UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

Estudio de redes SDN mediante Mininet y MiniEdit

Sergio Córdoba López

Tutor: José Óscar Romero Martínez

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València, para la obtención del Título de Graduado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Curso 2017-18

Valencia, 9 de septiembre de 2019

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

VLC/ CAMPUS
VALENCIA, INTERNATIONAL CAMPUS OF EXCELLENCE

EMAS
Gestión ambiental verificada
REG.NO. ES-CV- 000030

Resumen

En respuesta a la creciente evolución de la demanda y de la necesidad de desarrollo de las redes de comunicación aparece las “Sofware Difened Network” o “SDN”, una nueva y revolucionaria visión de la arquitectura de redes que permite desglosar las capas de control y de datos, además de emplear como elemento clave, un controlador de red. Gracias a estas dos características fundamentales las redes resultantes son más flexibles y dinámicas frente a las necesidades del medio.

Con este trabajo se pretende dar una visión más detalla de esta nueva tecnología, conociendo su evolución, así como comprender su funcionamiento y su relación con la tecnología OpenFlow. Para ello se presentará Mininet, un software de emulación de redes basado en la tecnología a estudiar. Centrándose en desarrollar de forma más detallada una de sus herramientas de trabajo, la plataforma Miniedit, la cual presenta una interfaz gráfica idónea para experimentar con los conceptos de las SDN y OpenFlow.

Resum

Com a resposta a la creixent evolució de la demanda i de la necessitat de desenvolupament de les xarxes de comunicació apareix les "Software Difened Network o SDN", una nova i revolucionària visió de l'arquitectura de xarxes que permet desglossar les capes de control i de dades, a més d'emprar com a element clau, un controlador de xarxa. Gràcies a aquestes dues característiques fonamentals les xarxes resultants són més flexibles i dinàmics fronts a les necessitats del medi.

Amb aquest treball es vol donar una visió més detalla d'aquesta nova tecnologia, coneixent la seva evolució, així com comprendre el seu funcionament i la seva relació amb la tecnologia OpenFlow. Per a això es presentarà Mininet, un programari d'emulació de xarxes basat en la tecnologia a estudiar. Centrant-se a desenvolupar de forma més detallada una de les seves eines de treball, la plataforma Miniedit, la qual presenta una interfície gràfica idònia per experimentar amb els conceptes de les SDN i OpenFlow.

Abstract

In response to the growing evolution of the demand and the need for the development of communication networks, there is the “Sofware Differred Network or SDN”, a revolutionary new vision of the network architecture that allows the control and data layers to be broken down, in addition to using as a key element, a network controller. Thanks to these two fundamental characteristics, the resulting networks are more flexible and dynamic compared to the needs of the environment.

This work aims to give a more detailed view of this new technology, knowing its evolution, as well as understanding its operation and its relationship with OpenFlow technology. For this, Mininet will be presented, a network emulation software based on the technology to be studied. Focusing on developing in more detail one of its work tools, the Miniedit platform, which presents an ideal graphical interface to experiment with the concepts of SDN and OpenFlow.

Agradecimientos

El documento no estaría completo si hacer una pequeña mención a mis allegados, que, con su perseverancia y tozudez, no permitieron que me olvidara de la tarea que me quedaba para finalizar mis estudios.

En primer lugar, me gustaría agradecer a mis padres y mi hermana la ayuda que me han brindado durante toca la carrera, por su apoyo y, sobre todo por aportarme vuestro punto de vista, sin el cual no habría conseguido completar esta labor.

Agradecer a María su apoyo incondicional, y sus ánimos incansables, que, sin saberlo, me aportaban la clave para cambiar el punto de vista y con ello hallar la solución.

También agradecer a todos mis amigos que se han preocupado por mí y por el desarrollo del TFG. Convirtiéndose en una pieza clave para superar las trabas del día a día. Haciendo especial mención a Rubén, por su incansable interés por el progreso del trabajo.

Solo resta daros las gracias a todos por aportar vuestro grano de arena, sin el cual no habría llegado hasta aquí.

Índice

Índice de Figuras

Capítulo 1. Introducción

En los tiempos que corren se podría afirmar que vivimos en una sociedad "conectada", ya sea a través del móvil, el ordenador, las video consolas, la TV y un largo etc., ligado a la gran influencia, e incluso adicción, a las redes sociales, tales como Facebook, Instagram y twitter, u otras plataformas de contenido multimedia, como Netflix, HBO, Sky o de Streaming (twitch o Youtobe), o el correo electrónico, que da paso al medio más empleados en el día a día, la mensajería instantánea, ya sea Whatsapp, Line, Hunk, o cualquiera de las múltiples aplicaciones existentes. Siendo, todas las aplicaciones nombradas, una fuente de influencia, además de una plataforma de difusión para millones de anuncios que se envían y reproducen en todo momento, los cuales, en la mayoría de los casos, incitan a un mayor uso de las estas y otras plataformas no citadas.

Con este frenetismo de consumo, resulta evidente la necesidad insaciable, por parte de los usuarios, de una mayor calidad de servicio, capaz de soportar las necesidades demandadas, para lo cual resulta indispensable el desarrollo de sistemas e infraestructuras cada vez más potentes.

Aunque la tecnología que envuelve las redes de comunicación nunca deja de evolucionar, las demandas del mercado tampoco llegan a ser cubiertas. Esto se debe a que tanto el desarrollo como la demanda van cogidas de la mano, creciendo de forma simultánea e imparable.

Todo esto induce a reflexionar sobre ¿qué necesidad es tan insaciable para que la evolución de las comunicaciones no consiga satisfacer la demanda?

Una de las razones con mayor peso, es la necesidad de una comunicación cada vez más rápida, no solo de la transmisión de mensajes o llamadas, si no, del envío de grandes volúmenes de datos de forma "instantánea". Si se tiene presente el contenido en streaming consumido en cualquier lugar, y cada vez con una mayor calidad, Representan algunas de las múltiples causas de esta obsolescencia de las infraestructuras de red.

Con el fin de hacer frente a la situación mencionada, nace una tecnología con capacidad para mejorar la respuesta frente a situaciones adversas, como la saturación, la caída de un equipo o un enlace. Este sistema innovador son las redes inteligentes conocidas como Software Defined Networks (SDN), cuya cualidad más llamativa es su capacidad para hacer frente a estas situaciones sin la intervención directa del administrador del sistema, esto es debido a que basan r su ejecución en un controlador de red.

Para ello resulta fundamental una comunicación o jerarquía piramidal, en cuya cúspide se encuentra el ya mencionado controlador, que emplea múltiples tablas de flujo como herramienta fundamental para la gestión de las redes basadas en SDN.

Capítulo 2. Objetivo del Proyecto

Con el presente proyecto se busca dotar a un público global, de un conocimiento actualizado de la evolución de las arquitecturas de redes y su desarrollo hacia la programabilidad y el dinamismo de su comportamiento, basándose en la necesidad de comprender estos conceptos que se engloban en las redes definidas por software o SDN.

Siguiendo esta premisa se realizará un estudio de las tecnologías que trabajan con este sistema, empleando las redes actuales como modelo de referencia, pero centrando el objetivo en los sistemas de emulación, que permitirán adquirir una mayor comprensión de las SDN.

Por tanto, en este proyecto se van a desarrollar diferentes ideas, tales como una breve reflexión de las redes actuales, el aprendizaje del funcionamiento de la plataforma de trabajo conocida como MiniNet, la cual emplea la tecnología SDN, en la cual se prestara un mayor atención en la comprensión del manejo de su plataforma gráfica, MiniEdit. Ambas se describirán en mayor detalle a lo largo del presente documento.

En definitiva, el desarrollo del trabajo se centrará en desarrollar:

- Los principales inconvenientes de las redes actuales
- Comprender como las SDN pueden mejorar o resolver las desventajas de los sistemas actuales.
- La relación entre las SDN y Open Flow.
- Los conceptos que abarcan la tecnología del Open Flow, en la que se basa la aplicación sujeta a estudio, Mininet.
- Conocer las funciones básicas de la plataforma Mininet.
- Explicar el entorno de trabajo grafico Miniedit, y comprobar el correcto funcionamiento de las funciones de comunicación por medio de una simulación.

Con todo ello, y en especial, a través de diversas simulaciones se pretende conocer las opciones que nos presenta esta interfaz gráfica. Además de servir como prueba para comprobar la correcta configuración de los equipos y la interconexión entre Mininet y Miniedit.

Capítulo 3. Metodología

Tratando de optimizar el tiempo invertido en la elaboración del documento, se ha distribuido el contenido en tres grandes bloques de trabajo desempeñados de forma consecuencial, en la medida de lo posible, permitiendo compaginar la elaboración con la vida laboral.

Durante el desarrollo del primer bloque se focaliza la atención en la recogida de información y conocimientos relacionados con las redes definidas por software, así como de una primera toma de contacto con el emulador Mininet.

Una vez desarrollada la etapa anterior, se persigue un esquema teórica de los sistemas estudiados. De manera que ayuden al lector a comprender los conceptos desarrollados en los primeros capítulos del documento, buscando explicar de forma clara y sencilla esta nueva tecnología.

El ultimo bloque de trabajo se centra en conocer y explicar el funcionamiento de Mininet, dando un mayor peso a la presentación y explicación de su entorno de trabajo gráfico, Miniedit. Para ello, una vez adquirido los conocimientos básicos para el manejo de la herramienta, se realizarán diversas simulaciones en las que se pone a prueba algunas de las características de configuración y emulación presentes en dicha interfaz.

Teniendo en cuenta el planteamiento inicial y el desempeño real del trabajo, se ha elaborado el siguiente flujo de tiempos.

Actividad a desarrollar	Inicio	Duración	Fin
Primer bloque de trabajo:	26-jul	11	06-ago
Metodología	26-jul	2	28-jul
Recogida de información	26-jul	15	10-ago
Primera toma de contacto con el entorno Mininet	31-jul	2	02-ago
Objetivos del proyecto	03-ago	3	06-ago
Segundo bloque de trabajo:	07-ago	20	27-ago
Introducción	07-ago	3	10-ago
Introducción de las redes actuales	10-ago	4	14-ago
Estudio teorico de las SDN	13-ago	6	19-ago
Comparativa entre las R. Actuales y las SDN	17-ago	4	21-ago
OpenFlow	20-ago	3	23-ago
Mininet estudio teorico	22-ago	4	26-ago
Miniedit estudio teorico	23-ago	4	27-ago
Tercer bloque de trabajo:	26-ago	13	08-sep
Puesta en marcha del entorno de trabajo	26-ago	7	02-sep
Analisis del entorno de trabajo Mininet	01-sep	4	05-sep
Analisis del entorno de trabajo Miniedit	03-sep	3	06-sep
Pruebas de las funcionalidades del entorno grafico	05-sep	3	08-sep
Conclusiones	07-sep	1	08-sep

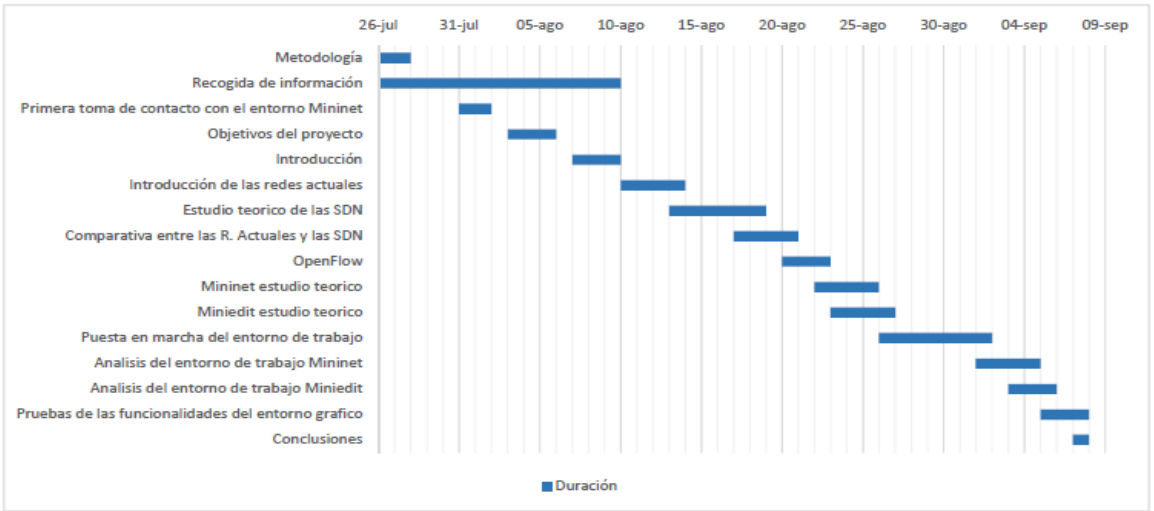


Figura 1. Diagrama de flujos (elaboración propia).

Capítulo 4. Redes Actuales

Las redes han evolucionado desde sus orígenes, desde la red más básica, que únicamente redireccionaba paquetes según su cabecera para alcanzar su destino; hasta llegar a las redes actuales más "inteligentes", que cuentan con una capacidad de gestión del tráfico de datos, fragmentación, etc., esta tecnología queda recogida bajo el nombre de redes activas.

Las redes activas dejan de ser insensibles a la información transmitida, pues constituyen una arquitectura de red en a que los nodos pueden realizar un procesamiento individual sobre los paquetes que transmiten, lo que se ha conseguido mediante la programación de dichos nodos intermedios, alcanzando un procesamiento a medida. Además, presentan una peculiaridad, permiten su configuración en línea, dando un paso hacia adelante para lograr una modificación dinámica del comportamiento de la red [1].

Teniendo en cuenta todo lo anterior, resulta evidente el aumento en las posibilidades de envió de grandes volúmenes de datos, más pesados, que son fragmentados en sub-paquetes más pequeños y posteriormente reconstruidos en el destino, de forma que se pueden adaptar con mayor facilidad a las configuraciones del medio, facilitando su transmisión.

Visto así, se podría considerar que son todo ventajas, puesto que él envió resulta más sencillo, rápido y con un mayor aprovechamiento del ancho de banda, pero este sistema presenta un gran inconveniente, un aumento en el procesado necesario para llevar a cabo la transmisión, puesto que es necesario que todos los fragmentos del paquete inicial cuenten con una cabecera con la misma información que la original, además de incluir la necesaria para su recomposición en el destino, como por ejemplo, el orden de fragmentación y envío.

Como resultado, se pueden producir retardos en la transmisión, derivadas de la falta de capacidad de procesado, por las pérdidas de información que dificulten o imposibiliten la recomposición del archivo original; o por la saturación del medio de transmisión.

Una de las causas que provocan dicha saturación es la resultante de la evolución imparable de las nuevas tecnologías y su cada vez mayor demanda La infinidad de conexiones y transmisiones que se producen cada segundo, por parte de millones de usuarios, resultan ser las principales causantes de acumulaciones de transmisiones, ya sea en un enlace o en algún nodo de la red.

Pese a estos inconvenientes, y como ya se ha mencionado, la optimización de los recursos resultante de estas técnicas de trasmisión, son un beneficio real frente a los sistemas predecesores.

Llegados a este punto se presenta una cuestión ¿Cómo se podrían minimizar estas desventajas? La respuesta se encuentra en la gestión de redes y su amplio abanico de protocolos de transmisión y gestión, que permiten al administrador de la red decidir el encaminamiento de la información y dar prioridades a unos flujos de datos frente a otros, de forma que pueda mantener y garantizar una calidad del servicio, con una mayor dinamización y aprovechamiento de los recursos, así como el aumento de la velocidad de procesado y transmisión.

Pese a la gran evolución del sector en este último ámbito, el aumento de velocidad por sí mismo no es suficiente para hacer frente a las demandas actuales, tampoco resulta útil como herramienta fundamental para afrontar los problemas a los que se exponen las comunicaciones, tales como caídas o fallos de transmisión.

