tabla 10-2.

Tabla 10-2 Cambios durante la ejecución.

Comando	**Descripción**
>py [sta].setPosition("x,y,z")	Ajustar la posición
>py [sta].setRange (100)	Ajustar el rango de la señal (m)
>py [sta].setAtennaGain('sta1-wlan0',5)	Ajuste en la ganancia de la antena (dBm)

Realizado por.- Cevallos, L.2018.

CAPÍTULO III

3. MARCO DE RESULTADOS, DISCUSIÓN Y ANÁLISIS DE RESULTADOS

3.1. Simulación y análisis

Para el análisis de las SDWN, se llevaron a cabo pruebas basándose en diferentes escenarios con puntos de acceso, estaciones móviles y fijas; para la visualización de la movilidad de estaciones inalámbricas el número de las mismas será limitado para tener una mejor visualización de su movilidad en la cuadricula de simulación.

A través del comando *pingall* en la CLI de Mininet Wi-Fi, se llevaron a cabo pruebas de conectividad, para la revisión de que las conexiones sean exitosas en los escenarios implementados.

En herramienta Wireshark, se captura y analiza el tráfico inalámbrico que pasa desde un punto de acceso hasta una estación inalámbrica, donde será habilitada la interfaz *hwsim0;* dicha interfaz reproduce las comunicaciones enviadas a las interfaces inalámbricas simuladas del punto de acceso, se habilita la interfaz con el comando *>sh ifconfig hwsim0 up*. Para capturar los paquetes de OpenFlow se deben capturar en la interfaz loopback *lo*, en el caso de utilizar el controlador por defecto de Mininet Wi-Fi. Pero al contar con el controlador remoto OpenDayLight los paquetes de OpenFlow se capturan desde la interfaz *enp0s9.*

De igual manera se utiliza la herramienta iperf3, la cual permite llevar a cabo pruebas en redes informáticas; crea flujos de datos TCP y UDP para medir el rendimiento de la red. En cada escenario se creará un servidor y un cliente iperf3, y se establece un intervalo de 60 sec para que el servidor se mantenga oyente al cliente.

3.1.1. Topología personalizada en scripts Python con estaciones fijas inalámbricas usando un controlador remoto.

Para iniciar la topología se requiere de la realización del escenario, para lo cual se utilizada los scripts de Python, la topología y direccionamiento IP fueron mencionados en el capítulo II. Una

vez dispuesto el script, se llevó a cabo la comprobación del mismo el cual se podrá visualizar tanto en Mininet Wi-Fi, como en la interfaz web del controlador. Para la visualización en la interfaz del controlador se enviará un ping a todas las estaciones desde la consola de comandos de Mininet Wi-Fi, la misma será llevada a cabo para que el controlador identifique a todos los elementos de la red. En la Figura 1-3 se observa el escenario en Mininet Wi-Fi, como en la interfaz del controlador.

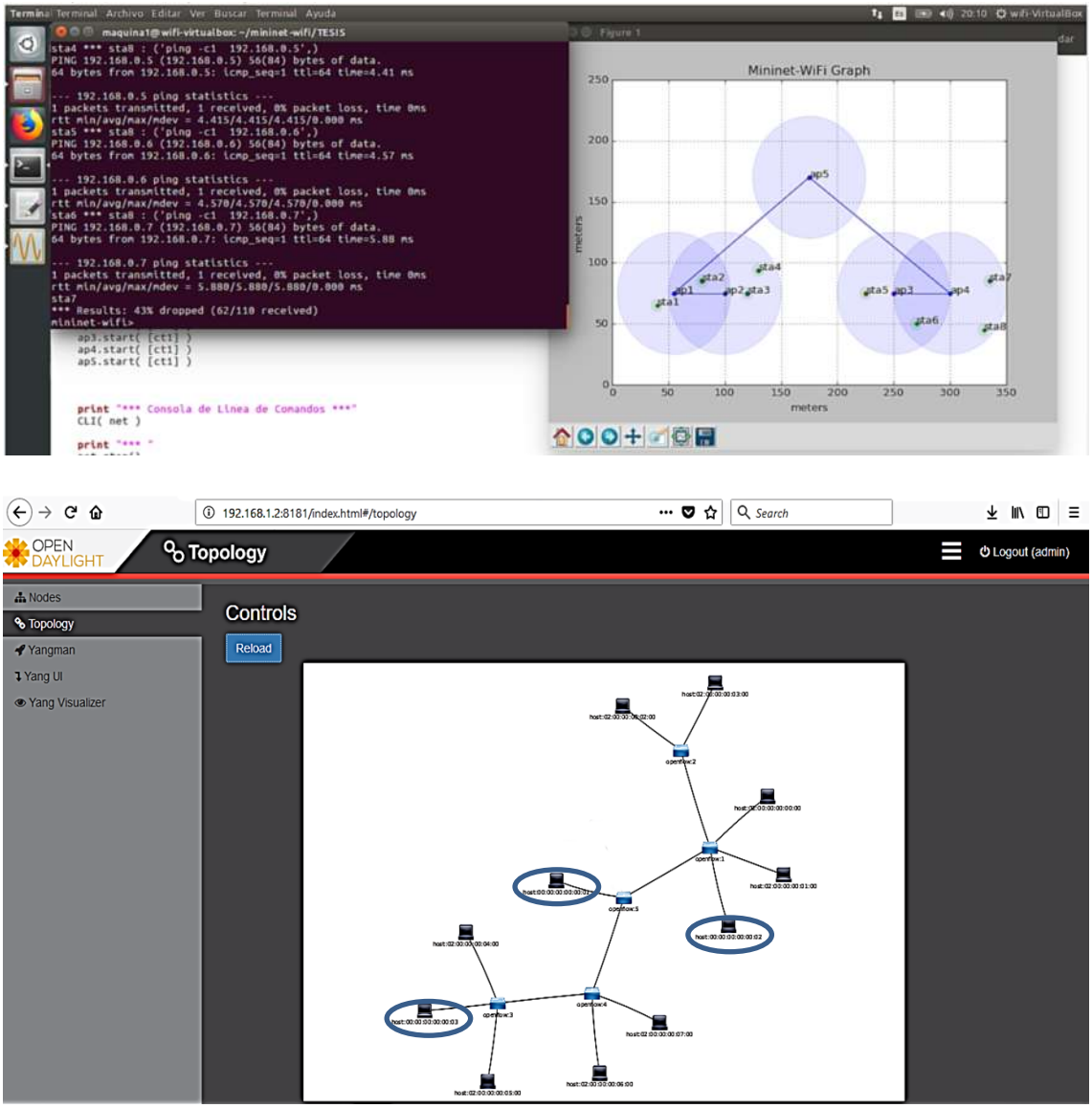


Figura 1-3 Visualización mediante Mininet Wi-Fi y OpenDayLight. Escenario 1.

Realizado por.- Cevallos, L.2018.

Cabe mencionar que en Mininet Wi-Fi solo se visualizan los puntos de acceso y las estaciones fijas en este caso, en la interfaz de ODL también se obtienen los hosts conectados en los ap5, ap1y ap3 como se muestra en la Figura 1-3, los hosts serán los marcados con azul.

Desde la pestaña 'Nodes' de la interfaz del controlador se puede ver información de cada AP, se obtendrá una tabla con todos los APs del escenario, al hacer click en 'Node Connector', Figura 2-3 se obtendrá información de cada puerto del AP.

Node Connector Id	Rx Pkts	Tx Pkts	Rx Bytes	Tx Bytes	Rx Drops	Tx Drops	Rx Errs	Tx Errs	Rx Frame Errs	Rx OverRun Errs	Rx CRC Errs	Collisions
openflow:1:1	82	380	5932	40745	0	0	0	0	0	0	0	0
openflow:1:2	245	427	21773	36851	0	0	0	0	0	0	0	0
openflow:1:LOCAL	0	0	0	0	0	0	0	0	0	0	0	0
openflow:1:3	346	326	30411	28213	0	0	0	0	0	0	0	0
openflow:1:4	11	493	870	42261	0	0	0	0	0	0	0	0

Figura 2-3 Pestaña Nodes, 'Node Connector' punto de acceso ap1. Escenario 1.

Realizado por.- Cevallos, L.2018.

Para conocer las tablas de flujo de cada uno de los puntos de acceso en la consola de comandos de Mininet Wi-Fi, se utilizará el comando >dpctl dump-flows. En la Figura 3-3 se observa las tablas de Flujo de todos los APs del escenario 1.

Figura 3-3 Tablas de flujo del Escenario 1.

Realizado por.- Cevallos, L.2018.

3.1.1.1. Captura de tráfico en Wireshark

Desde la CLI de Mininet Wi-Fi se inicia Wireshark con el comando *mininet-wifi>wireshark &,* para la captura de tráfico se activa la interfaz hwsim0 desde la consola CLI de Mininet Wi-Fi, la misma permitirá capturar el trafico inalámbrico y para capturar los paquetes OpenFlow se seleccionó la interfaz *enp0s9,* de ser el caso de utilizar el controlador por defecto de Mininet Wi-Fi la captura de trafico de paquetes OF se realizará en la interfaz loopback *lo.*

La captura será tomada al hacer ping desde la sta1 hasta la sta8. En la Figura 4-3 y 5-3 se observa tanto la captura de tráfico en la interfaz *hwsim0* y *enp0s9* respectivamente. En la Figura 4-3 se inicia con los 'Beacon', los cuales son generados por los APs para que otros dispositivos los encuentren.

La sta1 se encuentra dentro del rango de ap1 por lo que recibe los 'Beacon', seguidamente se encuentran 'request' solicitados por las estaciones y el 'response' de los APs, donde las estaciones informan parámetros como el SSID al que debe asociarse, entre otros.

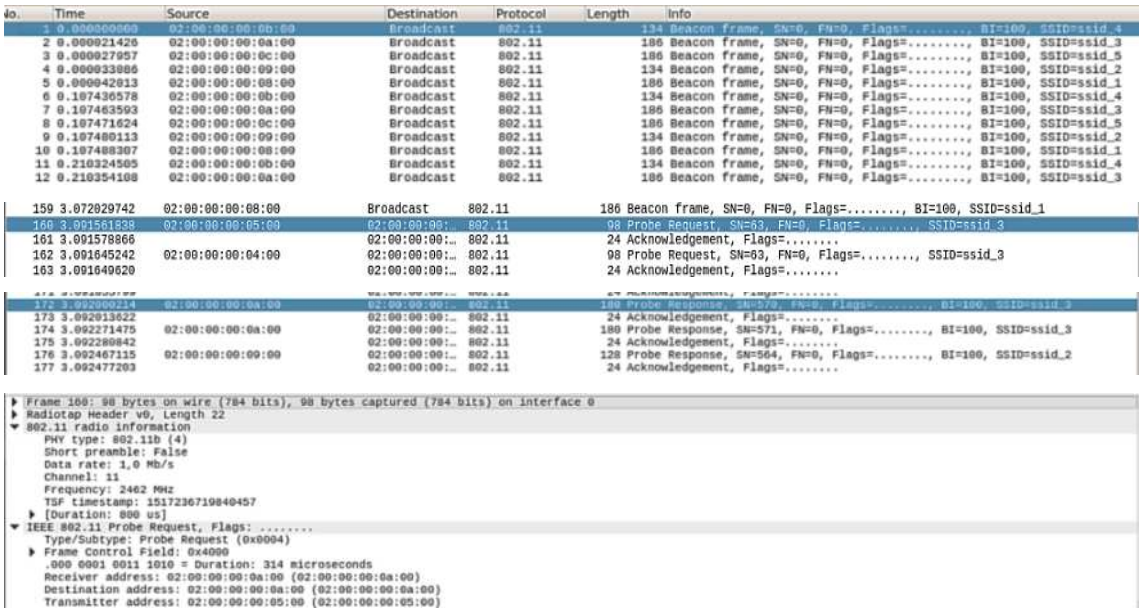


Figura 4-3 Captura de tráfico interfaz hwsim0.