Aun con todas las medidas tomadas por parte del administrador de red, ya sean los protocolos empleados, las rutas alternativas programadas, la configuración especifica de cada nodo, entre otras posibles medidas, pueden darse casos no previstos, o incluso, en los que las medidas tomadas puedan ser contraproducentes o ineficaces.

Haciendo hincapié en esta posibilidad se plantea un caso práctico en el que se podría dar dicha situación:

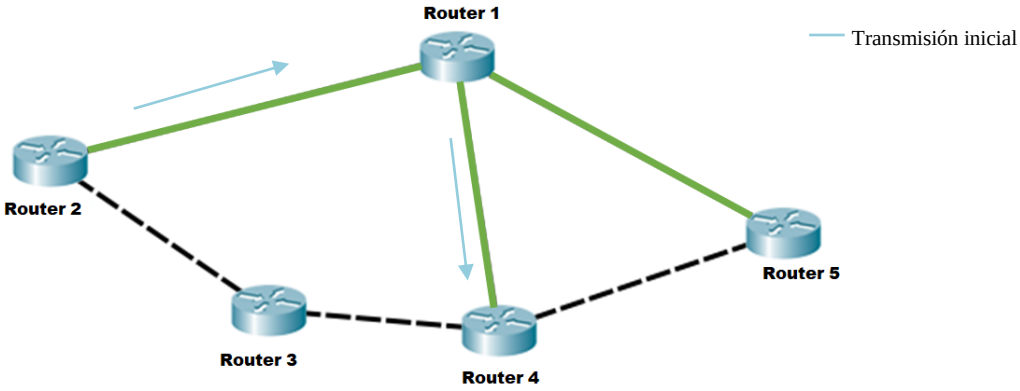


Figura 2. Red sencilla sin controlador (elaboración propia).

En la Figura 2 se aprecia una red sencilla en la que se pueden observar distintas velocidades y calidades de enlace entre los módulos, siendo las marcadas en verde las más rápidas y fiables, mientras que las discontinuas son las que presentan una mayor cantidad de retardos y pérdidas.

Tomando como punto de partida el Router 2 (R2), que realiza una comunicación, indicada por medio de flechas azules, con el Router 4 (R4). Tras tener en cuenta las configuraciones implementadas y teniendo en consideración la calidad de los enlaces, resulta evidente la ruta seguida del flujo de datos entre ambos, será a través del Router 1 (R1), que, aún presentando el mismo número de saltos que la ruta por el Router 3 (R3), la velocidad de los enlaces es superior y por tanto más eficiente.

En caso de que se produzca un fallo en la red causante de una caída en el enlace entre R1 y R4, obligará a redireccionar todos los paquetes que se transmiten entre R2 y R4. Llegados a este punto se observan dos alternativas, emplear el camino más corto o emplear el camino con mejores propiedades.

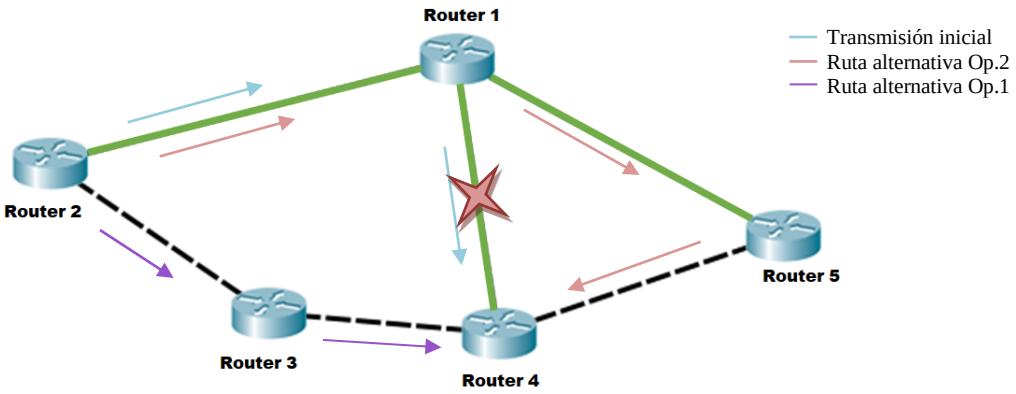


Figura 3. Red sencilla sin controlador, fallo de un enlace (elaboración propia).

Tras la detección del fallo entre los R1 y R4, las tablas de encaminamiento se actualizan y se recalculan sus valores asociados, tales como número de saltos, coste de la transmisión o la puerta de enlace, como resultado de este hecho, pueden ocurrir dos cosas: que se redirija la comunicación por el enlace alternativo (Op.1), o que continúe por la misma ruta (Op.2), en ambos casos la comunicación se puede verse afectada, ya sea por el aumento de los tiempos de transmisión, las

pérdidas que sufren los enlaces más lentos, o por una saturación de la ruta más rápida, aunque más distante.

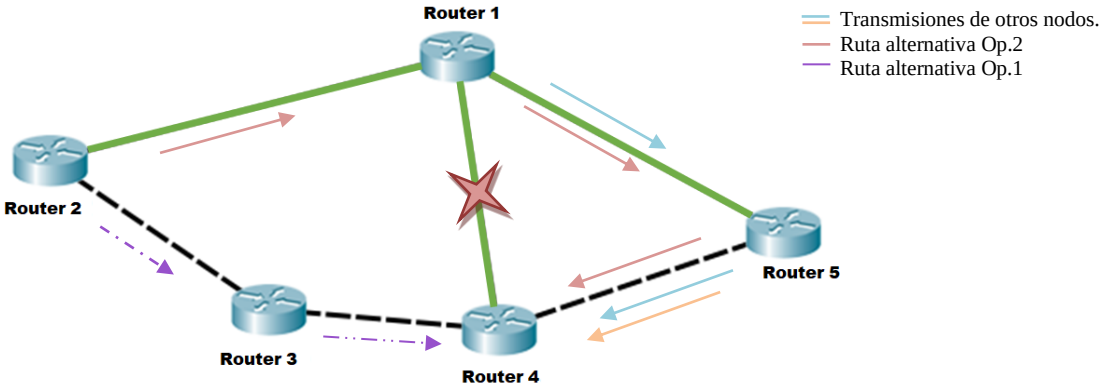


Figura 4. Red sencilla sin controlador, alternativas de transmisión (elaboración propia).

La ruta seguida por la transmisión, quedará definida por el protocolo y la configuración establecida en los equipos que conforman la red, pese a ello como se ha comentado, ambas opciones presentan condiciones desfavorables que pueden afectar a la transmisión y, por consiguiente, de forma directa a la calidad del servicio, o en ingles Quality of Service (QoS), que se ofrece. Frente a este escenario, y una vez detectada la situación de la red, el administrador de la red, el administrador podría intervenir directamente, con el fin de adaptar las transmisiones, tratando de mantener las características de la transmisión inicial.

Analizando los resultados derivados de la hipotética situación anterior, se aprecia claramente la rigidez de la red y por consiguiente, la lentitud de adaptación del sistema frente a situaciones adversas o imprevistas.

Ante este panorama, surge una alternativa innovadora y en constante evolución, las redes SDN, las cuales son mucho más eficientes, puesto que, al contar con un control centralizado, que analiza de forma constante la red y toma las medidas oportunas, retransmitiendo la nueva configuración a los diferentes nodos, pudiendo adaptar la configuración de la red de forma más eficiente y en menor tiempo, permitiendo así, alcanzar el objetivo de mantener la red optimizada y con una mayor calidad de servicio.

Llegados a este punto se plantean dos frentes a desarrollar, ¿Qué son las SDN? y ¿Cómo consiguen la dinamización de la red, y con ello una mayor eficiencia?

Capítulo 5. Software Defined Networks

Las redes definidas por software o SDN, agrupan un conjunto de técnicas asociadas a las transmisiones de información, que persiguen mejorar su rendimiento por medio de la simplificación, la implantación e implementación de los servicios de redes de forma más concreta, dinámica y escalable. Permitiendo una gestión, por parte del administrador, a un alto nivel de programación.

La ONF (Open Networking Foundation) [2], las define como una arquitectura de red dinámica, gestionable y adaptable, con un coste eficiente. Lo cual las hace idóneas para las altas demandas de ancho de banda y la naturaleza dinámica de las aplicaciones actuales.

Mediante la desaplicación del control de la red y la funcionalidad de reenvío de información, se consigue que el control de la red sea completamente programable, logrando la abstracción de las aplicaciones y servicios de red de la infraestructura de la red subyacente.

Este revolucionario concepto no es tan actual como se podría pensar en una primera toma de contacto. La idea de poder programar las redes nació con la aparición de internet, en torno a 1985, ya entonces se consideraba internet como una tecnología establecida, con una gran aceptación en la sociedad. Resultando evidente la necesidad de una gestión y evolución constantes, enfocadas a su capacidad de ser programables. Con esta idea en mente se pueden destacar tres fases o etapas en el desarrollo de esta tecnología:

1 – Aparición de las redes activas (1995 – 2000 aprox.).

2 – Separación de los planos de control y de datos, base de las SDN (2001 – 2007 aprox.).

3 – Desarrollo de Openflow, aparece la interfaz de programación (2007 – 2010 aprox.).

Estas últimas, las Interfaz de programación, o APIs (“ApplicationProgramming Interface”), son un conjunto de subrutinas, funciones y procedimientos que ofrece una biblioteca para ser utilizada por otro software como una capa de abstracción, o una forma de ocultar los detalles de implementación de diversas funcionalidades [3].

Cabe destacar que actualmente se considera a la tecnología de Openflow como la base del SDN, puesto que, una parte del datapath reside en el propio switch, mientras que el encaminamiento es manejado por un controlador. Pese a ello ambas partes se deben de poder comunicar de forma constante, lo cual se realiza por medio de la tecnología OpenFlow. Cabe destacar que, pese a ser el protocolo más empleado, no es el único que se puede emplear en las SDN; más adelante se desglosa con más detalle esta y las otras alternativas.

5.1 Estructura de las SDN

A causa de la masificación de dispositivos inteligentes disponibles en el mercado, según datos de la GSMA [4], se han producido 5.1 billones de suscriciones de servicios móviles a finales del 2018, aproximadamente el 67% de la población global. Lo que supone un aumento de un billón de usuarios en cinco años. Cabe comentar que se estiman 700 millones de nuevas suscripciones en los próximos años. A esto se suma a la integración de los servicios en la nube o Cloud Services,

y la virtualización de servicios, siendo solo algunas de las tendencias actuales que fuerzan una renovación o reestructuración de las arquitecturas tradicionales.

La GSMA, es una asociación de operadores móviles y compañías relacionadas, que se dedican a dar apoyo a la normalización, la implementación y la promoción del sistema de telefonía móvil. Además de realizar un informe anual de la evolución de estos sistemas a nivel global y nacional.

Las SDN presentan una estructuración basada en un diseño jerárquico, semejante a un trabajo entre cliente y servidor, estando el controlador de red por encima de los nodos que la conforman. Por el contrario, las arquitecturas habituales tienen limitada su capacidad de adaptación, puesto que cada nodo de la red actúa de forma individual, cuentan con una programación y una capacidad de decisión individual, perjudicando su capacidad de adaptación frente a las necesidades de los centros de procesado de datos.

Un de las características que posibilitan la mejora del rendimiento y la adaptabilidad de las redes definidas por software, es la división entre el plano de control, formado por la capa de control y la capa de aplicaciones; y el plano de datos, formada por la capa de infraestructura.

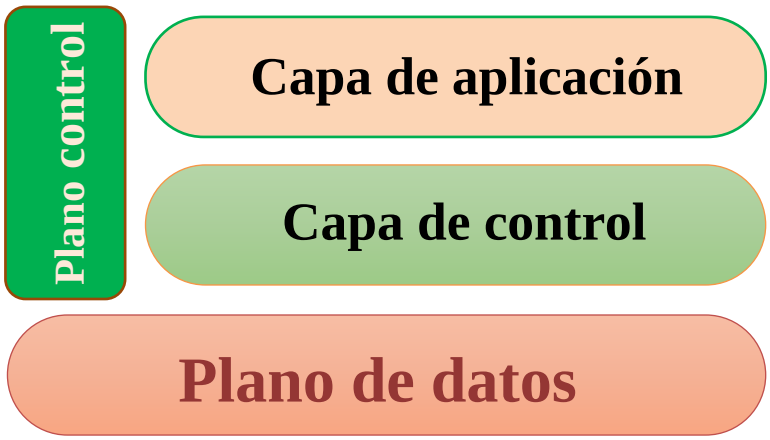


Figura 5. Estructura SDN (elaboración propia).

Teniendo clara la estructura básica de las SDN, se puede apreciar en mayor medida la distribución de su arquitectura presente en la siguiente figura:

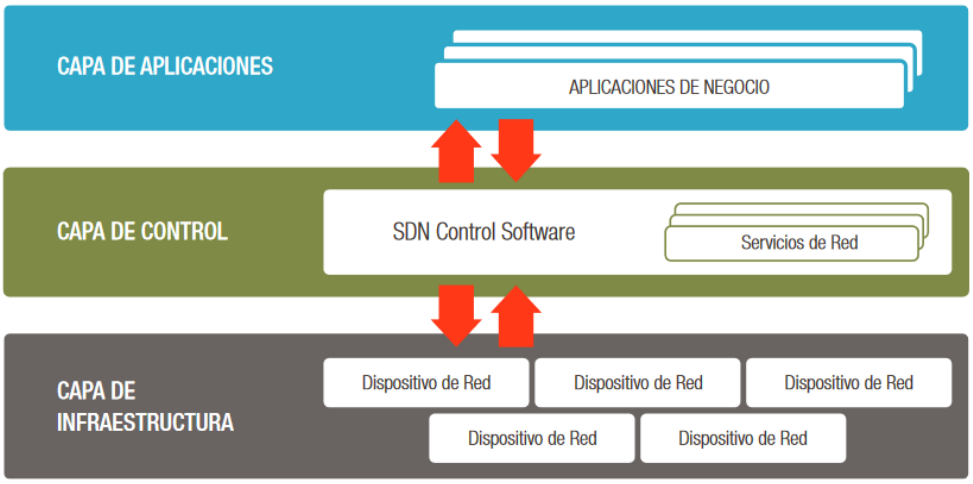


Figura 6. Arquitectura SDN [14].

Con la separación de los planos de control y de datos, se debe de establecer una comunicación entre ambos, para ello se emplea una interfaz API, que permite al controlador el control de los elementos del plano de datos. A través de esta distinción entre planos, se obtiene la capacidad de realizar una configuración remota y dinámica de todos los elementos de la red, además de proporcionar una visión global que permite realizar actuaciones más precisas.

Al dotar al controlador de red con estas funciones, se elimina la necesidad de una configuración individual y de forma local de los diferentes nodos.

Las diferentes capas, que forman los planos de control y datos, se distribuyen de la siguiente manera:

El ***plano de datos*** o ***capa de infraestructura***, está formada por los elementos de la red, tales como routers, host, switches o medios para la comunicación, encargados de la transmisión de los paquetes entre nodos. En otras palabras, se puede afirmar que es el esqueleto de toda la infraestructura.

La ***capa de control***, formada por el o los controladores de red quienes, empleando su visión global de la infraestructura, toman las decisiones para gestionar y adaptar la red. Para poder llevar a cabo estas adaptaciones se deben de configurar un componente básico en los elementos de la capa inferior, las tablas de flujo {3}. Formando así la musculatura del sistema.

Por último, la ***capa de aplicaciones*** [5], está formada por un conjunto de programas que se comunican directamente con los controladores, con el fin de transmitir sus necesidades y los comportamientos deseados en la red.