Realizado por.- Cevallos, L.2018.

En la figura 5-3 se observan los paquetes openflow entres sta1 y sta8, cual se lleva a cabo eun ping entres las mismas se puede obsevar, OpenFlow genera paquetes para gestionar la nueva informacion de la red. Cuando esto sucede el dispositivo que recibe el primer paquete envia la controlador 'packet_in' al que contesta con 'packet_out', con este intercambio las estaciones

envia la información de los paquetes que deben retransmitir y el controlador les responde con la informacion necesaria para gestionar la transmisión.

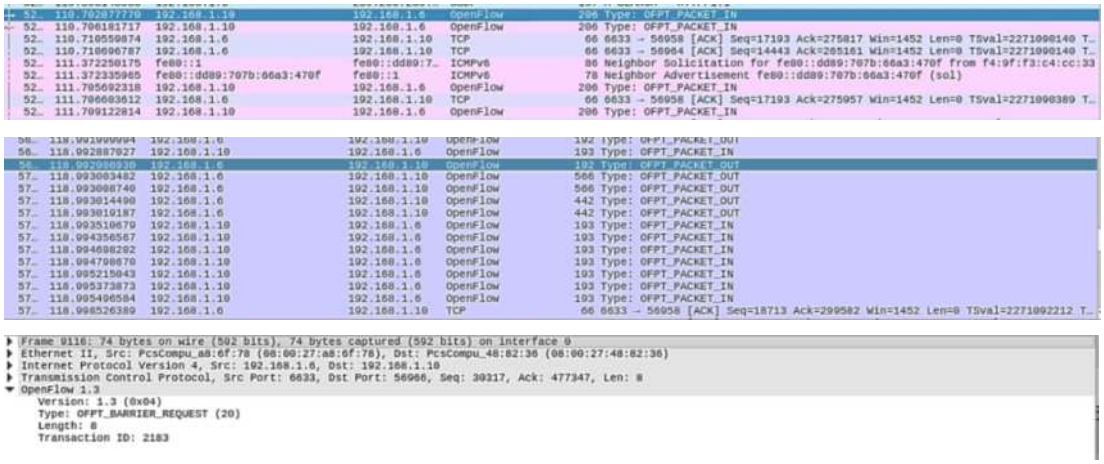


Figura 5-3 Captura de tráfico interfaz enp0s9.

Realizado por.- Cevallos, L.2018.

3.1.1.2. Iperf3 entres estaciones Fijas

En el escenario planteado se llevó a cabo un test con iperf3 como se muestra en la Figura 6-3. el investigador efectúo pruebas para obtener el throughput TCP y UDP. Donde se iniciará con un servidor y un cliente TCP. El servidor estará escuchando por 60sec, en intervalos de 3sec, por el puerto 5566. Delimitando dichos valores para evitar un número excesivo de pruebas las cuales afecten al rendimiento del emulador.

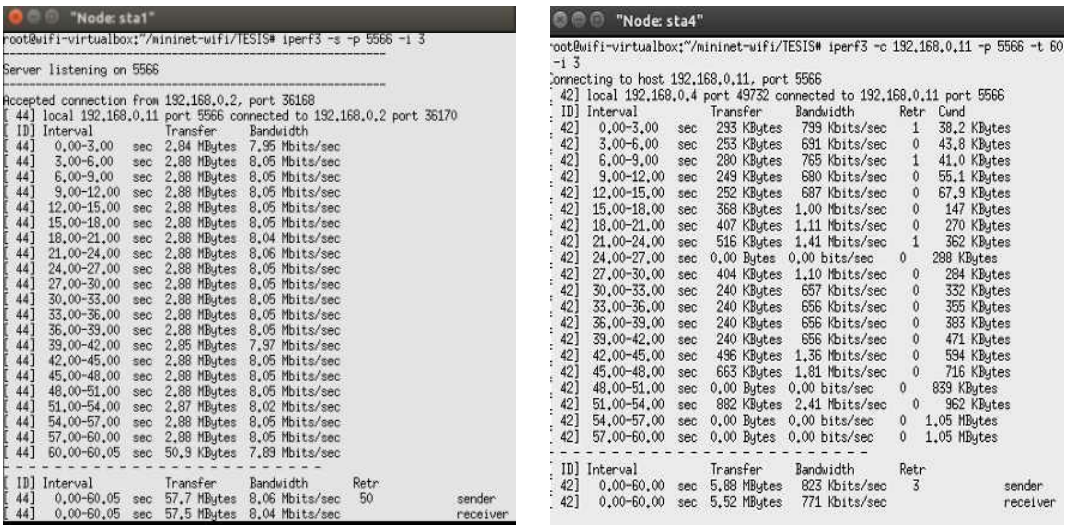


Figura 6-3 Iperf3 TCP. Entre sta1 y sta4. Derecha el servidor, izquierda cliente.

Realizado por.- Cevallos, L.2018.

Se consideraron diferentes distancias para llevar a cabo las pruebas se inicia con una distancia de 45m hasta llegar a una distancia de 290m. En la prueba uno entre las estaciones sta1 y sta2 se tiene una distancia de 45m, en el servidor se puede observar que se transfieren 57.7 MBytes de información, con un ancho de banda de 8.06 Mbits/sec. El mismo procedimiento se lleva acabo para una distancia 95m con las estaciones sta1 y sta4, se observa el cambio en el ancho de banda, en la Figura 7-3.

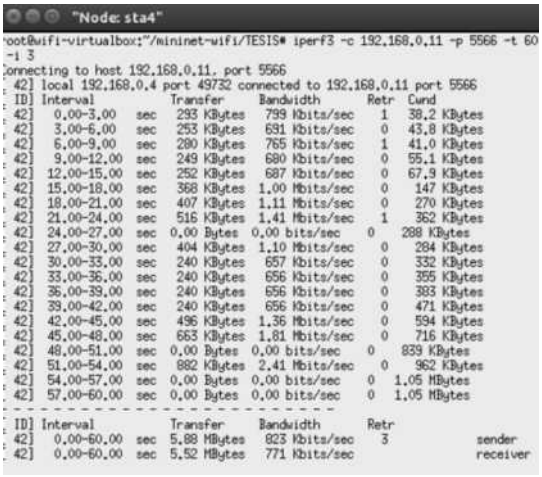


Figura 7-3 Iperf3 TCP. Entre sta1 y sta4

Realizado por.- Cevallos, L.2018.

Se lleva a cabo el mismo procedimiento para UDP. Donde se especifica en el cliente el uso de UDP. Cuando se trabaja con UTP, se obtendrá el jitter y el total de datagramas enviados y perdidos, Figura 8-3. Tanto, con las estaciones más distantes sta1 y sta8, como una de las más cercanas sta1 y sta2.

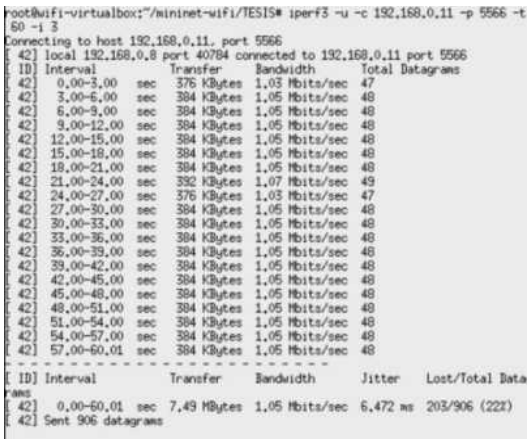


Figura 8-3 Iperf3 UDP. Entre sta1 y sta8.

Realizado por.- Cevallos, L.2018.

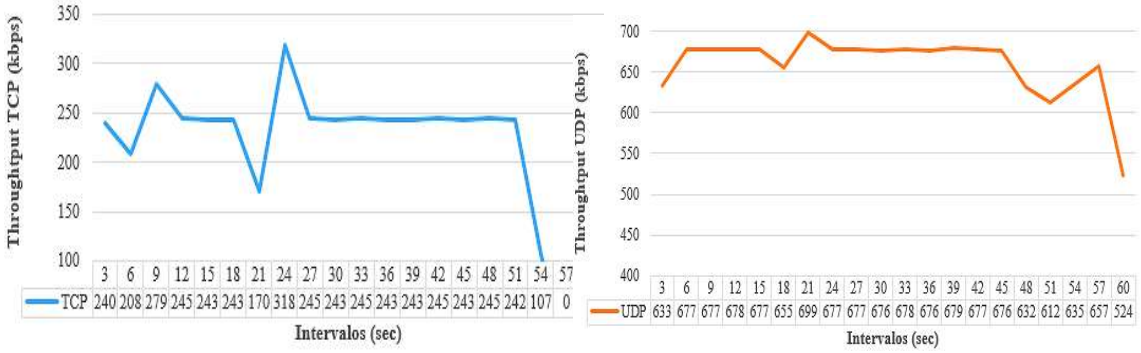


Grafico 1-3 Throughput TCP y UDP. Entre sta1 y sta2.

Realizado por.- Cevallos, L.2018.

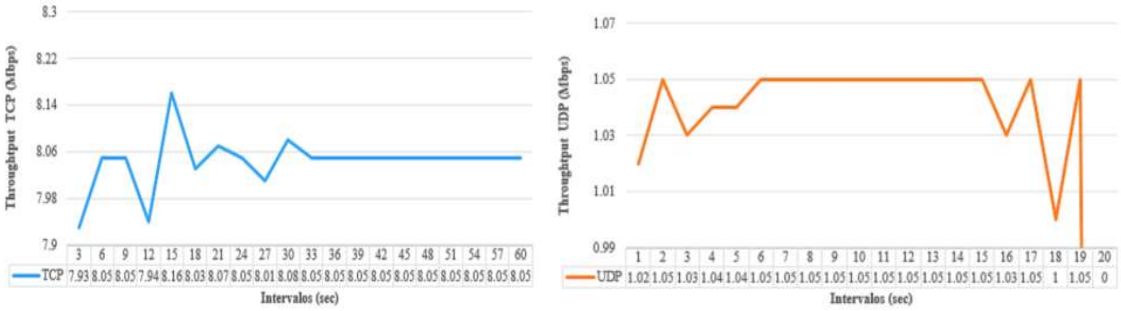


Grafico 2-3 Throughtput TCP y UDP. Entre sta1 y sta8.

Realizado por.- Cevallos, L.2018.

En las Gráficas 1-3 y 2-3 se obtienen los valores del throughput tanto para TCP como para UDP, la mayor diferencia entre las dos graficas radica en que a mayor distancia el throughtput irá decayendo de Mbits/sec a Kbits/sec, debido al deterioro de la señal o por un número mayor de dispositivos intermedios por los que deberán pasar los paquetes hasta llegar al destino.

Así también, cuando se realizó el *iperf3* con UDP se obtiene un jitter de 30.025ms y una pérdida de paquetes de un 20%, considerando a la sta1 como servidor y la sta8 como cliente. En el caso de sta1 como cliente y sta2 servidor, el jitter fue de 7.4ms y una pérdida de paquetes del 0.73%.