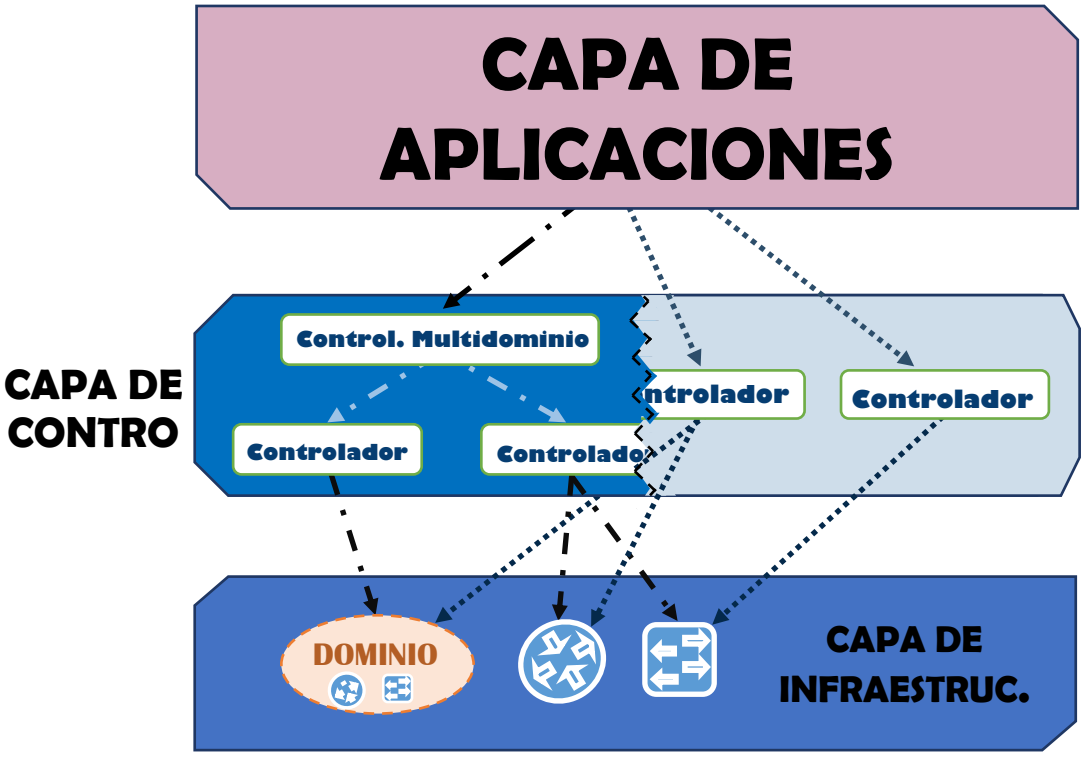


Figura 7. Esquema resumen funcionamiento SDN (elaboración propia).

En la Figura 6 se puede apreciar un esquema del funcionamiento de una red basada en SDN, la cual comienza con las peticiones de los usuarios, recogidas en la capa de aplicaciones, son transmitidas a los controladores individuales o en su defecto a controlador central, el cual transmite estos requisitos al resto de controladores; y finalmente comunicando la configuración pertinente a los componentes de la capa de infraestructura.

Las Tablas de flujo {3} o "Flow Tables", mencionadas con anterioridad, son la estructura básica para la comunicación, formadas por un conjunto de campos con las que se coteja la información de los paquetes entrantes en el dispositivo, como por ejemplo su origen o su destino; además de las acciones que se han de llevar a cabo con cada flujo de datos. Están estructuradas con el fin de abarcar la máxima información, de forma concreta, para procesar los paquetes entrantes, para ello cabe la posibilidad de trabajar con múltiples tablas o "Group Tables".

Conociendo el funcionamiento global de este sistema resulta pertinente analizar en profundidad el funcionamiento de cada uno de los niveles que conforman esta estructura.

5.1.1 Capa de Aplicación:

Como ya se ha mencionado, es empleada por las aplicaciones para transmitir sus necesidades o requisitos a la red, mediante una conexión con la capa de control a través de una API.

Algunos ejemplos de aplicaciones serían:

- Enrutamiento adaptativo, basado en el balance de cargas o en el intercambio de información entre capas.
- Simplificación del mantenimiento de la red, puesto que, al tener una visión global, un control centralizado de la misma y pudiendo comunicarse con los nodos que la conforman, resulta sencillo llevar a cabo las tareas de mantenimiento de la red.
- Incremento de la seguridad de la red, debido a que las SDN son capaces de analizar patrones de tráfico, pueden identificar con facilidad fallos de seguridad como sería un ataque por denegación de servicio, pudiendo actuar en consecuencia al realizar un re direccionamiento de paquetes a los IPS, o Sistemas de Prevención de Intrusión, o incluso, dar prioridad a usuarios. [6]

5.1.2 Capa de control

Siendo el componente básico el controlador de la red, el cual, no tiendo por qué ser único, siendo conveniente descentralizar su control sobre la red por medio de diversos equipos. Debido a su gran importancia que se deriva de ser el encargado del centralizado de la lógica que permite la gestión y, por tanto, de la capacidad de controlar el comportamiento de la red, puede ser objeto de ataques.

Este controlador se comunica con el resto de elementos de manera constante, realizando una interpretación de los requisitos de dichos componentes por medio de lenguajes de alto nivel como java, Python o C++, además, también debe de mantener una comunicación con las aplicaciones que hacen uso de la red.

Al disponer de toda la información referente al estado de la red y los requisitos de los usuarios, puede llevar a cabo diversas implementaciones, como, por ejemplo, una representación de los flujos de comunicación entre origen y destino, también llamadas TM o Matriz de Tráfico.

Siguiendo con el hilo inicial, es necesaria la disgregación de la información, por medio de controladores de backup o de apoyo, con esto se consigue aumentar la seguridad de la red, ya que un solo controlador supone un punto crítico frente a ataques o congestiones de la red. Además, el uso de un segundo controlador al distribuir la carga de trabajo, simplifica las labores de gestión, solventando una posible limitación de escalabilidad frente a redes amplias o de considerado tamaño.

Un ejemplo de la implementación de diversos controladores, es ***HyperFlow*** [7], que es un plano de control basado en eventos de distribución para OpenFlow, permitiendo desplegar cualquier número de controladores en una red. Proporcionando una gran escalabilidad manteniendo el control de la red centralizado, que comparten una visión global de la red entre todos los controladores, pero aplicando de forma local las configuraciones o configuraciones pertinentes Con ello se evita contactar de forma activa con nodos remotos, consiguiendo una disminución de los tiempos de configuración.

Una de las grandes ventajas que este sistema presenta es el hecho de ser resistente a las particiones de la red y a los fallos de componentes, por lo que garantiza el reenvío sin bucles.

En esta figura se puede apreciar un ejemplo de la aplicación de HyperFlow a una red OpenFlow multisitio con controles únicos y múltiples. Cabe aclarar que las asociaciones entre los controladores y los conmutadores se representan por colores.

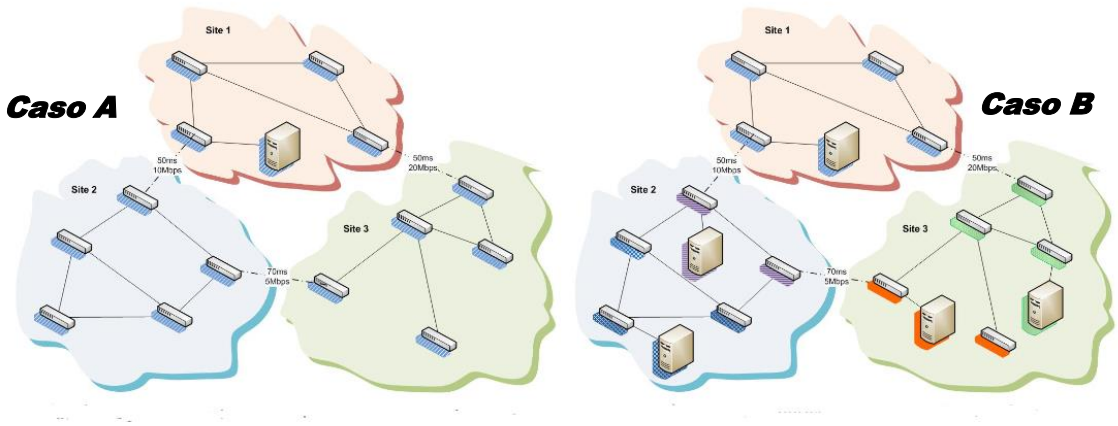


Figura 8. Ejemplo HyperFlow [7].

En la primera imagen o Caso A, se observa como la implementación de un solo controlador aumenta el tiempo de configuración, además de un aumento de las tasas de envió de flujos entre las zonas remotas, lo cual puede causar una congestión de los enlaces entre zonas.

Mientras que, en el segundo caso, Caso B, todas las peticiones son atendidas por controladores locales, minimizando el tráfico entre zonas, actualizando la información de los controladores por medio de sus vecinos. De esta forma se evita una posible saturación de los enlaces de interconexión entre áreas distintas.

Otro ejemplo sería ***FlowVisor [8]***, aunque su objetivo final sea semejante al anterior, no trabaja igual que HyperFlow. Este sistema busca la habilitación de múltiples controladores dentro de una red, pero a diferencia del otro sistema, lo realiza mediante la segmentación de la red y la delegación del control de cada segmento a un controlador de forma individualizada. Por lo que en lugar de tener una única red con múltiples dispositivos de controlador, se obtendrían múltiples áreas de gestión con un único controlador intercomunicados entre si.

5.1.3 Capa de infraestructura:

Esta capa está formada por todos los dispositivos hardware que forman parte de la red, los cuales realizan dos tareas, una de control, empleada para la comunicación con el controlador, informando a este del estado de la red, mediante esta información realiza la configuración del sistema; y otra de datos, redirigiendo el tráfico en función de las decisiones tomadas en el plano de control.

Como ya se ha comentado, los elementos básicos para esta capa son las tablas de flujo o Flow Tables, cuyo funcionamiento se puede resumir en la siguiente imagen, donde se recoge el comportamiento de los diferentes flujos tras acceder por primera vez a una tabla:

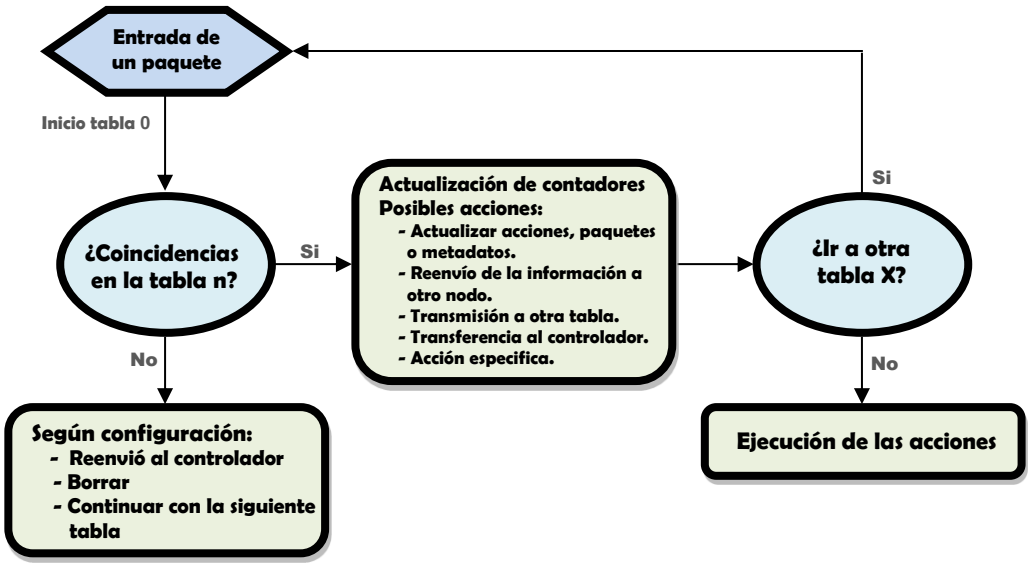


Figura 9. Diagrama de flujo del tratamiento de un paquete (elaboración propia).

Como se aprecia en a la imagen cada paquete que accede a una tabla es comparado con la tabla correspondiente con mayor nivel de aplicación, es decir, se emplean las tablas en orden ascendente.

Una vez fue comparado y en función del resultado obtenido al comparar los valores pertinentes en los campos de la tabla, se llevarán a cabo las medidas recogidas en su campo de acción, pudiendo ser el caso de que el nodo reenvíe el paquete a su controlador, quien lo analiza y procesa con el fin de actualizar las entradas de las tablas de los nodos, creando su ruta de transmisión.

Otra de las acciones posibles es el reenvió normal por el puerto correspondiente o incluso la eliminación del paquete, ya sea por no hallar ninguna concordancia, y por configuración se elimine; o por ser la acción configurada.

En caso de que se haya enviado al controlador y tras actualizar los datos de las tablas, al recibir nuevos paquetes del flujo mencionado, se retransmiten de acuerdo a la información transmitida por él controlador, simplificando y reduciendo el coste y tiempo de transmisión de los distintos paquetes.

Capítulo 6. Comparación redes actuales y SDN

Tras analizar lo tratado en los apartados anteriores se puede apreciar las grandes ventajas e innovaciones que supone la implantación de un sistema basado en SDN, por ejemplo, frente a una situación semejante a la vista durante el apartado de redes actuales, un controlador respondería con mayor velocidad, siendo una posible ejecución la siguiente:

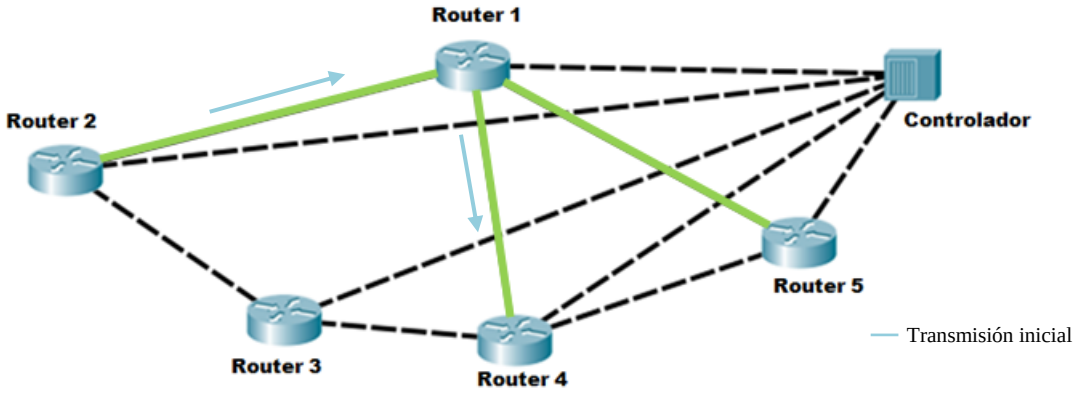


Figura 10. Red sencilla con controlador (elaboración propia).

El planteamiento inicial es el mismo, la red sencilla funcionando correctamente con sus diferentes tablas de flujo configuradas.

Durante la comunicación entre dos nodos se produce la caída de un enlace. Tras la detección de la pérdida de comunicación, se comunica el evento al controlador que analiza y elabora las medidas pertinentes:

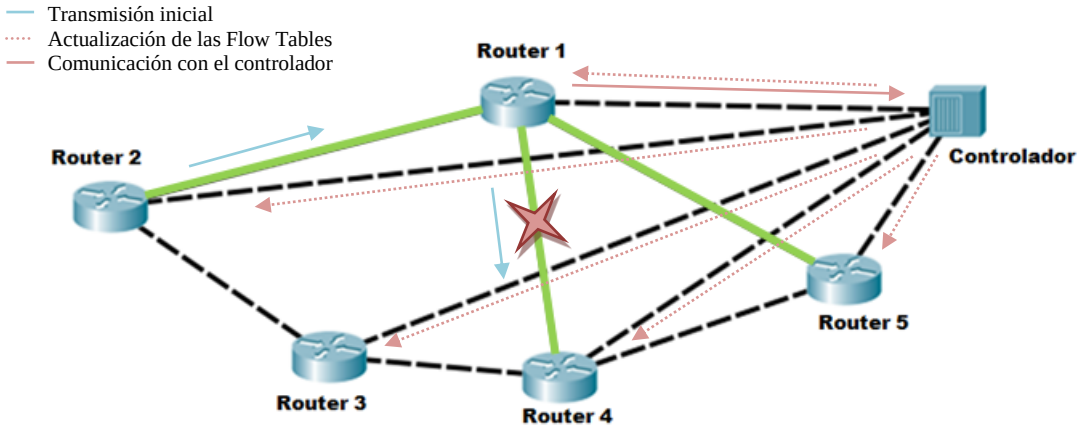


Figura 11. Detección de la caída y actualización de las tablas (elaboración propia).