De igual manera, se han creado dos archivos de 1MB y 500GB para ser transferidos entre las estaciones más cercanas sta1 y sta2 y las más alejadas sta1 y sta8. Como se observa en las Figura 9-3 y 10-3

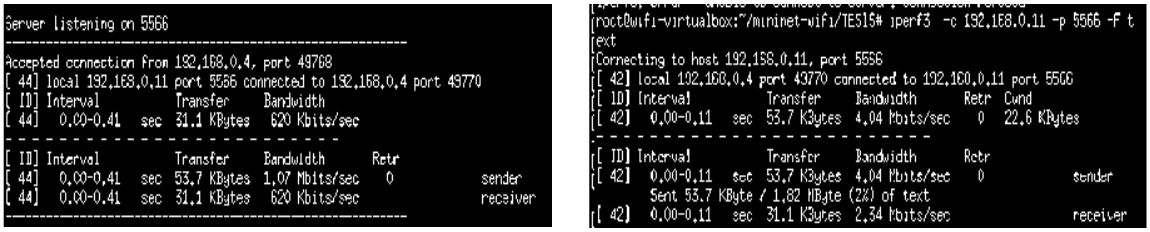


Figura 9-3 Iperf3 de archivo de tamaño 1MB. Entre sta1 y sta8.

Realizado por.- Cevallos, L.2018.

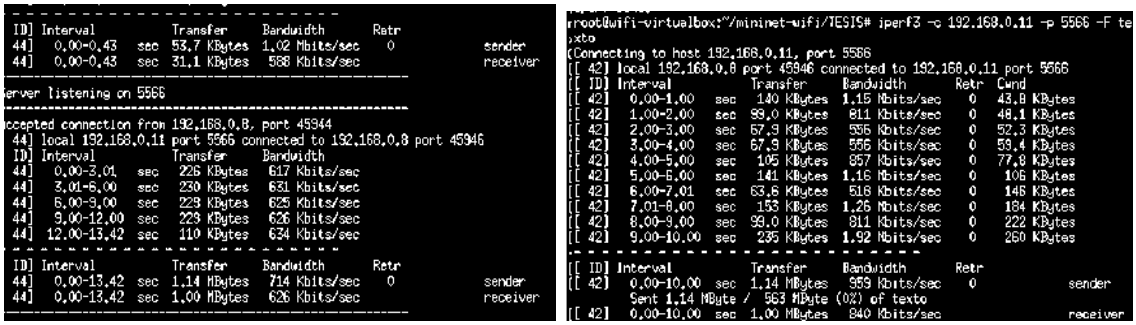


Figura 10-3 Iperf3 de archivo de tamaño 500MB. Entre sta1 y sta8.

Realizado por.- Cevallos, L.2018.

El intervalo de tiempo de envió de información pasa de 0.11 sec a 10 sec para un tamaño de archivos de 1MB y 500MB respectivamente. Así, también para enviar un tráfico distinto a los datos, se envió video entres las estaciones siendo el emisor de video la sta1 y las receptoras tanto la sta2 como la sta8, en la Figura 11-3 se observa el streaming de video entre las estaciones antes mencionadas.

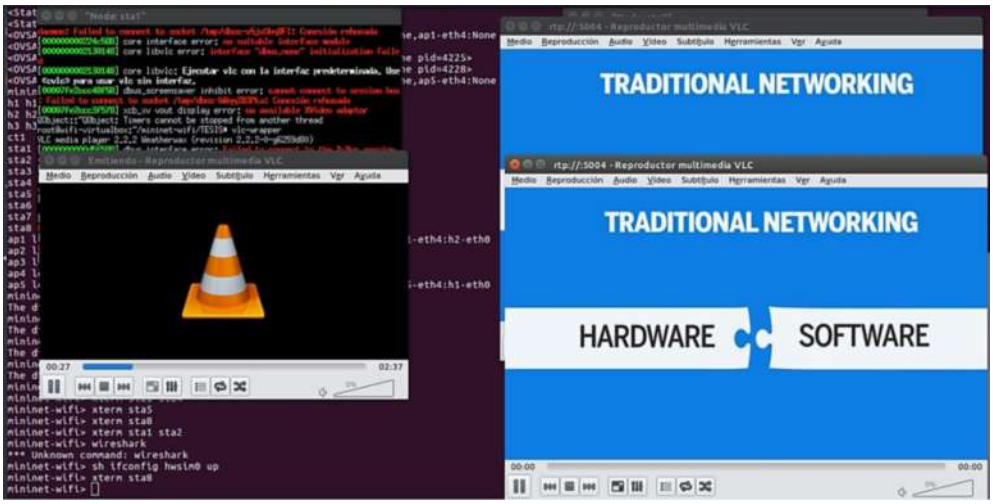


Figura 11-3 Streaming de video.

Realizado por.- Cevallos, L.2018.

El streaming del video se llevó a cabo con la ayuda de VLC Media Player, software que permite establecer una estación como emisora y al indicar las direcciones IP de las estaciones receptoras el video llegara a las mismas sin ningún inconveniente. Se transmite un archivo de video con extensión *.mp4* de 6.72MB. Para completar él envió de streaming de video se llevó a cabo el análisis de tráfico con Wireshark, Figuras 12.3 y 13.3

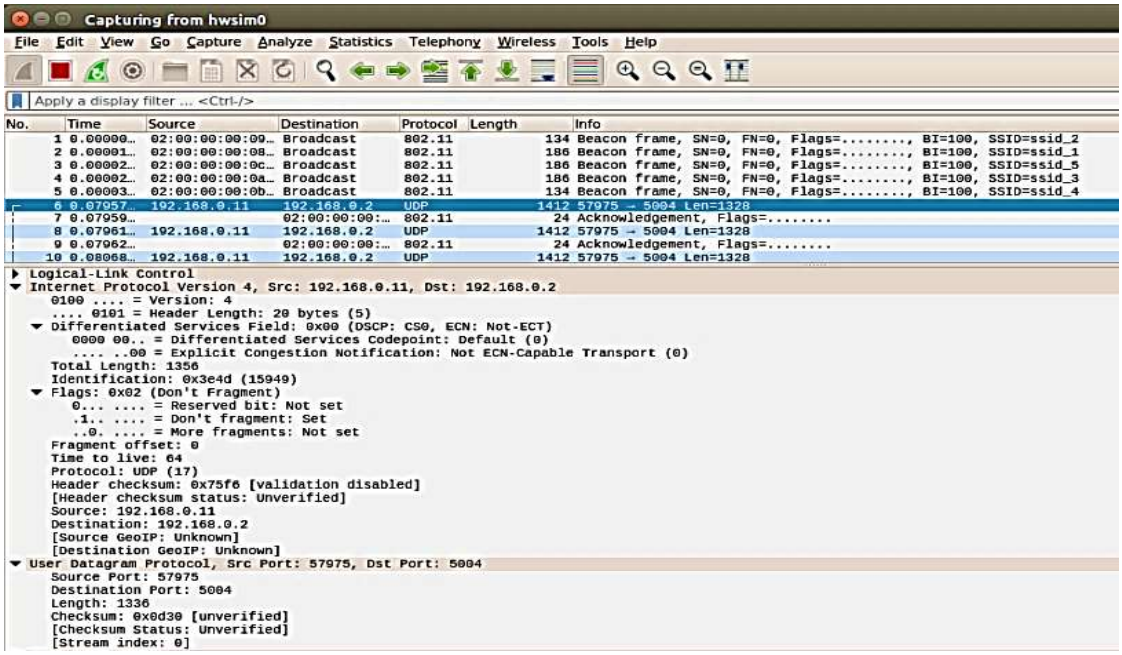


Figura 3-12.3 Streaming de video, captura con Wireshark entre sta1 y sta2.

Realizado por.- Cevallos, L.2018.

Para capturar el trafico inalámbrico se activa la interfaz hwsim0, para emitir el video se emplea el protocolo RTP (Protocolo de transporte en tiempo real) protocolo de nivel de aplicación; empleado para la transmisión en tiempo real de audio y video.

Generalmente, utiliza UDP en la capa de transporte para llevar a cabo la conexión, en la Figura 12-3 se observa los paquetes UDP enviados desde la estación emisora (sta1) hasta su destino (sta2), en la parte inferior encontramos el puerto de destino seleccionado 5004 en este caso.

Así, también en la Figura 13-3, se observa la captura de paquetes RTCP (Protocolo de Control RTP) también a nivel de aplicación, empleado para el control de la información en una sesión RTP.

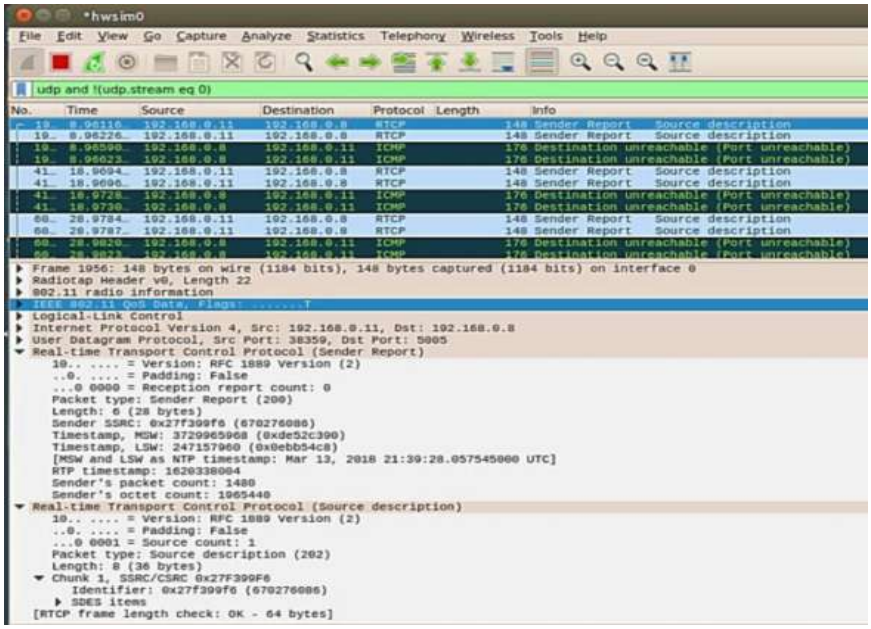


Figura 13-3 Streaming de video, captura con Wireshark entre sta1 y sta8.

Realizado por.- Cevallos, L.2018.

3.1.2. Topología personalizada en scripts Python con estaciones móviles inalámbricas usando un controlador remoto y modelos de movilidad.

De manera similar al escenario anterior se parte de la creación del script en Python, se virtualiza la red en Mininet Wi-Fi y se obtendrá el escenario tanto en Mininet Wi-Fi como en la interfaz web del controlador, Figuras 9-3 y 10-3.

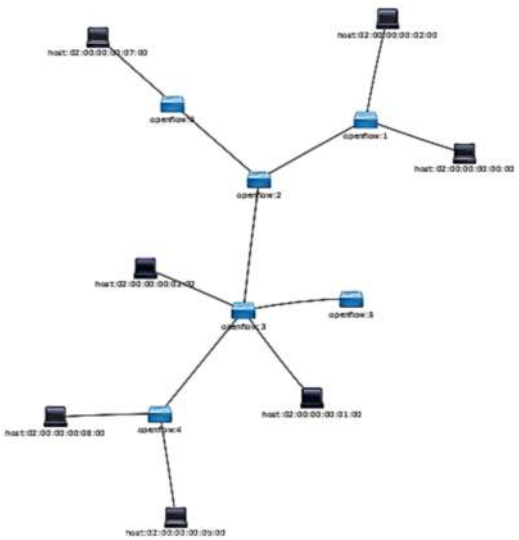


Figura 14-3 Topología en ODL escenario 2.

Realizado por.- Cevallos, L.2018.

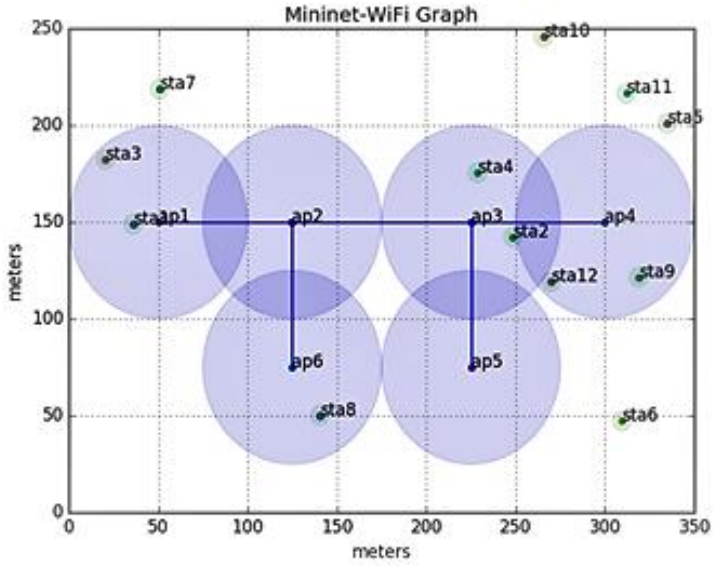


Figura 15-3 Topología en Mininet Wi-Fi escenario 2.

Realizado por.- Cevallos, L.2018.

La topología visualizada en la interfaz web del ODL (Figura 9-3), cambiará de acuerdo a los movimientos de las estaciones; como por ejemplo la sta2 podría estar asociada con el ap1 al inicio de la simulación, pero debido a la movilidad de la misma en todo la cuadricula de simulación, estará ira cambiando la asociación con el punto de acceso más cercano. Para el control de asociación de las estaciones de utilizo *'ssf'* (strongest signal first), las estaciones se asocian con la señal más fuerte primero.

En el CLI de Mininet Wi-Fi podemos obtener mayor información del diseño del escenario, a través de comandos como: *net, dump links*, entre otros. El comando *dump* muestra información de los nodos, direccionamiento IP y *net* muestra cómo están conectados los dispositivos, como se observa en la Figura 11-3.

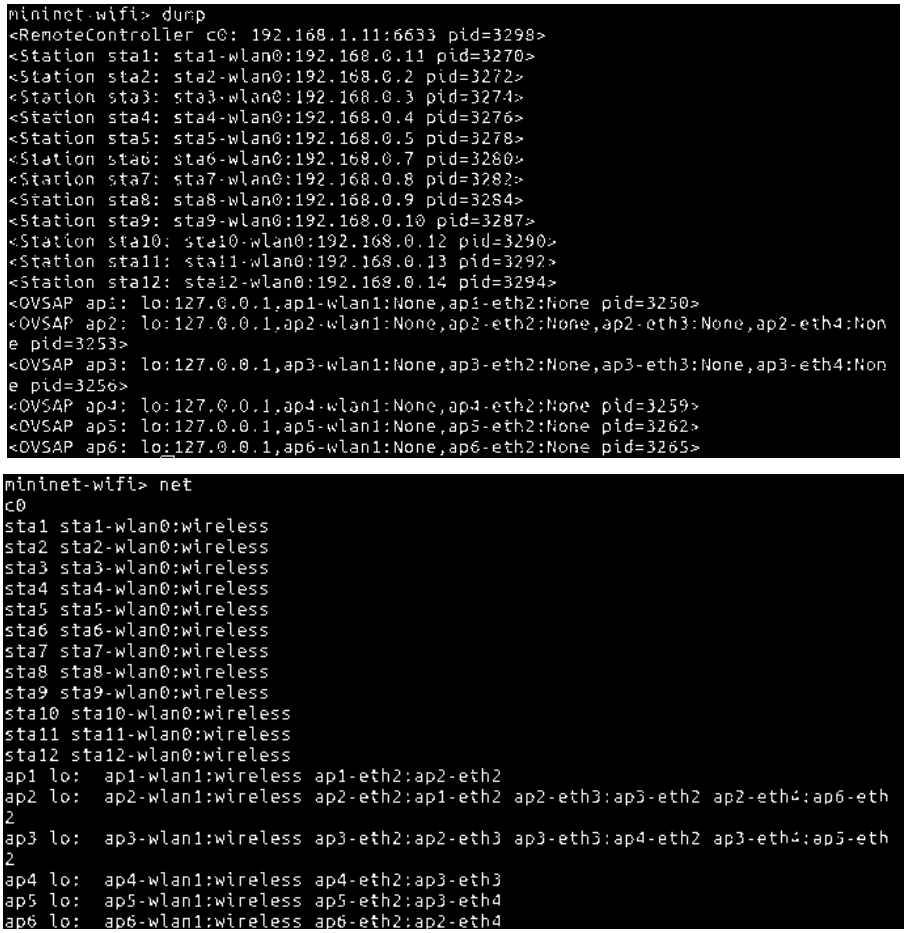
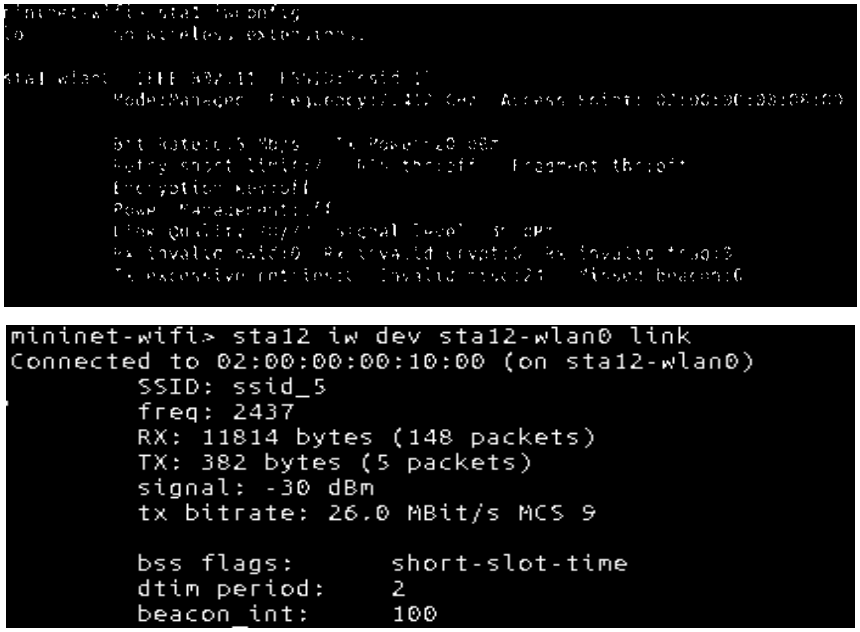


Figura 16-3 Comando *dump* y *net* en CLI de Mininet Wi-Fi.

Realizado por.- Cevallos, L.2018.

Con los comandos anteriores se verifica las conexiones de los puntos de acceso, debido a que las estaciones están conectadas de manera inalámbrica es necesario emplear otros comandos: *stax iwconfig o staX iw dev stax-wlan0 link.* Figura 12-3, donde x representa el número de la estación.



```
mininet-wifi> sta12 iw dev sta12-wlan0 link
Connected to 02:00:00:00:10:00 (on sta12-wlan0)
        SSID: ssid_5
        freq: 2437
        RX: 11814 bytes (148 packets)
        TX: 382 bytes (5 packets)
        signal: -30 dBm
        tx bitrate: 26.0 MBit/s MCS 9

        bss flags:      short-slot-time
        dtim period:    2
        beacon int:     100
```

Figura 17-3 Verificación de conexiones de las STAs.

Realizado por.- Cevallos, L.2018.

De igual manera podemos obtener las tablas de flujo de los dispositivos de la red. Como se indica en la Figura 18-3.

```
mininet-wifi> dpctl dump-flows
*** ap1 ------------------------------------------------------------------------
NXST_FLOW reply (xid=0x4):
*** ap2 ------------------------------------------------------------------------
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=29.092s, table=0, n_packets=1, n_bytes=42, idle_timeout=60, idle_age=29, prio
rity=65535,arp,in_port=1,vlan_tci=0x0000,dl_src=02:00:00:00:03:00,dl_dst=02:00:00:00:00:00,arp_spa=
192.168.0.4,arp_tpa=192.168.0.11,arp_op=2 actions=output:3
 cookie=0x0, duration=23.961s, table=0, n_packets=1, n_bytes=42, idle_timeout=60, idle_age=23, prio
rity=65535,arp,in_port=1,vlan_tci=0x0000,dl_src=02:00:00:00:03:00,dl_dst=02:00:00:00:00:00,arp_spa=
192.168.0.4,arp_tpa=192.168.0.11,arp_op=1 actions=output:3
 cookie=0x0, duration=23.888s, table=0, n_packets=1, n_bytes=42, idle_timeout=60, idle_age=23, prio
rity=65535,arp,in_port=3,vlan_tci=0x0000,dl_src=02:00:00:00:00:00,dl_dst=02:00:00:00:03:00,arp_spa=
192.168.0.11,arp_tpa=192.168.0.4,arp_op=2 actions=output:1
 cookie=0x0, duration=16.135s, table=0, n_packets=1, n_bytes=42, idle_timeout=60, idle_age=16, prio
rity=65535,arp,in_port=1,vlan_tci=0x0000,dl_src=02:00:00:00:0a:00,dl_dst=02:00:00:00:00:00,arp_spa=
192.168.0.13,arp_tpa=192.168.0.11,arp_op=2 actions=output:3
 cookie=0x0, duration=10.907s, table=0, n_packets=1, n_bytes=42, idle_timeout=60, idle_age=10, prio
rity=65535,arp,in_port=1,vlan_tci=0x0000,dl_src=02:00:00:00:0a:00,dl_dst=02:00:00:00:00:00,arp_spa=
192.168.0.13,arp_tpa=192.168.0.11,arp_op=1 actions=output:3
```

Figura 18-3 Tabla de flujos del escenario 2.

Realizado por.- Cevallos, L.2018.

En la interfaz web de ODL se puede obtener información de los puntos de acceso y el número de conexiones en el mismo, en la pestaña Nodes de la interfaz podemos tener mayor detalle de los APs como se observa en la Figura 19-3.

Node Id	Node Name	Node Connectors	Statistics
openflow:6	None	3	Flows | Node Connectors
openflow:5	None	3	Flows | Node Connectors
openflow:4	None	3	Flows | Node Connectors
openflow:3	None	5	Flows | Node Connectors
openflow:2	None	5	Flows | Node Connectors
openflow:1	None	3	Flows | Node Connectors

Figura 19-3 Pestaña 'Nodes' interfaz web de ODL. Escenario 2

Realizado por.- Cevallos, L.2018.

Al dar click en Node Connectors podemos observar algunas estadísticas de los nodos conectados con el ap3, Figura 15-3. Se considera también el cambio de los nodos conectados a los diferentes APs, debido al movimiento de las estaciones.

Node Connector Id	Rx Pkts	Tx Pkts	Rx Bytes	Tx Bytes	Rx Drops	Tx Drops	Rx Errs	Tx Errs	Rx Frame Errs	Rx OverRun Errs	Rx CRC Errs	Collision
openflow:3:3	59	162	6097	13952	0	0	0	0	0	0	0	0
openflow:3:4	42	180	5152	15100	0	0	0	0	0	0	0	0
openflow:3:LOCAL	0	0	0	0	0	0	0	0	0	0	0	0
openflow:3:1	66	183	4320	19373	0	0	0	0	0	0	0	0
openflow:3:2	110	112	10074	10178	0	0	0	0	0	0	0	0

Figura 20-3 Nodos conectados al ap3. Escenario 2

Realizado por.- Cevallos, L.2018.