Una vez el controlador sintetiza las medidas para afrontar el fallo, estas son transmitidas por la capa de control, actualizando las tablas de los elementos de la red, redirigiendo el flujo de datos.

La actualización de las tablas puede deberse al detectar o, en algunos casos, predecir la saturación de un enlace, ya sea por una gran pérdida de QoS o por el incremento del número de paquetes. El controlador tiene la capacidad de reaccionar en consecuencia redistribuyendo

el tráfico para tratar de suplir las pérdidas, intentando mantener la calidad de servicio y/o los tiempos de transmisión establecidos, en la medida de lo posible.

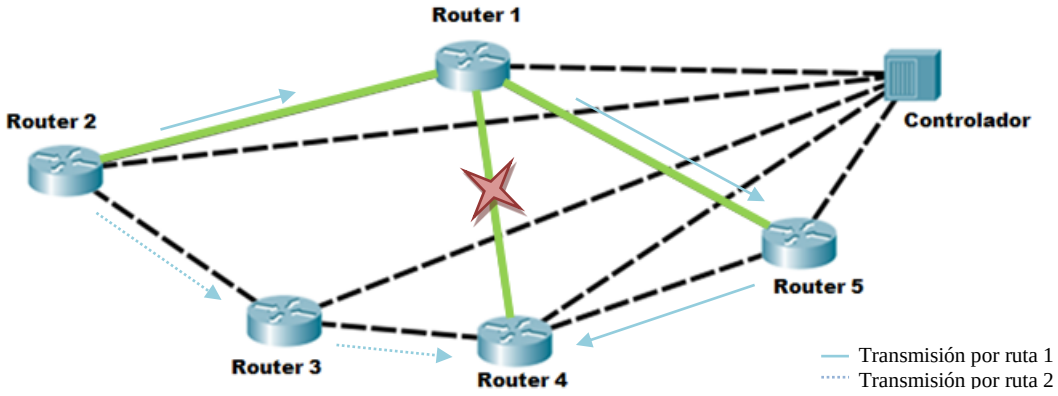


Figura 12. Adaptación de la transmisión según prioridad (elaboración propia).

Una vez restablecida la correcta comunicación por medio de los demás enlaces, cabe la posibilidad de que el enlace caído se restableciera, ya sea por la reparación del mismo o porque, tras sufrir la caída sea capaz de reiniciarse por sí mismo, restableciendo su servicio. Frente a esta situación el controlador, que se encuentra analizando el estado de la red, volvería a restablecer la comunicación inicial, realizando el mismo procedimiento que tras detectar la caída del enlace.

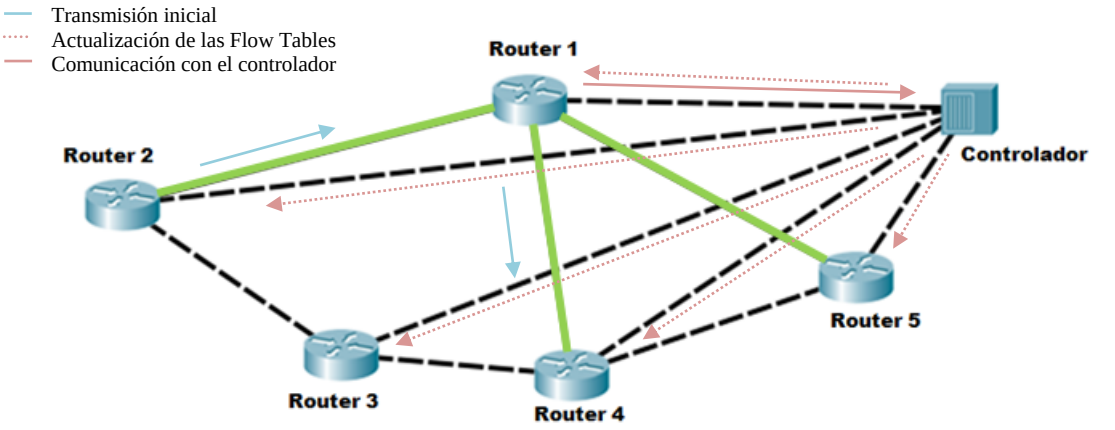


Figura 13. Restauración del enlace y actualización de las tablas (elaboración propia).

Resulta apreciable que esta sería una de las características por las que los sistemas basados en esta tecnología resultan ser muy superiores a los actuales, sin olvidar la gran escalabilidad y la simplificación de la gestión de redes muy amplias.

Teniendo presente estas ventajas surge una duda ¿Por qué no se está implantando a nivel global de forma casi inmediata? La respuesta se encuentra en el nivel de desarrollo de la tecnología, al ser una tecnología relativamente nueva a día de hoy se sigue investigando, por el contrario partida no todas las redes son capaces de soportar esta evolución, además del

impacto económico que supone la inversión para adaptarlas a este nuevo sistema. Todo ello supone un freno para su implantación.

Pese a su gran desarrollo, presenta algunas vulnerabilidades que pulir antes de ser implantadas en una red global. Una de estas limitaciones ya ha sido mencionada, la cual representa el talón de Aquiles de esta tecnología, el propio controlar de red, ya que al ser el cerebro de toda la red es susceptible de ser el foco de los ataques destinados a la misma, pudiendo llegar a comprometer toda la estructura.

A pesar de esto, se conocen casos de aplicaciones reales por parte de numerosas empresas, unos ejemplos de ello serian [9]:

- Nicira, o VMware, encontró la vitalización de red como forma de agilizar la provisión y operación de entornos masivos, siendo su objetivo actual, llevar el firewall hasta la máquina virtual en escenarios donde la seguridad es crítica.
- Nuage (Alcatel), enfocada en llevar las SDN a las redes WAN. Persigue el objetivo de conseguir que las redes privadas, de los operadores de telecomunicaciones, se conviertan en entornos dinámicos.
- HP ha apostado por adaptar los servicios de red de aplicaciones de mensajería, adaptando de forma dinámica sus requisitos de calidad o la segmentación de redes físicas en virtuales o “slices”.

Estos son algunos ejemplos de las aplicaciones reales de las SDN llevados al mercado actual, pero no son los únicos, puesto que empresas como Alcatel-Lucent ofrecen que sus productos, como OmniSwitch, por medio de una actualización gratuita cuenten con soporte para OpenFlow, proporcionando una interfaz de trabajo SDN. Lo que supone un gran avance para su implementación.

Capítulo 7. OpenFlow

Este protocolo emergente, y de código abierto, surgió en 2008 por el proyecto "OpenFlow: Enabling Innovation in Campus Networks" en la universidad de Stanford. Este protocolo busca una centralización del control de paquetes, tomando forma el concepto de la red como un conjunto, lo que deja de lado el concepto de los conmutadores individuales.

En los equipos convencionales el enrutamiento y la transferencia ocurren en el mismo dispositivo, es decir cada equipo elabora sus propias tablas de encaminamiento con las que guían los paquetes hasta su destino. Empleando la tecnología de OpenFlow se separa la trayectoria de los datos (Datapath) con el encaminamiento a alto nivel (Control path). Tomando como referencia un switch convencional, en el que ambos procesos se realizan en el propio dispositivo, al emplear este protocolo, una parte del Datapath reside en el mismo, pero es un controlador el encargado del control path, centralizando las decisiones que impliquen la transmisión de paquetes. De esta forma se consigue la programabilidad de la red; ambas partes se comunican entre sí por medio del protocolo ya mencionado (OpenFlow).

Resulta evidente la similitud con las SDN, pero hay que aclarar que no son lo mismo, aunque se le considera la primera interfaz estándar, este es solo uno de los protocolos empelados en la metodología de las redes programables, dicho protocolo le aporta una mayor eficiencia en la gestión de los recursos. [10, 11]

Otros ejemplos de diferentes protocolos que luchan por desbancar la fama al estudiado son [12]:

- Border Gateway Protocol (BGP), es un protocolo empleado para la comunicación entre hosts de gateways en una red de sistemas autónomos. También se le clasifica como un protocolo de ruta vertical o de vector distancia.
- NETCONF, es un protocolo de gestión de red Internet Engineering Task Force (IETF). Basada en la llamada de procedimiento remoto o RPC, proporciona una forma segura para configurar cualquier dispositivo de la red.
- Protocolo de Gestión de Base de Datos Open vSwitch (OVSDB), protocolo de configuración destinado a administrar las implantaciones de un controlador virtual con capacidad para la automatización de la red y el soporte de interfaces y protocolos de administración, conocido como Open vSwitch.
- Perfil de Transporte MPLS (MPLS-TP), o perfil de transporte para la comunicación de etiquetas multiprotocolo, está diseñado para ser utilizado como una tecnología de capa de red en redes de transporte.

Existen al menos dos elementos básicos para el funcionamiento de esta tecnología, las tablas de flujos o Flow Tables (FT) y el canal seguro. En dichas tablas se recoge toda la información necesaria para la transmisión de los datos, pudiendo contener información adicional, como por ejemplo un contador para estadísticas.

El canal seguro resulta ser el elemento clave para el correcto funcionamiento del sistema, puesto que sirve de enlace entre el nodo y el controlador, empleado para la transmisión de información y comandos entre ambos puntos.

El funcionamiento de las FT está basado en una comparativa del paquete recibido con los campos de la tabla y por prioridad de ejecución y realizando las comparativas de forma descendente hasta encontrar una coincidencia. En caso de no haber coincidencias el paquete sería descartado por el dispositivo, con el fin de evitar posibles pérdidas de información, se debe configurar una acción

por defecto, pudiendo ser, por ejemplo, él envió del paquete desconocido al controlador para la actualización de las tablas de flujo.

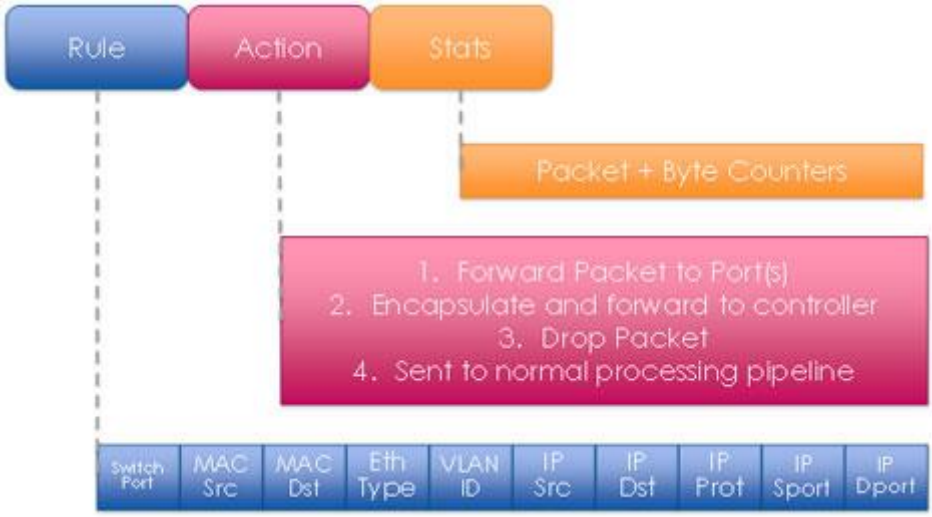


Figura 14. Campos de una Flow Table [13].

Como se puede apreciar en la imagen los campos básicos de la tabla serían la ruta y la acción a realizar, además de estos hay diversos campos, pudiendo ser citados como principales:

- ***Rule***, donde se identifica el flujo, ya sea por el destino, el origen, el puerto por el que se transmite, etc.
- ***Action***, recoge que se debe hacer con los paquetes de dicho flujo, algunas de las acciones posibles son las que podemos observar en la imagen: reenviar el paquete por un puerto, encapsularlo y enviarlo al controlador, borrar el paquete o enviarlo por un enlace con procesamiento normal.
- ***Stats***, donde en función del paquete y diversos contadores, se recoge información útil para realizar estadísticas.
- ***Priority***, empleado para clasificar el flujo de datos, este campo es empleado para garantizar calidad de servicio o QoS.
- ***Timeout***, o "vida" del flujo de datos antes de ser descartado por el dispositivo.

Las Flow Table pueden aplicarse tanto a flujos individuales como a un conjunto, estas tablas de grupos, permiten implementar un conjunto de acciones a un grupo determinado, para ello cuenta con dos campos principales, el identificador del grupo y el tipo del mismo, no menos importante continúa siendo el campo de acciones a implementar al conjunto. De forma opcional también se puede implementar el campo de estadística, pudiendo ser empleado para conocer el número de paquetes procesas, además de otras funciones estadísticas.

Cabe destacar que un dispositivo puede tener un conjunto de tablas de flujo, en las que se recogen todas las acciones a realizar, en el caso de que un paquete llegue a un dispositivo con esta configuración, se realizara el mismo procedimiento, se compara con las entradas de tabla con mayor prioridad, en caso de no hallar coincidencias se repetirá el procedimiento con las demás, siguiendo el orden de prioridad entre ellas, hasta obtener afinidad con una de las entradas de una

flow table, en caso de no encontrar ninguna coincidencia, y en función de su configuración, se descartará o se enviará al controlar para poder crear una nueva entrada para dicho flujo.

Sirva la imagen siguiente como aclaratorio del procedimiento seguido por un Switch con un grupo de tablas:

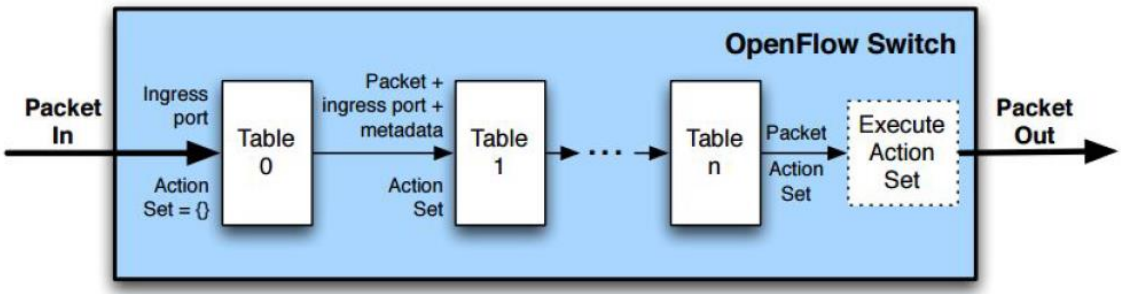


Figura 15. Funcionamiento grupo de tablas de flujo [14].

Capítulo 8. Mininet

8.1 Introdución Mininet

Hoy en día resulta casi imprescindible realizar una simulación previa de todo proyecto llevado a cabo, ya sea la optimización de una red, la automatización de procesos, la investigación de nuevos entornos y protocolos, etc. Centrándonos en el ámbito que nos atañe, aún resulta más importante realizar estas simulaciones, ya que nos permiten realizar un desarrollo previo, una depuración de errores y la opción de realizar pruebas en un entorno seguro.