3.1.2.1. Captura de tráfico en Wireshark

En la Figura 16-3 y 17-3, se llevó a cabo un *ping* entre una las estaciones sta4 y sta3. En la primera imagen Figura 15-3 se muestra a la estación sta4 dentro del rango de cobertura del ap6, del mismo modo la sta3 se encuentra asociada con el ap4 al momento de realizar es *ping;* este se realiza con éxito.

El movimiento de las estaciones provoca el cambio constante de sus posiciones, y tanto la sta4 como la sta3 pasan a no estar asociadas con ninguno de los puntos de acceso, por lo que será imposible llegar a conectarse; el *ping* no alcanza su destino. Figura 16-3

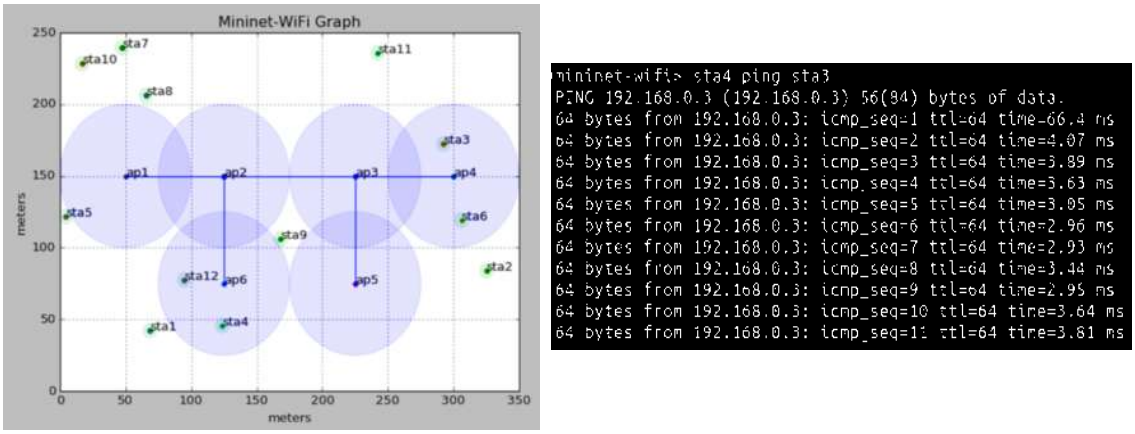


Figura 21-3 Nodos conectados al ap3. Escenario 2

Realizado por.- Cevallos, L.2018.

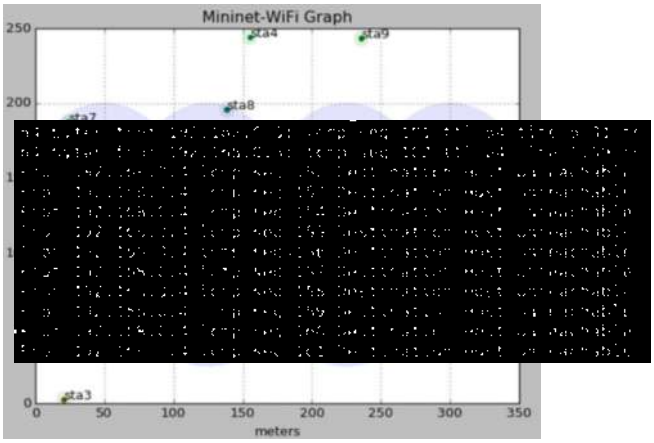
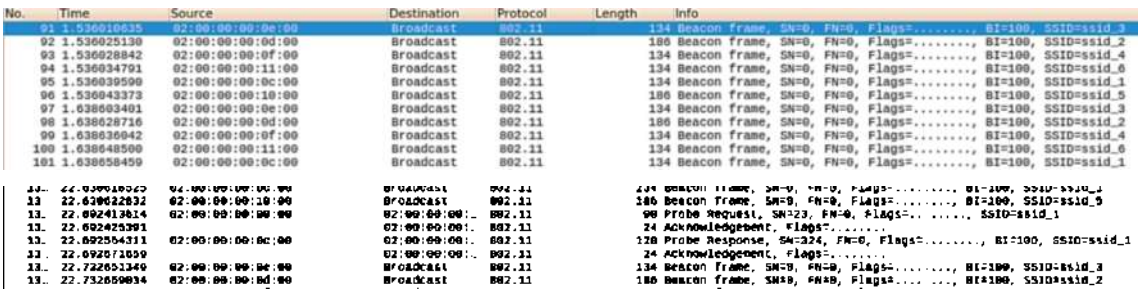


Figura 22-3 Nodos conectados al ap3. Escenario 2

Realizado por.- Cevallos, L.2018.

Cuando las estaciones estén dentro del rango de cobertura nuevamente, se llevó a cabo el *ping* nuevamente. Para llegar a un mejor análisis del todo el tráfico al realizar el *ping* se realiza capturas de tráfico en Wireshark. Se inicia con la captura en la interfaz hwsim0, la misma será activada desde la CLI de Mininet Wi-Fi.



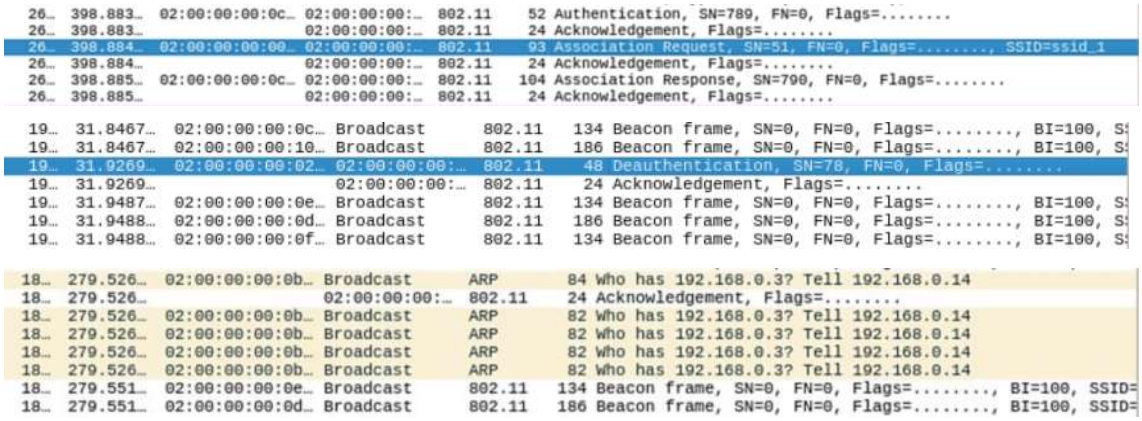


Figura 23-3 Captura de tráfico Wi-FI, interfaz hwsim0, Escenario 2

Realizado por.- Cevallos, L.2018.

La Figura 23-3 muestra la captura de tráfico Wi-Fi, en un inicio podemos encontrar '*Beacon*', generados por los APs para que otros dispositivos los encuentren. Cuando las estaciones se encuentran dentro del rango de las estaciones, se obtiene '*request*' de parte de las STAs y los 'response' de parte de los APs; la estación informa los parámetros de conexión y el SSID al que se está asociando.

Cuando los APs como las STAs tienen conocimiento de la existencia del otro, empieza el proceso de conexión. La estación se autentifica '*Authentication*' y solicita asociarse '*Association*' con el punto de acceso con la señal más fuerte, en el caso de utilizar *ssf* en el control de asociación.

Al llegar al límite del rango de cobertura del AP la estación iniciara el proceso de desconexión '*Deauthentication*'. Así, también podemos encontrar paquetes ARP, utilizado para obtener la dirección MAC perteneciente a cada dirección IP.

Para capturar los paquetes OF se llevó a cabo la captura en la interfaz *enp0s9*, Figura 19-3. los paquetes '*packet_in'* se consigue la versión de OF y también se obtiene la posición de OF en la encapsulación.

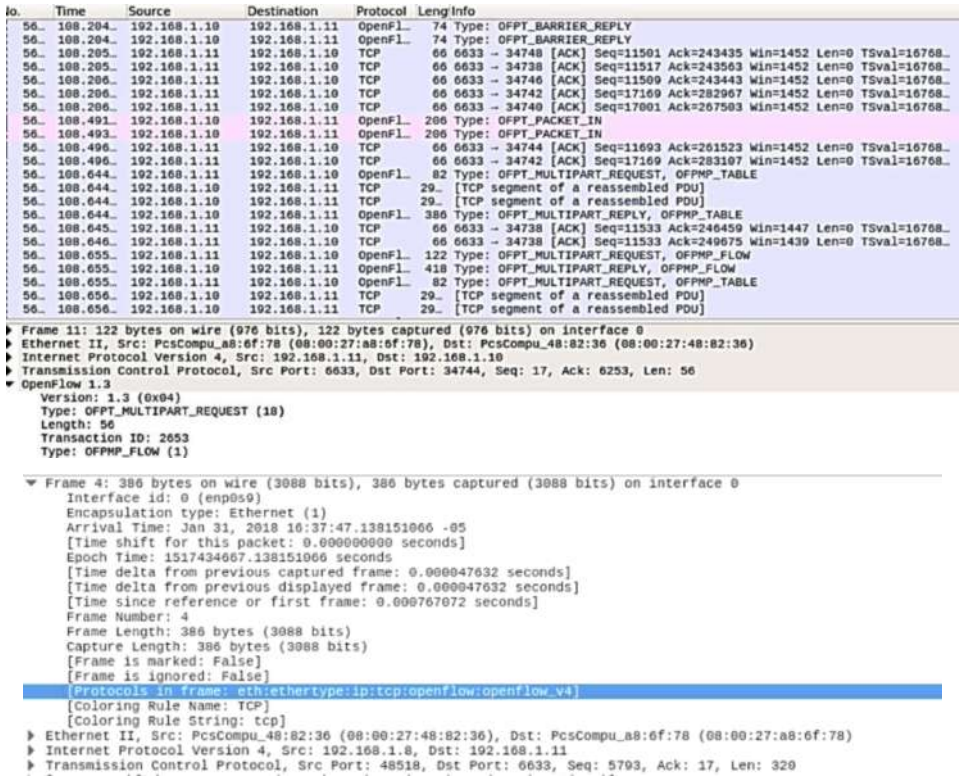


Figura 24-3 Características del mensaje OpenFlow

Realizado por.- Cevallos, L.2018.

Se realizó la captura de tráfico en la interfaz *eth2* del punto de acceso *ap5*, en la cual se obtiene 3 tipos de paquetes ICMP, LLDP Y ARP. Los paquetes LLCP son generados por los switch OpenFlow.

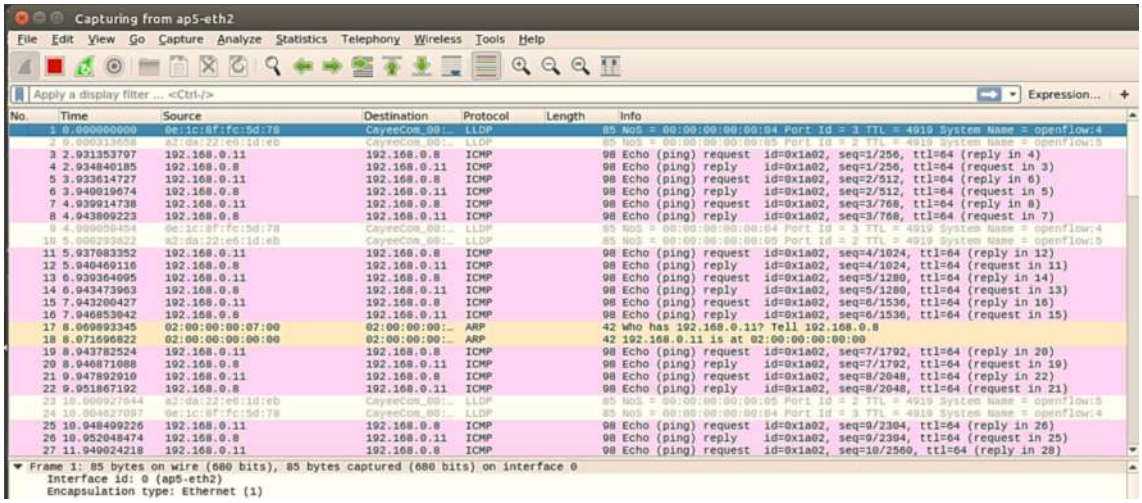


Figura 25-3 Captura de tráfico, interfaz *eth2* del *ap5*. Escenario 2.

Realizado por.- Cevallos, L.2018.

3.1.2.2. Iperf3 en estaciones móviles

En el escenario 2 se lleva a cabo un test con iperf3 como se muestra en la Figura 21.3. Se obtiene el troughput TCP y UDP para cada uno de los modelos de movilidad, mencionados anteriormente. En los scripts Python se cambiará el modelo de movilidad para obtener los datos de cada tipo de movilidad. Se llevará a cabo el test con *iperf3* durante 60s, en intervalos de 3sec.

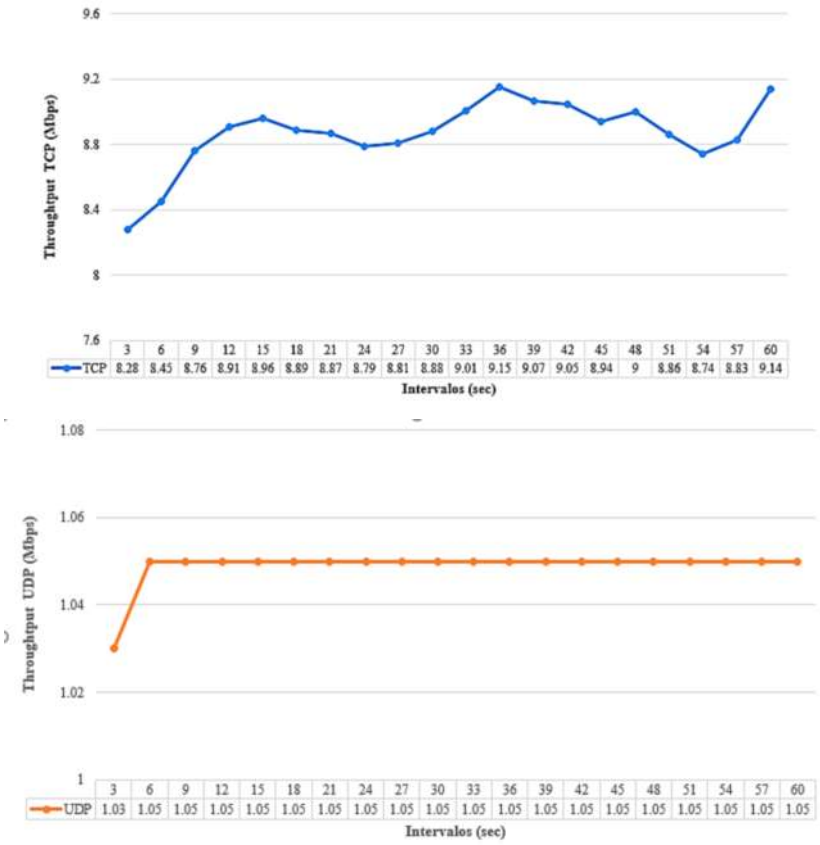


Grafico 3-3 Iperf3 TCP y UDP. Escenario 2. Modelo RandomWalk.

Realizado por.- Cevallos, L.2018.

De manera similar, el mismo test se llevó a cabo con cada tipo de movilidad. En la Gráfico 3-3, se detalla las variaciones de throughtput tanto para TCP como UDP, para el modelo RandomWalk, los flujos de datos de TCP tiene un promedio de throughput de 8.88 Mbps, y en UDP de 1.05 Mbps. La Tabla 1-3, muestra el promedio de throughput de TCP y UDP para los modelos de movilidad.

Tabla 1-3 Promedio de Throughtput TCP y UDP.

Modelo de Movilidad	Promedio de Throughtput TCP (Mbps)	Promedio de Throughtput UDP (Mbps)
RandomWalk	8.88	1.05
TruncatedLevyWalk	3.37	1.05
RandomDirection	8.80	1.05
RandomWayPoint	0.59	1.05
GaussMarkov	9.09	0.998
ReferencePoint	8.34	1.05
TimeVariantCommunity	8.03	0.929

Realizado por.- Cevallos, L.2018.

En la Tablas el modelo RandomWayPoint tiene el menor throughtput en TCP de 0.59, mientras que para UDP los valores se encuentran entre a 0.9 y 1.05 para todos los modelos.

En los test con UDP se obtienen de igual manera los valores de jitter (ms) y el porcentaje de la perdida de datagramas (paquetes de datos) para cada uno de los modelos de movilidad. Los resultados de los mismo se pueden observar en los Gráficos 4-3 y 5-3. Para mayor detalles de cada test ver Anexo G.

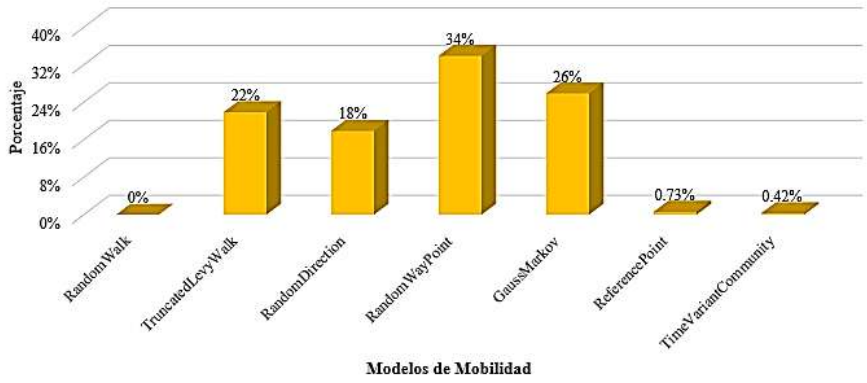


Grafico 4-3 Perdida de datagramas en los modelos de movilidad

Realizado por.- Cevallos, L.2018.

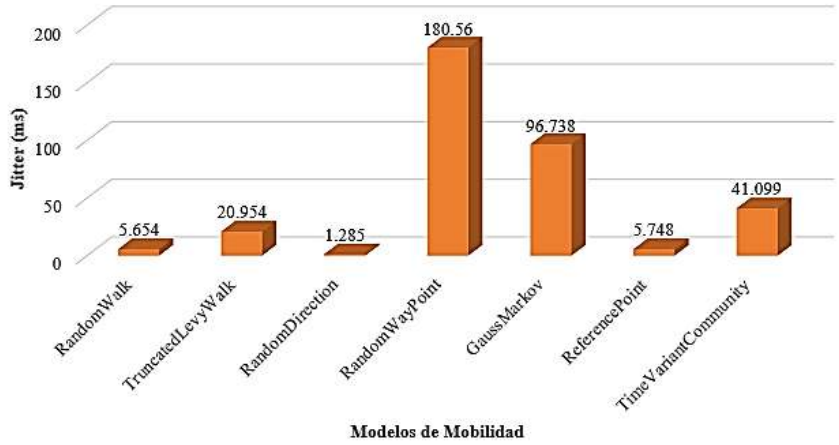


Grafico 5-3 Jitter en los modelos de movilidad

Realizado por.- Cevallos, L.2018.

El modelo RandomWayPoint presenta la mayor pérdida de datagramas con un 34% y un jitter de 180.56ms, por otro lado, se podría decir que los mejores modelos son ReferencePoint y RandomWalk con un 0,73% y 0% de perdida paquetes y con un jitter de 5.64ms y 5.7ms respectivamente.

Al igual que con las estaciones fijas se han creados dos archivos de 1MB y de 500MB, los cuales serán enviados entre las estaciones, las ubicaciones de las mismas irán cambiando de acuerdo el tipo de movilidad que se emplea en los scripts de Python. En todos los modelos de movilidad se mantienen una variación de segundos en la transferencia de los datos, entre un rango de 0.12 sec y 10 sec. En las Figuras 26-3 y 27-3 se observa él envió de los archivos

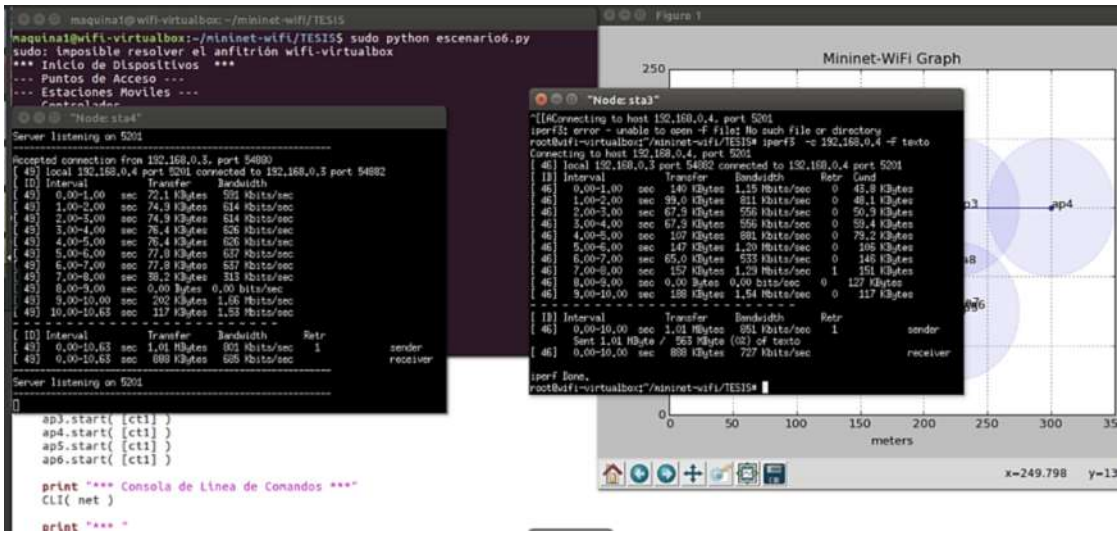


Figura 26.3 Iperf3 de archivo de tamaño 500MB. Tipo de movilidad GaussMarkov

Realizado por.- Cevallos, L.2018.