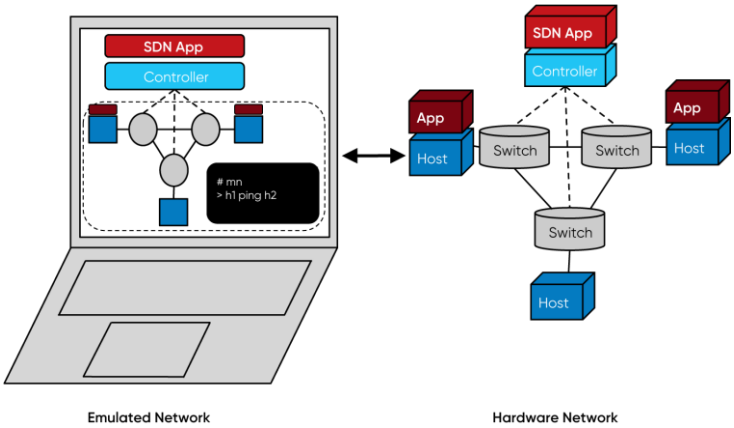


Figura 16. Simulación de proyectos [15].

En este caso nos centraremos en el entorno de Mininet, un emulador de redes basada en Linux, que da la posibilidad de crear redes virtuales (switches, routers, hosts, etc) bajo el dominio de un controlador basado en SDN. Tras conocer este sistema se realizará un estudio de uno de sus componentes, MiniEdit.

Cabe mencionar la diferencia entre un emulador y un simulador, pese a que ambos softwares son muy parecidos, un emulador es un software con capacidad de ejecutar programas en una plataforma distinta a la que se había programados, mientras que un simulador únicamente se centra en representar o simular el comportamiento del programa.

Además de Mininet existen otras alternativas para la simulación de arquitecturas SDN:

- Ns-3, es un potente simulador empleado en ámbitos educativos e investigación de código abierto y gratuito, que persigue el desarrollo de simulaciones realistas, pudiendo ser implementado a tiempo real. [16]
- EstiNet, este simulador perteneciente a la compañía con el mismo nombre, enfocada a soluciones de red dedicadas a productos basados en SDN; proporciona un entorno grafico en el que poder llevar a cabo las simulaciones. [17]
- VNX, o Virtual Networks over linuX, es una herramienta de código abierto que permite la creación de escenarios de prueba virtuales, desarrollada en la Universidad Politécnica de Madrid. [18]

Estas son algunas de las alternativas a Mininet que se pueden encontrar en el mercado, pese a ello, el emulador sujeto a estudio, además de ser el más empleado, cuenta con numerosas ventajas [19], tales como su velocidad de simulación, la capacidad de crear redes personalizadas en las que implementar programas de uso reales, tales como Wireshark en su versión para Linux, la

programabilidad en la transmisión de paquetes, entre otras. Una ventaja añadida es el hecho de al ser un emulador, permite su ejecución en un amplio abanico de dispositivos (ordenadores, servidores, etc.), además de dar una gran capacidad de difusión, permite compartir y replicar resultados de forma sencilla y la emulación por medio de Python scripts. Una característica más a tener en cuenta es el hecho de ser un proyecto de código abierto en constante desarrollo.

La limitación más restrictiva de Mininet es el hecho de únicamente tener soporte para Linux, siendo necesario emplear versiones con los kernels más modernos, siendo Ubuntu la recomendación de los desarrolladores, garantizando su buen funcionamiento en dicho sistema. Otra limitación a tener presente es el no tener tiempo virtual en sus simulaciones, es decir, las mediciones se basarán en tiempo real, lo que imposibilita emplear redes de alta velocidad.

8.2 Instalación Mininet:

A la hora de instalar el entorno de trabajo se nos presentan dos opciones, el empleo de una máquina virtual Mininet o, por el contrario, una instalación nativa. Ambas opciones están disponibles para sistemas Linux, pero podemos emplear la primera opción para emplear este entorno en otros sistemas operativos, ya sea Windows o Mac Os. A continuación, se describirán los pasos a seguir para ambas instalaciones.

Instalación por medio de una máquina virtual *[20],* será necesario contar con una instalación previa de máquinas virtuales, como podría ser el caso de Virtual Box o VMWare. Tras esto se deberá de proceder con la descarga de la imagen de Mininet VM, la cual podremos obtener en la web de Github [21], tras descargar y descomprimir el documento descargado, obtenemos dos archivos, el disco de datos virtual y un segundo archivo que contiene los parámetros de configuración.

Tras los pasos anteriores se debe abrir el archivo correspondiente desde nuestra máquina virtual, y con ello ya podremos empezar a trabajar con Mininet. Cabe la posibilidad de poder acceder a este sistema vía SSH.

Para ello debemos de comprobar que el adaptador de red este configurado, se recomienda crear un segundo adaptador configurado como "host-only", el procedimiento a seguir puerde variar en función del programa para máquinas virtuales que se esté empleando.

Para seleccionar el adaptador, desde la máquina virtual comprobaremos si está activo este nuevo elemento, mediante el comando ***ifconfig –a***, en la siguiente figura se aprecia un ejemplo del adaptador sin configurar:

```
mininet@mininet-vm:~$ ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:0c:29:f1:c2:db
          inet addr:192.168.234.145  Bcast:192.168.234.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:269 errors:0 dropped:0 overruns:0 frame:0
          TX packets:279 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:25177 (25.1 KB)  TX bytes:24644 (24.6 KB)

eth1      Link encap:Ethernet  HWaddr 00:0c:29:f1:c2:e5
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Figura 17. Configuración adaptadora de la máquina virtual (elaboración propia).

Comprobada la configuración, se debe ejecutar el comando ***sudo dhclient eth1***, con ello el adaptador quedará configurado y tras comprobar su dirección IP, se podrá realizar el enlace por medio de un enlace SSH.

```
mininet@mininet-vm:~$ sudo dhclient eth1
mininet@mininet-vm:~$ ifconfig -a_

eth1      Link encap:Ethernet  HWaddr 00:0c:29:f1:c2:e5
          inet addr:192.168.67.129  Bcast:192.168.67.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:746 (746.0 B)  TX bytes:684 (684.0 B)
```

Figura 18. Adaptador configurado (elaboración propia).

Tras esto desde el terminal del ordenador se podrá acceder a la máquina virtual Mininet empleando el comando ***ssh –Y mininet@192.168.67.129***, siendo la IP la configurada en el adaptador. En caso querer acceder desde un terminal OS se debe remplazar la ***Y*** por una ***X***; en ambos casos, tras identificarnos, se podrá hacer completo uso de las prestaciones de Mininet..

En cualquier caso, es sistema pedirá una acreditación por parte del usuario, y con ello poder hacer uso de sus funciones, para ello deberemos de emplear las credenciales de ***mininet*** tanto para el campo de usuario, como para el campo de contraseña.

Instalación nativa *[22]*, este método solo puede ser llevado a cabo en sistemas basados en Linux, preferentemente en el sistema Ubuntu.

Para poder realizar esta instalación, se debe tener acceso a internet desde el equipo donde se va a proceder a la instalación, hecha las comprobaciones previas emplearemos los siguientes comandos:

- ***git clone git://github.com/mininet/mininet***, por medio del cual se importa el contenido de dicho repositorio.
- ***cd mininet y git checkout -b***, cumple la funcionalidad de comprobar las versiones disponibles para instalar.

Como se puede apreciar en la imagen se dispone de diversas opciones de instalación, siendo recomendable para un uso genérico emplear la primera opción, la cual realiza una instalación completa en el directorio principal. Para ello se debe emplear la instrucción ***mininet/útil/install.sh -a*** o en caso de no ser necesario especificar la ruta, emplear únicamente ***install.sh -a***.

```
git clone git://github.com/mininet/mininet
cd mininet
git tag  # list available versions
git checkout -b 2.2.1 2.2.1  # or whatever version you wish to install
cd ..
install.sh -h
To install everything (using your home directory): install.sh -a
To install everything (using another directory for build): install.sh -s mydir -a
To install Mininet + user switch + OVS (using your home dir): install.sh -nfv
To install Mininet + user switch + OVS (using another dir:) install.sh -s mydir -nfv
mininet/util/install.sh -a
```

Figura 19. Instalación nativa de Mininet (elaboración propia).

Tras la instalación es recomendable empelar el comando ***sudo mn –test pingall*** con el único propósito de comprobar las funcionalidades básicas de Mininet, y con ello la correcta instalación.

Es probable que a la hora de seguir estos pasos nos encontremos con que el sistema nos devuelva "*command not found*" al ejecutar el primer comando ***git clone git…,*** en caso de ser así la solución llega tras la ejecución de las siguientes tres líneas de comandos:

sudo apt-get update
sudo apt-get upgrade
sudo apt-get install git

Con ello comprobaremos las actualizaciones del sistema e instalaremos la versión correspondiente de Git necesarias para poder obtener el software requerido para llevar a cabo el estudio de este entorno de trabajo.

8.3 Entorno de trabajo de Mininet

Una vez implementado el sistema resulta necesario hacer un repaso de los comandos básicos para poder implementar una estructura de red. Por medio del comando "help" podremos acceder a un pequeño resumen de las instrucciones y su sintaxis:

```
mininet> help

Documented commands (type help <topic>):
========================================
EOF    gterm  iperfudp  nodes        pingpair      py      switch
dpctl  help   link      noecho       pingpairfull  quit    time
dump   intfs  links     pingall      ports         sh      x
exit   iperf  net       pingallfull  px            source  xterm

You may also send a command to a node using:
  <node> command {args}
For example:
  mininet> h1 ifconfig

The interpreter automatically substitutes IP addresses
for node names when a node is the first arg, so commands
like
  mininet> h2 ping h3
should work.

Some character-oriented interactive commands require
noecho:
  mininet> noecho h2 vi foo.py
However, starting up an xterm/gterm is generally better:
  mininet> xterm h2
```

Figura 20. Comandos y sintaxis básica (elaboración propia).

En la imagen podemos apreciar que la sintaxis básica para programar los diferentes nodos que pueden conformar una red *"(nombre del nodo) comando"*, de esta forma el sistema reconoce el nodo en concreto sin necesidad de espaciarlo por su IP, cabe destacar que para ejecutar comandos interactivos orientados a las características de los nodos necesitan el empleo de *"noecho"*, tal y como se muestra en la figura anterior.

Tras comentar la sintaxis básica se debe prestar atención a los comandos indicados:

- "***EOF***", este comando se emplea para detener y salir de mininet.
- "***dpctl***", es una utilidad de administración que permite cierto control sobre el conmutador OpenFlow, permitiendo agregar flujos a las tablas, consultar sus características y cambiar otras configuraciones.
    - *Sintaxis: dpctl [OPTIONS] SWITCH COMMAND [ARG ...]*

  - *Añadir un flujo / Eliminar un flujo – add-flow / del-flow*
- "***dump***", realiza el volcado de la información de los nodos.
- "***exit***", esta función nos permite salir de mininet.

- "***gterm***" y "***xterm***", ambos comandos se emplean para abrir una ventana de comandos de un nodo.
- "***help***", comando para desplegar la ayuda genérica o de un comando en concreto.
- "***intfs***", lista las interfaces de los diferentes nodos.
- "***iperf***", realiza una prueba simple de TCP iperf entre dos hosts, permitiendo medir la velocidad de conexión máxima entre estos.
  - Sintaxis: "iperf (nodo 1) (nodo 2)"
- "***iperfudp***", realiza una prueba semejante a la anterior, pero por UDP.
  - Sintaxis: "iperfudp bw (nodo 1) (nodo 2)"
- "***link***", levanta o tumba un enlace entre dos nodos.
  - Sintaxis: "link h1 h2 (up/down)"
- "***links***", muestra el estado de todos los enlaces establecidos.
- "***net***", muestra las conexiones de la red.
- "***nodes***", lista todos los nodos de la red.
- "***noecho***", se emplea para ejecutar un comando interactivo con eco desactivado.
- "***pingall***", se emplea para hacer un ping entre todos los hosts, devolviendo un resumen de todos los pings realizados.
- "***pingallfull***", realiza un ping ente todos los hosts, devolviendo todos los resultados de las operaciones.
- "***pingpair***", comando útil para realizas pruebas entre los dos primeros hosts.
- "***pingpairfull***", realiza la misma función que el comando anterior, da como respuesta todos los resultados de esta operación.
- "***ports***", muestra la información de los puertos y las interfaces de los diferentes switch.
- "***px***", se empela para ejecutar una declaración de Python.
- "***py***", se emplea para evaluar una expresión de Python.
- "***quit***", se empela para salir.
- "**sh**", es necesario para poder emplear comandos de terminal externos a la interfaz de mininet.
- "**source**", se emplea para leer comandos de un archivo.
- "***switch***", mediante este comando podemos arrancar o parar un switch.
- "***time***", permite medir el tiempo necesario para medir el tiempo necesario de cualquier comando en este sistema.
- "***x***", permite crear un túnel X11 para un nodo dado.

Al iniciar la plataforma de Mininet se crean diversos nodos con los que poder practicar los comandos anteriores. Estos nodos estarían formados por dos hosts, un switch OpenFlow y un controlador.

En caso de querer añadir un nuevo host con el que poder ampliar las simulaciones de la red, se debe emplear el comando ***py net.addHost('<nombre del host>')*** para incluirlo en el escenario de trabajo.

Una vez incluido se deben de configurar las conexiones que le permitirán comunicarse con el resto de la red. Para ello se empleara la instrucción ***py net.addLink(s1, net.get('<nombre del host>'))***, tras aplicar los cambios en el switch, empleando ***py (nombre del switch).attach('(nombre del puerto de enlace)')***; y configurar una ip valida al nuevo host, quedará listo para la comunicación con el resto de la red.

Para configurar el nuevo terminal se debe emplear ***py net.get('(nombre del terminal)'.cmd('ifconfig (puerto del terminal) (ip)')***, de esta forma quedara asociada la IP deseada al puerto que se está empleando.

En la figura 20 se puede observar un ejemplo de la creación del host 3 y de su correcta comunicación, con los otros hosts, por medio del comando ping:

```
mininet> py net.addHost('h3')
<Host h3:  pid=3954>
mininet> py net.addLink(s1, net.get('h3'))
<mininet.link.Link object at 0x7f85f29668d0>
mininet> py s1.attach('s1-eth3')
mininet> py net.get('h3').cmd('ifconfig h3-eth0 10.0.0.10')
mininet> h1 ping h3
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=1 ttl=64 time=13.0 ms
--- 10.0.0.10 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9188ms
rtt min/avg/max/mdev = 0.072/1.418/13.079/3.888 ms
mininet> h2 ping h3
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=1 ttl=64 time=9.40 ms
--- 10.0.0.10 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4077ms
```

Figura 21. Ejemplo creación de un nuevo host en Mininet (elaboración propia).

Si se desean añadir otros elementos a la red como por ejemplo un switch se debe empelar el comando ***net.addSwitch('(nombre del nuevo nodo)')***.

Pese a ello la forma más habitual de trabajar con esta tecnología es haciendo uso del amplio abanico de herramientas que ofrece, como la creación de una red por medio del comando "***topo***", que nos permite crear redes sencillas en función de su topología ya sea single, linear, en árbol o personalizada.

Un ejemplo de ello sería la estructura resultante de emplear ***$ sudo mn –topo linear, (número de switches y hosts)***. De la ejecución de esta instrucción se obtendría una red con una topología simple formada por el número de nodos indicados.

El inconveniente al método anterior es que las redes resultantes carecen de controlador de red.

Por medio del comando ***controller***, se implementarían topologías con un controlador, pudiendo especificar el tipo de controlador, la topología, el numero de componentes y el tipo de switch.

Un ejemplo de ello sería la ejecución del argumento:

$ sudo mn –controller=remote –topo sigle, 3 –mac –switch ovsk, protocols=OpenFlow10 --nat

Obteniendo una arquitectura con un controlador remoto, con una topología formada por un switch conectado a tres hosts en una estructura single. Los terminales tendrían su propia dirección MAC, mientras que, en el caso del controlador, al no especificarse la IP recibirá una por defecto. Para especificar una IP para el controlador, bastaría con añadir a la instrucción el argumento: ***ip=(dirección ip)***. En el caso del switch, se ha especificado el tipo que se va a implementar, ovsk, y su protocolo, OpenFlow en su versión 1.0.

Una de las herramientas que ofrece mininet es Miniedit, la cual permite la creación y simulación de redes de una forma muy simplificada y aun alto nivel mediante una interfaz grafica.

8.4 ¿Qué es Miniedit?

Dentro del propio editor podemos encontrar la carpeta de *examples*, donde podemos encontrar diversos ejemplos y herramientas, en la que destaca la herramienta de Miniedit, una extensión de Mininet, que nos permite crear redes de forma sencilla sobre un terminal gráfico.

Esta interfaz facilita la creación de redes, realizandose la programación de las mismas en un según plano a priori oculto para el usuario, pese a ello esta plataforma presenta ciertas limitaciones en comparación con todas las capacidades que presenta el propio Mininet.

Para poder ejecutar esta función deberemos cerrar las simulaciones que se estén ejecutando en el entorno de Mininet, siendo necesario emplear el comando ***exit***, o similar, para cerrar la simulación creada en la interfaz principal, mediante el comando ***killall controller***, se finaliza los procesos del controlador de la red, y por ultimo se debe ejecutar ***sudo mn –c***, con el cual se limpian todos los elementos que hayan podido ser creados en la simulación anterior.

Una vez cerradas todas las simulaciones o con un terminal nuevo, se pueden emplear estas dos instrucciones para arrancar el entorno de trabajo:

sudo ~/mininet/examples/miniedit.py
sudo python ./mininet/examples/miniedit.py

8.4.1 Comandos y descripción de la interfaz de trabajo

Una vez arrancado Miniedit presenta una interfaz de usuario simplificada, con una zona de trabajo y una fila de herramientas representadas por iconos, en el lado izquierdo, además de una barra de menú en la parte superior.

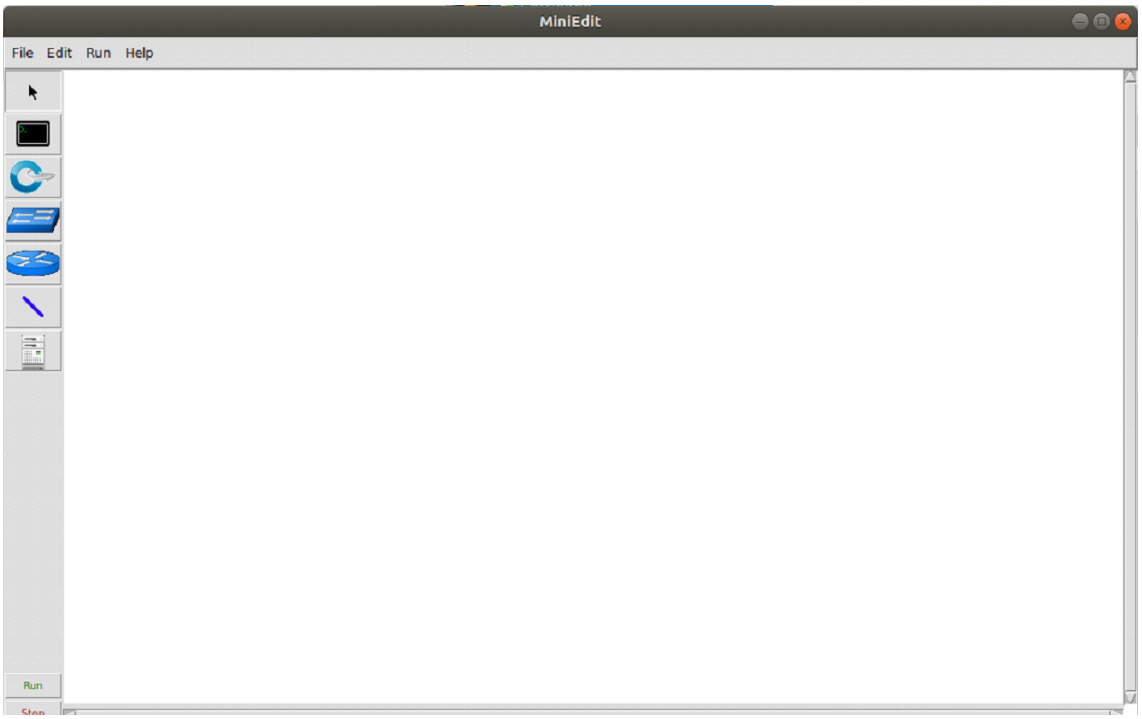


Figura 22. Entorno de trabajo MiniEdit (elaboración propia).

En el lateral del lienzo de trabajo, podemos observar las herramientas de trabajo, concretamente son las siguientes:

 ***FLECHA DE SELECCIÓN***: Se empela para mover los nodos por el escenario de trabajo, esta función no es estrictamente necesaria para seleccionar los nodos, puesto que podemos seleccionarlos con el resto de herramientas. Empleando el botón derecho para poder emplear el menú de configuración,

 ***HOST***: Permite añadir elementos host o terminales al lienzo, mientras este seleccionada nos permite añadir múltiples elementos, los cuales pueden ser configurados mediante las propiedades en el menú desplegado con el botón derecho.

 ***CONMUTADOR***: Crea un Open vSwitch, que implementa una plataforma de cálida, capaz de hacer una gestión estándar y habilitar de forma programable las funciones de forwarding o transmisión de la información, pudiendo ser configurados de la misma forma que el elemento anterior.

 ***CONMUTADOR TRADICIONAL***: Crea un switch Ethernet con una configuración predeterminada, funcionando de forma independiente al controlador. Este equipo no puede ser configurado, parten con el protocolo Spanning Tree deshabilitado, por lo que no se deben conectar en bucle.

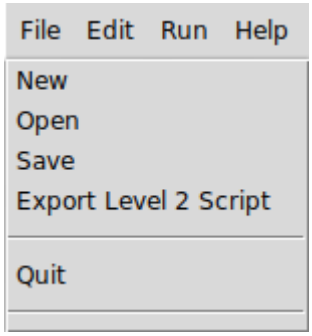
 ***ROUTER TRADICIONAL***: o Router Legacy, crea enrutadores básicos, que funcionan de forma independiente al controlador. En definitiva, es un host con la IP Forwarding habilitada. Este nodo no puede ser configurado desde MiniEdit.

 ***ENLACES***: Crea los diferentes enlaces entre los elementos de la red. Las propiedades de cada enlace se pueden configurar, mediante el botón derecho del ratón.

 ***CONTROLADOR***: Permite implementar diversos controladores de red, por defecto se crea un controlador de openFlow. Se pueden configurar otro tipo de controladores modificando las propiedades o configuración de los mismos, desde el menú desplegable con el botón derecho sobre él.

EJECUTAR / PARAR: Por medio de este elemento se tiene control sobre la simulación de la red conformada. Mientras se está ejecutando una simulación, por medio del botón derecho del ratón sobre un elemento, se muestran funciones operativas como abrir un terminal, ver su configuración o establecer el estado de un enlace (activándolo o desactivándolo) exclusivas del mudo de emulación.

Conociendo las funciones de los botones de la interfaz de trabajo, resulta conveniente familiarizarse con el empleo de los menús de la parte superior. Comenzado por el lado izquierdo, el primer menú es el menú de archivo o “File”:



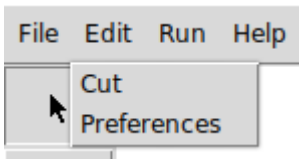
Desde este menú se puede acceder a un nuevo lienzo de trabajo, abrir uno ya creado o, guardar o exportar el actual. Estas dos últimas funciones resultan de gran utilidad a la hora de simular en mininet entornos de trabajo creados con la herramienta grafica.

Con la opción de exportar se genera un archivo con extensión ".py", la cual puede ser abierta desde el programa de edición Python.

Por medio de la última opción se cerraría el escenario de trabajo actual.

Figura 23. Menú de archivo en la plataforma de Miniedit (elaboración propia).

El segundo menú que se presenta es el menú de edición o "Edit", un sencillo menú que cuenta con la herramienta "Cut", su funcionalidad es semejante a una herramienta de borrado, su método de empleo se resume en seleccionar el elemento que se quiera cortar o borrar y seleccionar la función en dicho menú.



La otra opción que figura en este menú es la ventana de *Preferences* o preferencias, donde se pueden realizar las configuraciones generales de la zona de trabajo.

Figura 24. Menú de edición en la plataforma de Miniedit (elaboración propia).

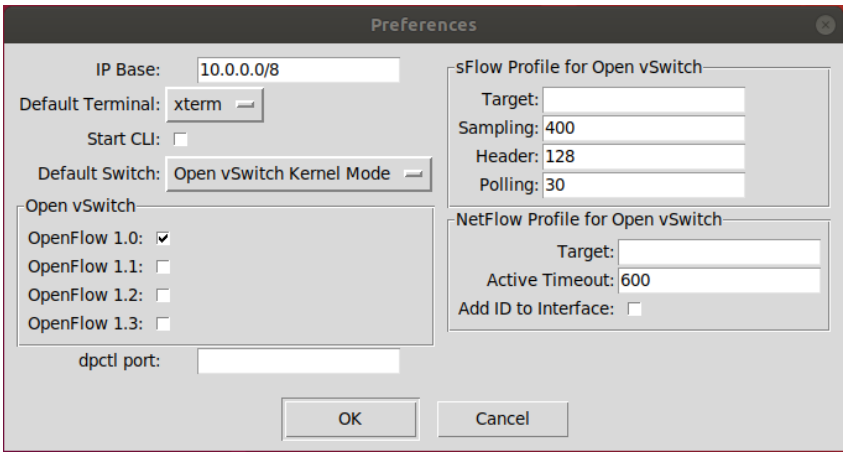


Figura 25. Preferencias del entorno de trabajo Miniedit (elaboración propia).

Desde la interfaz que se abre tras seleccionar *Preferences* se pueden editar valores globales del área de trabajo, tales como la dirección IP por defecto, incluyendo su máscara de red, el terminal de trabajo de los hosts, el modelo y la versión de los switch creados con tecnología OpenFlow; además del puerto de dpctl a emplear.

En la parte derecha de la pantalla se habilita la edición de las propiedades de los protocolos sFlow y NetFlow, los cuales se pueden habilitar en las propiedades de cada switch.

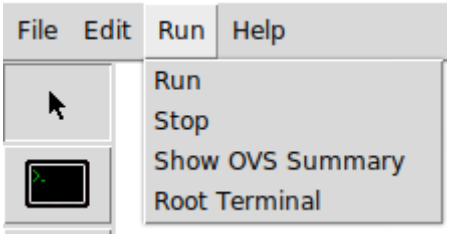
NetFlow [23], es un protocolo de red propiedad de Cisco Systems, empleado para capturar información acerca del tráfico IP.