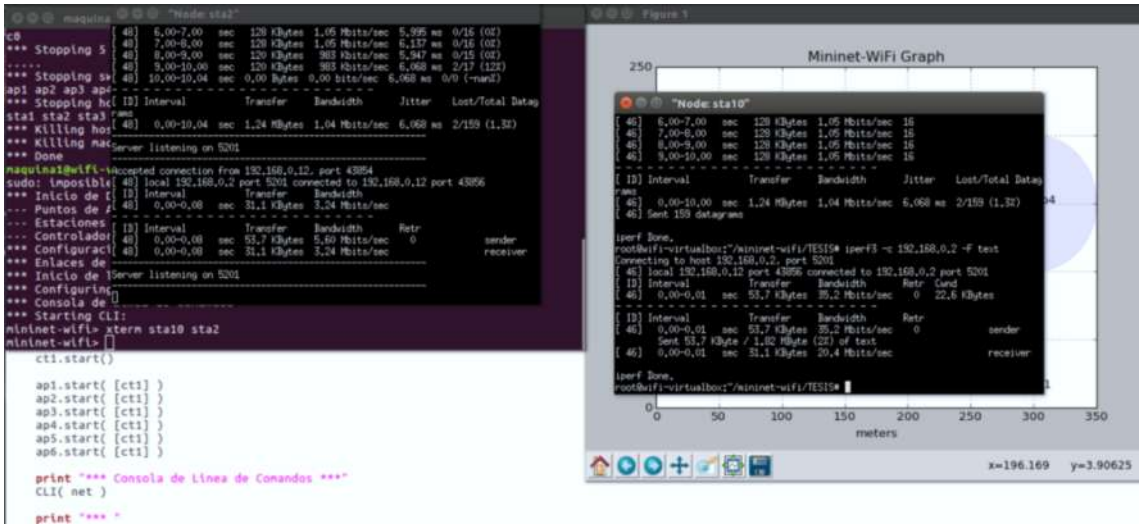


Figura 27-3 Iperf3 de archivo de tamaño 1MB. Tipo de movilidad ReferentPoint

Realizado por.- Cevallos, L.2018.

De igual manera se efectúa el streaming de video entre diversas estaciones, tomando en consideración la movilidad de las mismas, Figura 28-3. Debido a la movilidad de las estaciones se evidenció una demora en la reproducción del video por parte de los receptores, de igual manera al salir de área de cobertura las estaciones pierden comunicación con la estación receptora, así, en la reproducción del video se vio pausada, además de la pérdida del audio y de la imagen hasta nuevamente establecer la conexión.

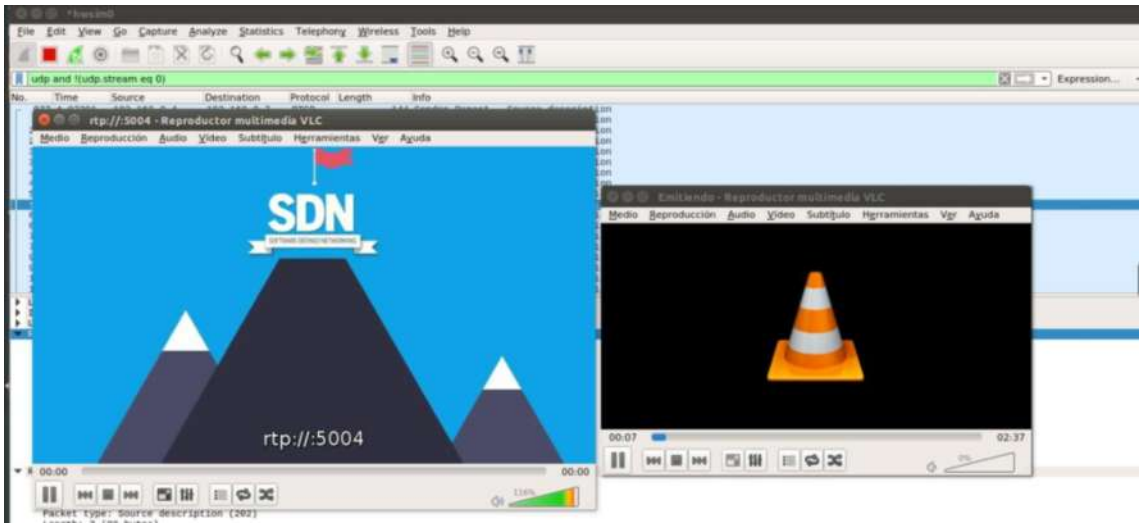


Figura 28-3 Streaming de video. Tipo de movilidad TruncatedLevyWalk

Realizado por.- Cevallos, L.2018.

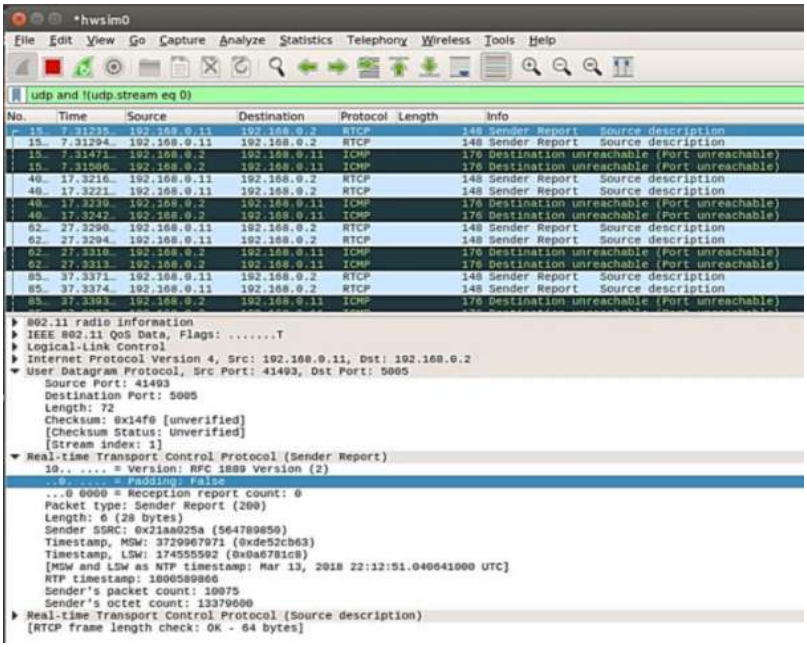


Figura 29-3 Streaming de video. Tipo de movilidad RandomWalk

Realizado por.- Cevallos, L.2018.

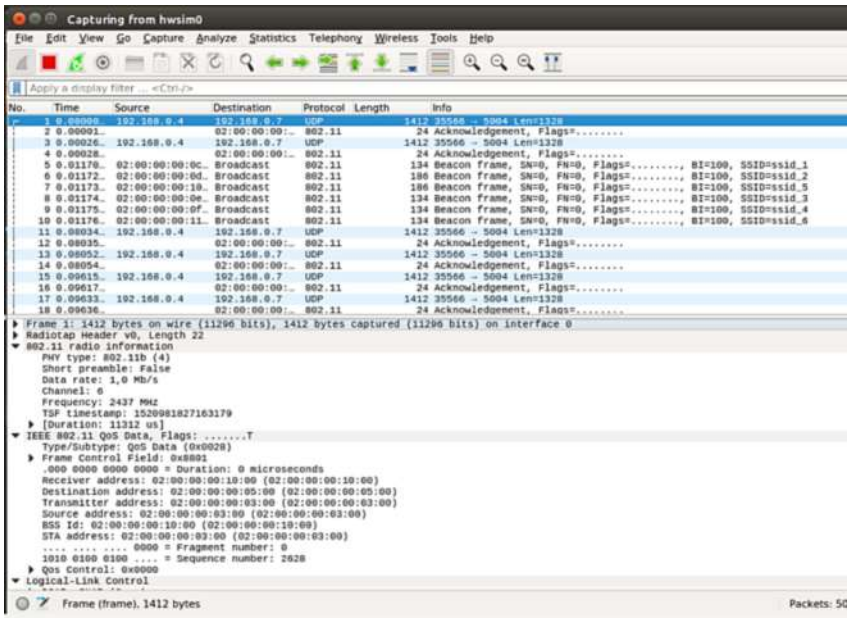


Figura 30-3 Streaming de video. Tipo de movilidad TruncatedLevyWalk

Realizado por.- Cevallos, L.2018.

En las Figuras 29-3 y 30-3 se observa la captura de trafico inalámbrico, los paquetes encontrados de UDP son utilizados por RTP para enviar el video a las estaciones receptoras, y para el control de sesión con RTP se emplea el protocolo RTCP.

CONCLUSIONES

- La mayoría de los despliegues de SDN hoy en día, son de SDN cableadas. Como con cualquiera otra evolución tecnológica, la migración a SDWN se llevará a cabo en fases. De esta manera se pretende dar una gestión completa y unificada de toda la red cableada e inalámbricas de empresas e instituciones.

- Las SDN conjuntamente con el protocolo Openflow, presentan multiples beneficios, logran funcionar en casi cualquier equipo indistintamente de su proveedor; proporcionan un control detallado de la red al verla como una sola unidad, facilitan su gestión y control. Así, también las SDWN proporcionan las características más importantes de las SDN para mejorar las redes inalámbricas, al optimizar el mayor despliegue de servicios, el mejor rendimiento de aplicaciones, entre otras en toda la red.

- Mininet Wi-Fi, aporta en el campo de las SDWN no solo con la implementación de los diferentes modelos de movilidad, también es posible definir modelos de propagación, o llevar a cabo la implementación conjuntamente con dispositivos reales.

- MiniEdit es la herramienta grafica de Mininet, proporciona al usuario una manera más sencilla de crear escenarios con hosts, swicthes, controladores, entre otros. Sin embargo, no facilita el diseño de topologías con estaciones y puntos de acceso inalámbricos.

- Las herramientas como Mininet Wi-Fi y el controlador OpenDayLight, facilitan la implantación de SDWN en un entorno virtual, al facilitar la implementación de diversas topologías de red y su posterior estudio.

- Cada modelo de movilidad implementado en Mininet Wi-Fi se establecerá con el análisis de trazas diferentes, las cuales determinaran el patrón de movimiento de los nodos inalámbricos para cada modelo.

- Al realizar las pruebas de rendimiento para el escenario con los diferentes modelos de movilidad, se obtiene que los modelos ReferencePoint y RandomWalk obtiene un jitter de 5,64ms y 5,7ms, respectivamente, y valores de 0.73% y 0% en la pérdida de datagramas, debido a que RandomWalk crea patrones de movilidad más realistas y flexibles, mientras que ReferencePoint crea patrones de comportamiento grupal.

RECOMENDACIONES

- Se sugiere la implementación conjunta de las herramientas para SDWN: Mininet Wi-Fi y el controlador OpenDayLight con dispositivos reales, con escenarios con los diferentes tipos de movilidad, además de la incorporación de los modelos de propagación indoor y outdoor para controlar el flujo de datos.

- Para la instalación de las plataformas de simulación, se recomienda llevar a cabo la instalación de Mininet Wi-Fi y el controlador OpenDayLight en máquinas físicas diferentes, debido a que utilizan muchos recursos para su instalación y ejecución.

- Para el correcto funcionamiento del controlador OpenDayLight, se necesita de la instalación de los *features* (características) del controlador de acuerdo a las necesidades de los usuarios.

- Estudiar el comportamiento de la herramienta Mininet Wi-Fi con varios controladores, basados en diferentes lenguajes de programación, para determinar su comportamiento con las SDWN.

- Llevar a cabo un mayor número de pruebas con equipos de mayor capacidad de procesamiento, para optimizar la virtualización de escenarios más complejos, con un mayor número de puntos de acceso y estaciones.

GLOSARIO

API.- Conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizada por un software como una capa de abstracción.

IDE.- Aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software, contiene un compilador, un intérprete, o ambos.

Linux Kernel (Núcleo de Linux).- Columna vertebral de cualquier sistema Linux, piedra angular del desarrollo de código abierto. Se tiene acceso al mismo en modo Privilegiado.

MLME.-Entidad gestora donde residen las máquinas de estado MAC capa física (PHY), autenticación, asociación, envío y recepción de beacons.

WPA.- Solución de seguridad inalámbrica, depende de protocolos de autenticación y de un algoritmo de cifrado cerrado: TKIP (Protocolo de Integridad de Clave Temporal) genera claves aleatorias.

Canal.- Medio de transmisión entre el emisor y receptor, por el que viajan las señales portadoras de la información.

Iperf3.- Herramienta para llevar a cabo pruebas en redes informáticas. Su funcionamiento se basa en crear flujos de datos TCP y UDP y medir el rendimiento de la red.

BIBLIOGRAFÍA

ACOSTA, Eva. Conectividad por wifi va en aumento en ciudades del Ecuador. *El 92% de usuarios móviles se conecta por Wifi en Ecuador.* [ed.] Metro, 2017, pp. 2-4..

Agencia de Regulacion y Control de las Telecomunicaciones. Boletín Estadístico del Sector de Telecomunicaciones. Quito. 2015, pp 22-24.

ALBÁN, Pablo & **BRITO, Darío.** *Diseño e implementación del prototipo de una red definida por software (SDN) en la Universidad de las Fuerzas Armadas ESPE (Trabajo de Titulación).* (Pregrado) Universidad de las Fuerzas Armadas. departamento de Eléctrica y Electrónica. Sangolquí. 2015, p. 21.

ALVAREZ, Raul. *Estudio de las redes definidas por software mediante el desarrollo de escenarios virtuales basados en el controlador OpenDayLight (Trabajo de Titulación).* (Master) Universidad Politécnica de Madrid. Escuela Técnica Superior de Ingenieros de Telecomunicación. Madrid. 2015, pp- 21-49.

ARTECHE, Carlos. *Despliegue de una maqueta de red basada en Openflow (Trabajo de Titulación).* (Pregrado) Universidad de Cantabria. Facultad de Ciencias. España. 2014, pp. 8-20.

BAHIT, Eugenia. *Curso: Python para Principiantes.* Buenos Aires, Argentina : SafeCreative. 2013, pp. 10 - 40.

CABEZAS, Luis & **GONZÁLES , Francisco.** *Redes Inalambricas.* Juan Ignacio Luca de Tena-Madrid : Anaya Multimedia. 2010: pp 15-17.

CENTENO, Alejandro, et al. "Controladores SDN, elementos para su selección y evaluación." s.l. *Revista digital de las Tecnologias de la informacion y las Comunicaciones. Telem@tica*, . Vol. 13. 2014, pp 3-5.

CHICO, Juan. *Implementación de un prototipo de una red definida por software (SDN) empleando una solución basada en hardware (Trabajo de Titulación).* (Pregrado) Escuela Politécnica Nacional. Facultad de Ingeniería Electrónica y Electrónica. Quito . 2013,pp. 30-40.

CHRISTENSSON, P. *Python Definition.* [En línea] Sharpened Productions , 15 de Junio de 2010. [Citado el: 06 de Julio de 2017.] https://techterms.com/definition/python.

Citrix Systems. SDN 101: An introduction to software-defined networking. [En línea] 2017. [Citado el: 01 de Junio de 2017.] https://www.citrix.es/products/netscaler-adc/resources/sdn-101.html.

CRUZ, Marely & **GARCIA, Yosuan.** *"Análisis de la QoS en redes inalámbricas."* . *Revista Cubana de Ciencias Informáticas*. La Habana. 2013, pp. 86 -96.

DUARTE, Eduardo & **LOBO, Richard.** *Emulacion de escenarios virtuales, en una SDWLAN (Software Defined Wireless Local Area Network), de un Campus Universitario. (Trabajo de Titulación)* .Universidad de los Andes., Universidad de los Andes. Venezuela .Vol. 1. 2015, p. 10

ESPAÑA, Natalia. *Diseño y simulación de una red definada por software (SDN) (Trabajo de Titulación).* (Pregrado) Universidad Central del Ecuador. Carrera de Ingeniería Informática. Quito . 2016, pp. 8-10.

FLORIAN, M. *Propagación de señales.* Repositorio Universidad de las Américas. Puebla - Mexico. 2005, p. 2.

FONTES , Ramon, et al. *Mininet-WiFi: Emulating Software-Defined Wireless Networks.* Campinas, Sao Paulo, Brazil : School of Electrical and Computer Engineering. School of Electrical and Computer Engineering. 2015, pp. 2-6.

FONTES, Ramon & **ROTHENBERG, Christian.** *The User Manual, Mininet-WiFi.* Brazil : INFORMATION & NETWORKING TECHNOLOGIES RESEARCH & INNOVATION GROUP. 2017, pp. 9-25.

FONTES, Ramon, et al. *How far can we go? Towards Realistic Software-Defined Wireless Networking Experiments.* s.l. : The Computer Journal. 2017,pp. 5-10.

Galeon. Breve introducción a las Redes Inalámbricas. [En línea] 2010. [Citado el: 20 de 08 de 2017.] http://ieeestandards.galeon.com/aficiones1573087.html.

GONZÁLEZ, Raúl. *Python para todos.* España, 2010. pp. 7-12.

HENAO, Jose. *Guía de implementación y uso del emulador de redes Mininet (Trabajo de Titulación).* (Pregrado) Universidad Tecnologica de Pereira. Ingeniería en Sistemas y Computación. Pereira. 2015, pp- 9-60.

IEEE Working Group. Official IEEE 802.11 Working Group Project Timelines. [En línea] Institute of Electrical and Electronics Engineers, Inc, 15 de Mayo de 2017. [Citado el: 2017 de Mayo de 2017.] http://www.ieee802.org/11/Reports/802.11_Timelines.htm.

INEC, Instituto Nacional de Estadísticas y Censos. Tecnologías de la Informacion y Comunicación (TIC´s). [En línea] 2016, pp. 6 -11. [Citado el: 30 de Octubre de 2017.] www.ecuadorencofras.gob.ec.

Institute of Electrical and Electronics Engineers. IEEE 802.11 Wireless Local Area Networks. [En línea] Institute of Electrical and Electronics Engineers, Inc, 2008. [Citado el: 29 de Mayo de 2017.] http://www.ieee802.org/11/.

JAYAKUMAR, Geetha & **GANAPATHI, Gopinath.** "Reference Point Group Mobility and Random Waypoint Models in Performance Evaluation of MANET Routing Protocols". *Journal of Computer Systems, Networks and Communications*, Vol. 2008. India .2008, pp. 1-8.

LEYTON, Johanna. *Diseño de una red definida por software empleando una solución basada en software, para la infraestructura de Cloud de la facultad de Ingeniería en Ciencias Aplicadas (FICA) (Trabajo de Titulación).* (Pregrado) Universidad Técnica del Norte, Facultad de Ingeniería en Ciencias Aplicadas. Ibarra .2016, pp. 9-20.

Linux Foundation. OpenDaylight: Platform Overview. [En línea] 2016. [Citado el: 06 de Julio de 2017.] https://www.opendaylight.org/platform-overview/.

LOPEZ, Francisco. *El estándar IEEE 802.11. Wireless LAN.* [En línea] 2014, pp. 4-20. [Citado el: 24 de 10 de 2017.] http://www.dit.upm.es//~david/TAR/trabajos2002/08-802.11-Francisco-Lopez-Ortiz-res.pdf.

Magazcitum. *Redes complejas hechas fácilmente, ¿será posible?* [En línea] 2013. [Citado el: 01 de Junio de 2017.] http://www.magazcitum.com.mx/?p=2353.

MARÍN, Yanko. *Plataforma de pruebas para evaluar el desempeño de las redes definidas por software basadas en el protocolo Openflow (Trabajo de Titulación).*(Maestría) Universidad Central "Marta Abreu" de las Villas. Facultad de Ingeniería Electrónica. Cuba, Santa Clara. 2016, pp. 30-40.

MARTÍNEZ , Alberto. *Medium and mobility behaviour insertion for 802.11 emulated networks. (Trabajo de Titulación).* (Maestria) Universidad Politecnica de Catalunya. Barcelona. 2015, pp. 5-11.

MARTÍNEZ, Jackson. *Estudio del funcionamiento de la herramienta Mininet (Trabajo de Titulación).* (Pregrado) Universidad Católica de Pereira. Facultad de Ciencias básicas e Ingenieria. Pereira . 2015, pp. 55-60.

Mininet Team.. Mininet Overview. [En línea] 2017. [Citado el: 08 de Junio de 2017.] http://mininet.org/overview/.

MOSCOSO, Esteban. *Desarrollo de una aplicación para la implementación de calidad de servicio por priorización de tráfico sobre una Red Definida por Software (SDN) (Trabajo de Titulación).*(Pregrado) Escuela Politécnica Nacional. Facultad de ingeniería Eléctrica y Electrónica. Quito. 2016, pp. 28-30.

Network World. SDN marca el futuro del networking. [En línea] 11 de 03 de 2013. [Citado el: 28 de 09 de 2017.] http://www.networkworld.es/sdn/sdn-marca-el-futuro-del-networking.

NÚÑEZ, Alex. *Red definida por software (SDN) en base a una infraestructura de software de libre distribución (Trabajo de Titulación).* (Pregado) Universidad Técnica de Ambato. Facultad de Ingeniería en Sistemas, Electrónica e Industrial. Ambato . 2015, pp. 7-32.

Open Networking Foundation. Open Networking Foundation. [En línea] 2010. [Citado el: 03 de Mayo de 2017.] https://www.opennetworking.org/.

Open Networking Foundation. OpenFlow Switch Specification. [En línea] 26 de 03 de 2015, pp. 4-100. [Citado el: 12 de 08 de 2017.] https://www.opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf.

Python Tutorials. Python Overview. [En línea] 2017. [Citado el: 06 de Julio de 2017.] http://www.tutorialspoint.com/python/python_overview.htm.

RAJASRI, K, et al. SDN and Openflow. A tutorial. [En línea] 2011, pp. 23-26. [Citado el: 06 de Junio de 2017.] https://www.clear.rice.edu/comp529/www/papers/tutorial_4.pdf.

RAMOS, Catalina. "*Openflow abre las nuevas puertas para innovar". REUNA, Ciencia y Educación en la red*, REUNA, Ciencia y educación en la red. Canada . 2012, pp. 4-6.

REID, Neil & **SEIDE, Ron.** *802. 11 (Wi- Fi) Manual de redes inalambricas.* McGraw-Hill Interamericana. Mexico. 2003, pp. 5-10.

SAVU, D & **STANCU, S.** Software Defined Networking: technology details and openlab research overview. [En línea] 14 de Febrero de 2014, pp. 7-37. [Citado el: 02 de Junio de 2017.] http://openlab.cern/publications/presentations/software-defined-networking-technology-details-and-openlab-research.

SDX Central . What is an OpenDaylight Controller? AKA: OpenDaylight Platform. [En línea] 2016. [Citado el: 06 de Julio de 2017.] https://www.sdxcentral.com/sdn/definitions/sdn-controllers/opendaylight-controller/.

SHAHZAMAL, Md, et al. *Mobility Models for Delay Tolerant Network: A Survey.* International Journal of Wireless & Mobile Networks. 2014, pp. 121-130.

VELASCO , Manuel. *Diseño e implementación de una aplicación prototipo para ofrecer el servicio de DHCP sobre una SDN (Trabajo de Titulación).* (Pregrado) Escuela Politecnica Nacional. Quito . 2016,p. 4-15.

WI-FI ALLIANCE. The worldwide network of companies that brings you Wi-Fi. [En línea] Wi-Fi Alliance, 2017. [Citado el: 29 de Mayo de 2017.] http://www.wi-fi.org/.

WOLFGANG, Braun & **MENTH, Michael.** *Software-Defined Networking Using OpenFlow: Protocols.* Department of Computer Science, University of Tuebingen. Alemania. 2014, pp. 5-10.

YÁNEZ, Catherine & **GALLEJOS, Fabián.** *Implementación de un Prototipo de Red Definida por Software para el Hotspot-Espoch Mediante un Controlador Basado en Openflow (Trabajo de Titulación).* (Pregrado) Escuela Superior Politécnica de Chimborazo. Facultad de Informática y Electrónica. Ecuador. 2015, pp. 9-12.

ANEXOS