sFlow [24], es un estándar de muestreo que permite el análisis de los flujos de datos, en la capa de aplicación, de forma simultánea en todas las interfaces.

El ultimo campo editable desde esta ventana es el arranque de la línea de comandos o CLI. Dicha funcionalidad resulta muy útil a la hora de comprobar la configuración de las redes creadas en este entorno de trabajo, puesto que nos permite hacer uso de las funcionalidades de la plataforma Mininet, sin la necesidad de guardar y abrir la estructura creada desde otra ventana de comandos.

Una vez activa esta función se puede observar como en la línea de comandos o CLI desde la que se ha arrancado la herramienta Miniedit pasa a tener el prompt "mininet>".

Esta habitación nos permite seguir los pasos de la creación de la red una vez se iniciada la simulación, así como de todas las modificaciones referentes a la misma, algunos ejemplos serían la modificación de las propiedades de un enlace, el tipo de switch o de un controlador.



El siguiente menú presente en Miniedit es el menú de "Run" o de ejecución, desde el cual podremos iniciar o detener la simulación de la estructura abierta en el lienzo de trabajo.

Además de dotar de la opción de abrir un nuevo terminal Root, desde el que poder ejecutar el entorno de Mininet.

Figura 26. Menú de ejecución en la plataforma de Miniedit (elaboración propia).

La opción más importante de este menú es "Show OVS Summary", desde la cual se tiene acceso a la ventana de comandos llamada "OVS Summary", desde la cual se puede observar la configuración de todos los puertos de los nodos de comunicación de la red implementada.

En la siguiente figura se puede apreciar un ejemplo de la funcionalidad de este terminal, en el cual se puede apreciar la configuración de todos los puertos en uso en ambos switches.

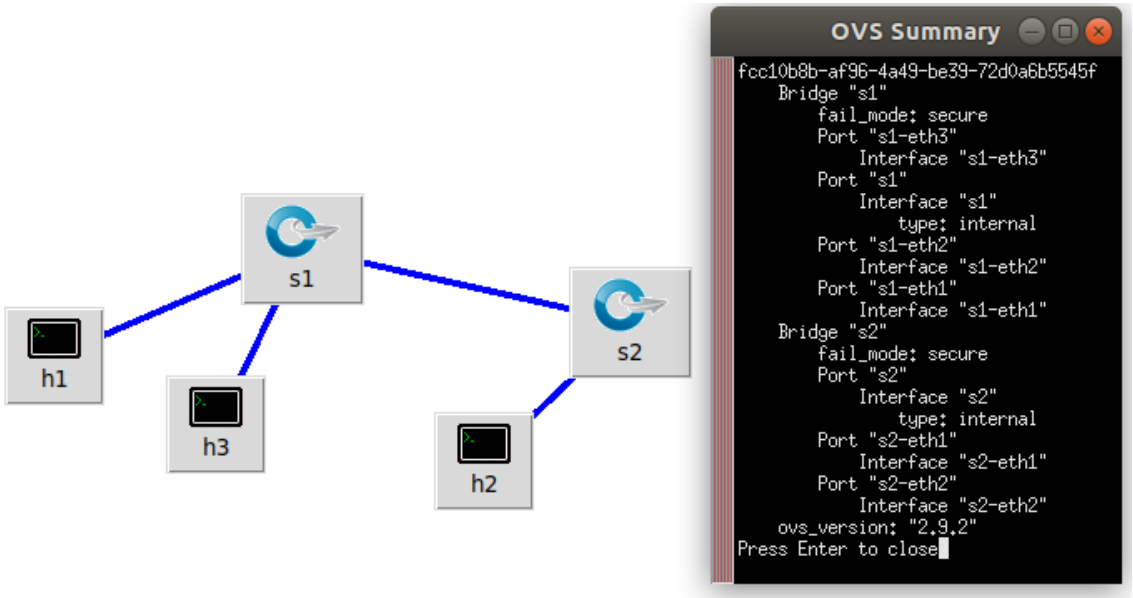


Figura 27. Funcionalidad OVS Summary en Miniedit (elaboración propia).

El ultimo menú de la barra de herramientas es el destinado a dar soporte, reconocible por el nombre de “Help”, su única funcionalidad es dar información al usuario acerca del editor que se está empleando.

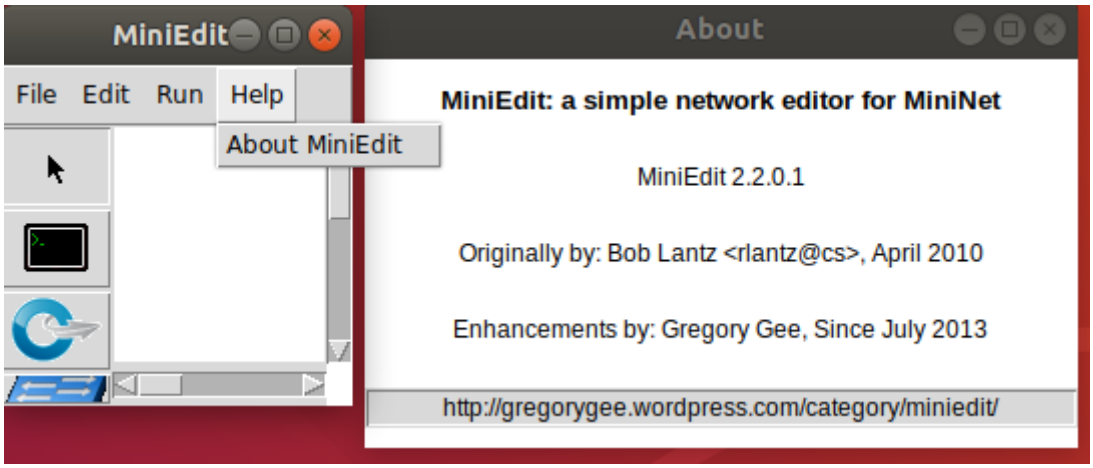


Figura 28. Menú de ayuda en la plataforma de Miniedit (elaboración propia).

Tras haber adquirido una visión ampliada y conocer las funcionalidades básicas del entorno de trabajo, resulta conveniente crear y emular una sencilla arquitectura de red con la que poder comprobar el funcionamiento de las funcionalidades ya descritas, así como las que hacen referencia a los distintos componentes de una red.

8.5 Desarrollo práctico de una red en MiniEdit

Con esta idea presente, se propone la creación de una arquitectura sencilla compuesta por tres switches, con tecnologia OpenFlow e interconectados; por comunicación directa con un controlador de red. Además de diversos usuarios o host, conectados a cada uno de los vSwitches.

Siguiendo las caracteristicas anteriormente mencionadas se realiza la siguiente red:

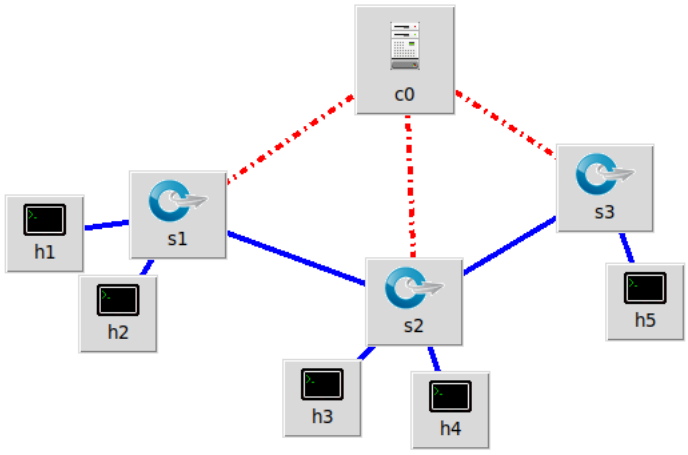
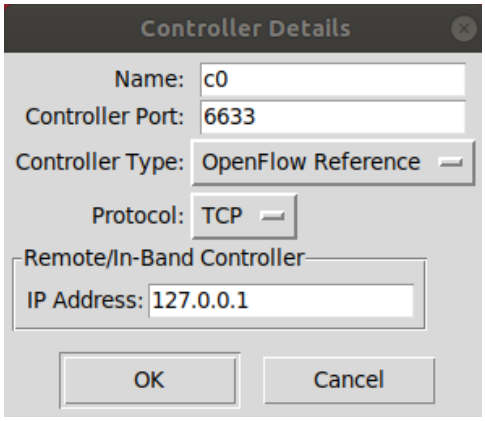


Figura 29. Arquitectura de red a simular en Miniedit (elaboración propia).

En primer lugar se comprobaran las configuraciones por defecto de cada uno de los componentes de la red, así como de las opciones que presentan. En primer lugar se accedera a las propiedades del controlador de red:

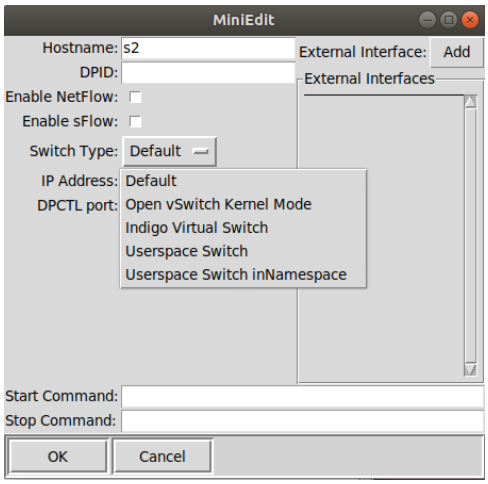


Entre los campos editables que presenta el controlador se encuentran el nombre o etiqueta que se emplea en mininet para hacer referencia al mismo, el puerto que emplea, el protocolo a emplear (TCP o SSL), su IP remota y el tipo de controlador, pudiendo elegir entre:

- OpenFlow Reference.
- In-Band Controller.
- Remote Controller.
- OVS Controller

Figura 30. Propiedades del controlador en entorno Miniedit (elaboración propia).

En el caso de los Switches las opciones son mayores, además de permitir la edición de los campos basicos, como serían el nombre del nodo, su DPID, su dirección IP y su puerto DPCTL, permite la habilitación de los protocolos NetFlow y SFlow, configurables desde la ventana de preferencia en el menu de edicción.



Tambien se dispone de la posibilidad de establecer un comando de aranque y de parada, así como de establecer una interfaz externa.

Pese a disponer de varios tipos de switch, para esta simulación se empleara la configuración por defecto, la cual esta establecida en la ventana de preferencias del entorno, Figura 25. Dicha configuración corresponde con un “Open vSwitch” siendo la primera opcón de las listadas en la Figura 31.

Figura 31. Propiedades del switch en entorno Miniedit (elaboración propia).

Las propiedades configurables básicas de cada host son semejantes a las vistas en los switches, nombre del nodo, dirección IPE y la capacidad de establecer comandos de inicio y paro. Estos terminales también cuentan con la posibilidad de configurar una ruta por defecto, indicar la cantidad de cores y de CPU disponible. Además de poder configurar un directorio privado, una interface VLAN y otra externa.

Al igual que para los casos anteriores dejaremos sus valores por defecto.
Al no indicar una IP especifica para cada host estarán configurados con la dirección por defecto acabada por su número de host, es decir, en el caso del terminal de la Figura 32, el “h3”, su IP

por predeterminada sería 10.0.0.3 con una máscara de 8 bits (255.0.0.0).

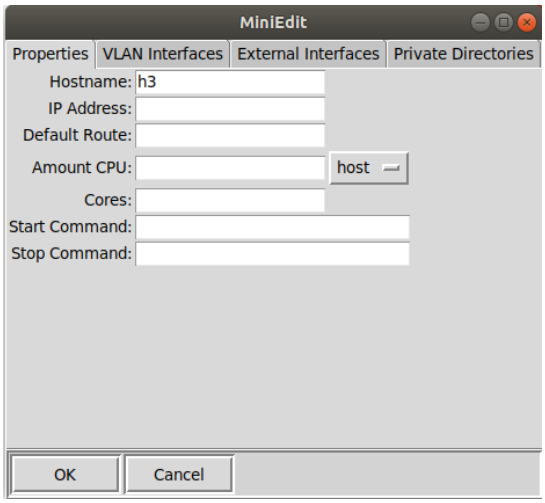


Figura 32. Propiedades de un host en Miniedit (elaboración propia).

Por último, en la configuración de los enlaces es posible establecer condiciones o calidad de enlace de una forma más detalla, permitiendo realizar simulaciones lo más realistas posibles, pudiendo configura el speedup, el retraso o delay, el porcentaje de perdidas, el tamaño máximo de la cola y la fluctuación del retardo o jitter.

Para la simulación emplearemos un ancho de banda de 100MBits.

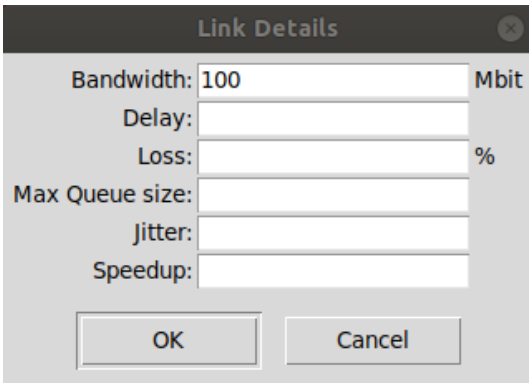


Figura 33. Propiedades de un enlace en Miniedit (elaboración propia).

Tras repasar la configuración de todos los elementos, se debe comprobar la correcta configuración de los puertos empleados en los diferentes equipos, para ello hacer uso del terminal OVS Summary,Figura 27, donde se puede apreciar la correcta configuración, así como el controlador al que se encuentra asociado cada switch, en este caso todos hacer referencia al mismo, C1, con IP 127.0.0.1.

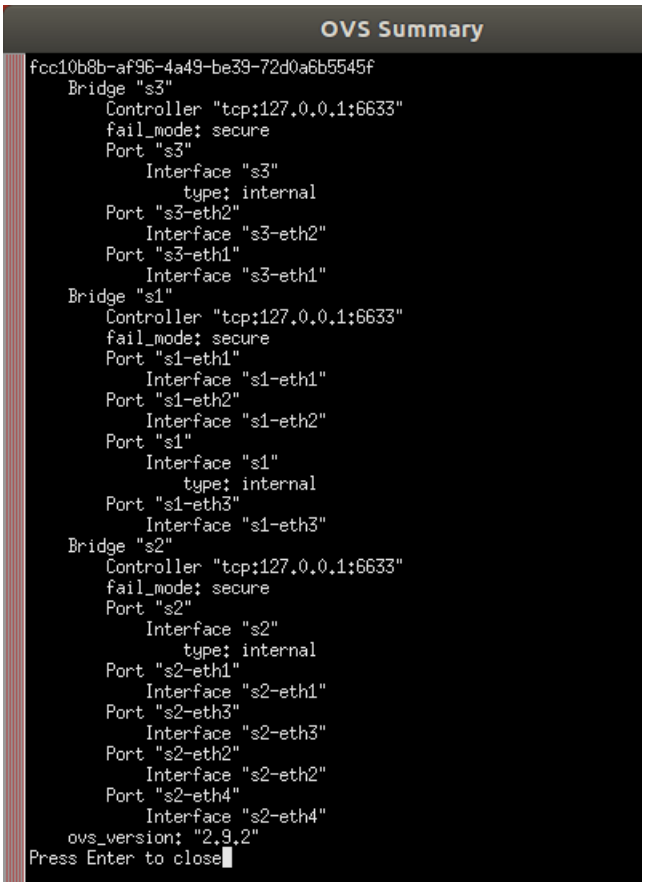


Figura 34. Conf. de los puertos de la red a simular en Miniedit (elaboración propia).

Antes de comenzar una transmisión, y con las configuraciones previas realizadas, se puede observar que no hay contenido en las tablas de flujos de los diferentes enrutadores, para poder comprobar esta afirmación se debe emplear el comando ***dpctl dump-flows*** en el terminal con las funciones de mininet activas. Para ello se ha habilitado la función CLI en la configuración de las preferencias del entorno de trabajo.

La afirmación anterior hace referencia a la parte superior de la Figura 35.

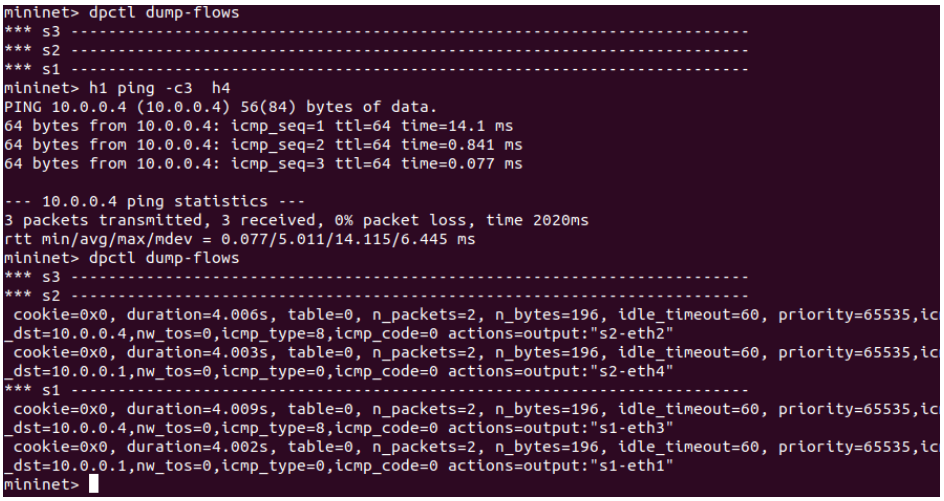


Figura 35. Comprobación de la comunicación en Miniedit (elaboración propia).

Una vez realizada la comprobación, se inicia una transmisión entre el host 1 y el host 4, en este caso se emplea un ping con el fin de comprobar la comunicación y la actualización de las FlowTables.

Realizada la transmisión, como se puede observar en la figura anterior, las tablas de los switch implicados en la transmisión se han actualizado, gracias a lo cual se habilita la correcta comunicación entre dichos terminales.

Comprobada la comunicación, se van a comprobar los efectos de las alteraciones de los enlaces entre los diferentes nodos. En este caso se simulará la caída de un enlace, el aumento de la tasa de retardos y el incremento del número de perdidas en el medio.

Antes de realizar ninguna prueba con los enlaces se deben de borrar las entradas de las tablas por medio de la función ***dpctl del-flows***.

En primer lugar, se procede a deshabilitar un enlace desde la interfaz gráfica, tras lo cual se intentará realizar una comunicación entre el host 1 y el host 3. Cabe esperar que la comunicación entre ambos nodos falle y, al no tener conexión, sus respectivas entradas a las tablas de flujo no se generen.

Cabe destacar que al inutilizar un enlace desde la interfaz grafía, este se muestras como una línea azul discontinua, a diferencia de los enlaces activos que se muestran con una línea continua.

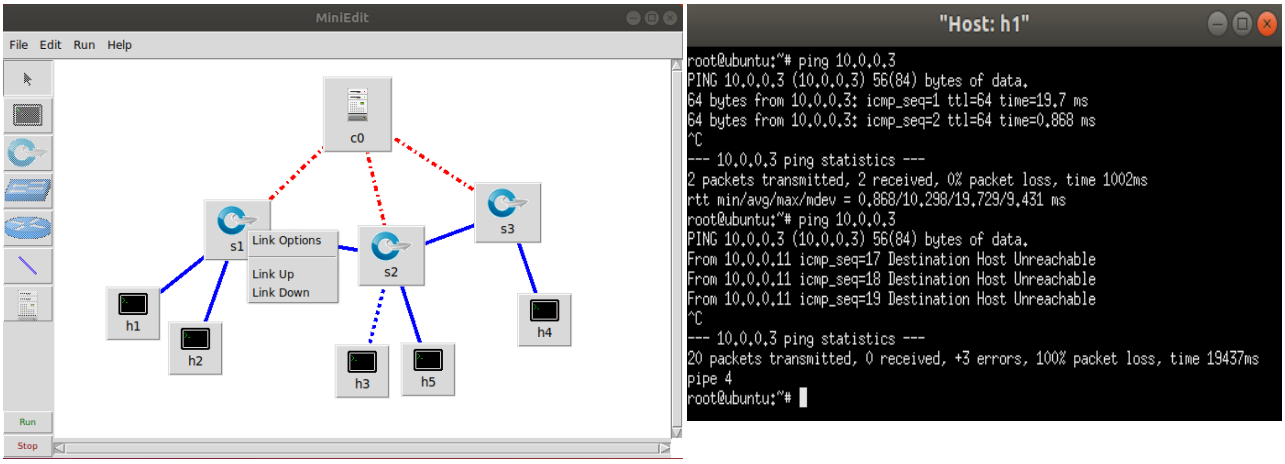


Figura 36. Fallo de comunicación tras la caída de un enlace en Miniedit (elaboración propia).

```
mininet> dpctl dump-flows
*** s3 ------------------------------------------------------------------------
*** s2 ------------------------------------------------------------------------
 cookie=0x0, duration=21.091s, table=0, n_packets=2, n_bytes=196, idle_timeout=60, priority=65535,icmp,in_port="s2-eth4"
w_dst=10.0.0.4,nw_tos=0,icmp_type=8,icmp_code=0 actions=output:"s2-eth2"
*** s1 ------------------------------------------------------------------------
 cookie=0x0, duration=21.095s, table=0, n_packets=2, n_bytes=196, idle_timeout=60, priority=65535,icmp,in_port="s1-eth1"
w_dst=10.0.0.4,nw_tos=0,icmp_type=8,icmp_code=0 actions=output:"s1-eth3"
mininet>
```

Figura 37. Tabla de flujos con la caída de un enlace en Miniedit (elaboración propia).

Tras realizar la simulación se puede afirmar que, al perder la comunicación por un enlace, en caso de no disponer de una alternativa de comunicación, la tabla de flujos no se actualiza, generando únicamente la ruta de comunicación con el controlador de la red.

Teniendo en cuenta que la habilitación y des habilitación de un enlace desde la interfaz gráfica única mente se puede realizar con la simulación en marcha, se vuelve a normalizar el estado del enlace caído.

Con la simulación detenida se editan las características del enlace de comunicación entre los dos switches, en un primer caso práctico se aumenta el delay o retardo en la comunicación.

Una vez hecha la modificación y la simulación en marcha se el terminal del host 1, el cual emplearemos para hacer un seguimiento de la comunicación con el host 3.

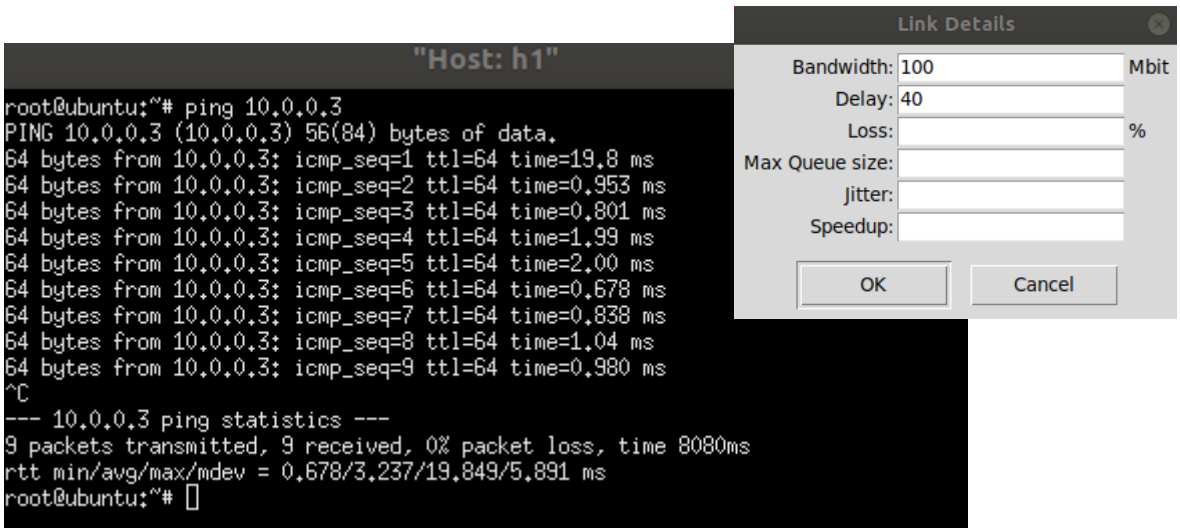


Figura 38. Comunicación con tasa de delay en Miniedit (elaboración propia).

Tras la visualización de nueve paquetes ICMP resultantes de realizar el ping entre ambos terminales resulta evidente la fluctuación del retardo en cada una de las transmisiones causada por el retraso establecido.

Por último, se realiza el caso práctico con el que se pretende observar el comportamiento de la red al aumentar la tasa de perdidas en un enlace.

Para ello, y al igual que el caso anterior, se debe detener la simulación para poder realizar la configuración del enlace.

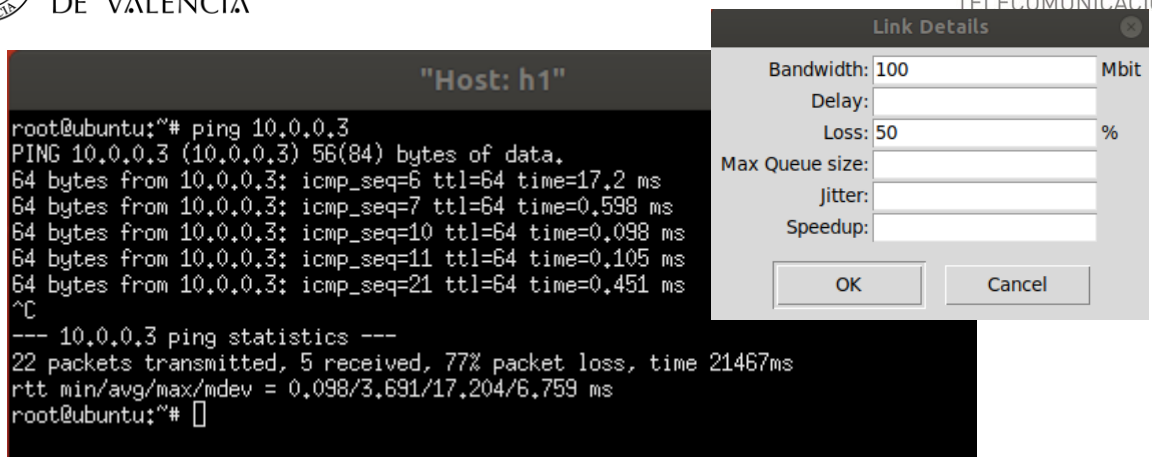


Figura 39. Comunicación con tasa de perdidas en Miniedit (elaboración propia).

Como se puede observar en la imagen anterior de los 21 paquetes que se han trasmitido entre los terminales, únicamente han alcanzado su destino cinco de ellos. Por lo que quedaría demostrado el correcto funcionamiento de las funciones de configuración de los distintos enlaces.

Para la realización de estas simulaciones, se ha decidido hacer el seguimiento desde el propio terminal, siendo posible emplear la herramienta de Wireshark, que es un analizador de tráfico, desde el cual se puede realizar el mismo procedimiento.

En caso de querer hacer el seguimiento de las mismas simulaciones desde la aplicación de Wireshark, se debe de abrir desde la línea de comandos *Root* de Miniedit, a la cual se puede acceder desde el menú de *Run*, Figura 26.

Una vez abierta la aplicación, y tras es el escaneo de los nodos disponibles para su monitorización, se debe seleccionar el nodo o el puerto que se quiere escuchar o analizar, en este caso se hará el seguimiento al puerto ethernet 1 del primer switch, *s1-eth1*.

A modo de ejemplo se repetirá la segunda simulación, con un delay de 40 unidades. Como se puede apreciar en la siguiente figura el resultado obtenido es el mismo a los valores resumidos en la CLI del host 1.

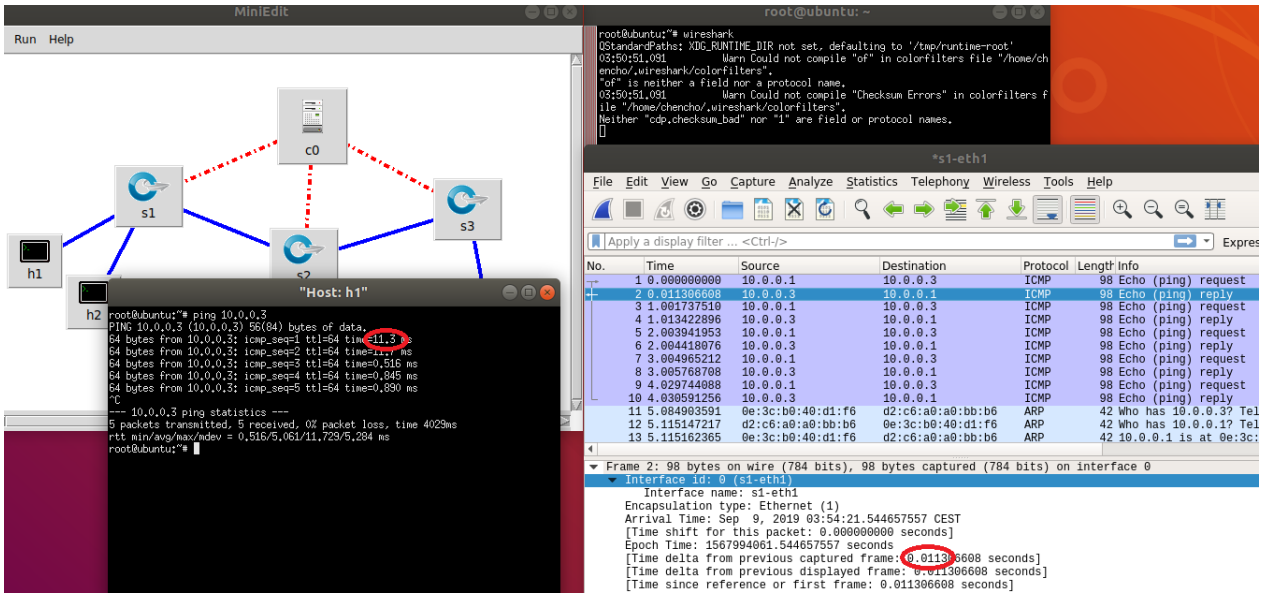


Figura 40. Comunicación con delay y Wireshark en Miniedit (elaboración propia).

Una vez completadas las emulaciones, para poder finalizar la misma de forma correcta, se debe finalizar primero el CLI de Mininet, por medio de la instrucción ***exit***, y tras esto se puede proceder a detener la ejecución del entorno de Miniedit, por medio del botón *Run / Stop*.

Este procedimiento se especifica al habilitar la CLI, y resulta aconsejable seguirlo para evitar fallos en la emulación que puedan imposibilitar su correcta ejecución.

Una vez detenida la ejecución de la red, se puede realizar el guardado de la misma para una próxima ejecución. Siendo necesario distinguir el método de guardado en función de la plataforma desde la que se quiera volver a abrir la red creada.

En caso de querer seguir empleando la interfaz de Miniedit, se debe emplear la función ***Save*** en el menú de archivo, generando un archivo con extensión ***.mn***, compatible con esta interfaz.

Por el contrario, si se desea poder seguir realizando las emulaciones en Mininet, se debe emplear la opción de ***Export Level 2 Script***, la cual crea un documento con extensión ***.py***, empleada en este entorno de trabajo.

Ambas opciones se encuentran en el menú de archivo Figura 23.

Capítulo 9. Conclusiones y propuestas de trabajo futuro

Mediante el desarrollo del presente trabajo se ha perseguido la idea de dar una visión global y un entendimiento más detallado del funcionamiento de las redes definidas por software, pudiendo resumir en términos globales dicho concepto en:

Las SDN son una plataforma de innovación constante, cuya finalidad es conseguir una colaboración entre redes que respondan a las necesidades de las aplicaciones, es decir, que las aplicaciones controlen la red, siendo capaces de programar la mima

Donde queda plausible la ruptura con el planteamiento inicial de que la red es ajena a los requisitos de las aplicaciones que trabajan en ella, configurándose según sus propias prioridades. Y la gran importancia de OpenFlow en la división de las capas, permitiendo dicha programabilidad.

Tras un estudio más teórico de las cualidades del emulador Mininet, queda plausible su gran capacidad de desarrollo, siendo una herramienta potente y ágil a la hora de realizar emulaciones de redes con topología SDN. Aumentando su potencial de trabajo al permitir la ejecución de cualquier comando de sistema en los elementos de red creados.

Además de incluir herramientas tan practicas como es el caso de Miniedit, que mediante su interfac simplifica en gran medida la creación de redes mas complejas, con un mayor número de interconexiones e incluso con diversas cualidades de enlace. Pese a estas funcionalidades, no deja de ser un complemento de Mininet, teniendo que recurrir a esta ultima para llevar a cabo ejecuciones más concretas o detalladas.

Con todo ello resulta evidente la revolución que supone la implementación de esta tecnología en ámbitos cotidianos.

A la hora de pensar en posibles propuestas de trabajo resulta inevitable sentirse abrumado frente al gran potencial que demuestra esta tecnología, pese a ello los principales retos a seguir desarrollando se podría englobar en dos conceptos, compatibilidad y *robusted* de la red.

Este primer concepto hace referencia a la necesidad de asegurar una compatibilidad entre dispositivos, la cual repercutirá directamente en la imagen de esta tecnología, pudiendo resultar más atractiva para los operadores, y con ello facilitando su implantación a un nivel más global.

El segundo concepto va referido a la necesidad de asegurar que la red sea robusta frente a posibles ataques o fallos en la misma, puesto que, como ya se ha comentado, el controlador supone la pieza clave y al mismo tiempo el punto débil de este sistema. Siendo un punto importante a desarrollar.

Un punto añadido como propuesta de trabajo es el acercamiento de esta tecnología a las aulas, emplear el potencial de herramientas como Mininet para dar a conocer el futuro de la arquitectura de redes.

Cada vez resulta más visible el nivel de desarrollo, aceptación e implementación de este nuevo sistema a nivel empresarial, dejando de ser un concepto más teórico centrado en los entornos de laboratorio, para empezar a ganar mercado como tecnología pionera, con la capacidad de marcar las bases de la arquitectura de redes de un futuro cada vez más próximo.

Capítulo 10. Bibliografía

[1] D.L. Tennenhouse, J.M. Smith, W.D. Sincoskie, D.J. Wetherall y G.J. Minden. "A Survey of Active Network Research". IEEE Comunications Magazine, pp. 80-86, January 1997

[2] Dueñas C.L., Marín Y.A., Cruz H., "Red SDN vs Red Tradicional". https://www.researchgate.net/publication/319097382_SDN_Network_vs_Traditional_Network. [Online].

[3] Enciclopedia libre, Wikipedia. "Interfaz de programación de aplicaciones". https://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones. [Online].

[4] GSMA corporation, "The Mobile Economy 2019". https://www.gsmaintelligence.com/research/?file=b9a6e6202ee1d5f787cfebb95d3639c5&download. [Online].

[5] Punt informàtic. "¿Cómo funciona la red definida por software (SDN)?". https://puntinformatic.com/como-funciona-la-red-definida-por-software-sdn/. [Online]

[6] Serrano D.A, "Redes Definidas por Software (SDN): OpenFlow". Cap. 2.1.

[7] Tootoonchian A. y Ganjali Y., "HyperFlow: A distributed Control Plane for OpenFlow" https://pdfs.semanticscholar.org/c368/944be5267a1c3c72de98f0078ea7ec4823e5.pdf?_ga=2.11561396.120191785.1564686213-1014933316.1564686213. [Online].

[8] OpenFlow Consortium, OpenFlow switch specification. https://www.opennetworking.org/documents.php. [Online]

[9] Fraile R., "Casos de uso de SDN: La búsqueda continúa". https://empresas.blogthinkbig.com/casos-de-uso-de-sdn-la-busqueda-continua/. [Online]

[10] Enciclopedia libre, Wikipedia. "OpenFlow". https://es.wikipedia.org/wiki/OpenFlow#Ejemplo_1. [Online].

[11] Rouse M., "OpenFlow". https://searchdatacenter.techtarget.com/es/definicion/OpenFlow- [Online].

[12] McNickle M., "Cinco protocolos SDN que no son OpenFlow". https://searchdatacenter.techtarget.com/es/cronica/Cinco-protocolos-SDN-que-no-son-OpenFlow. [Online].

[13] SDxCentral Staff, "What is OpenFlow? Definition and How it Relates to SDN". https://www.sdxcentral.com/networking/sdn/definitions/what-is-openflow/. [Online].

[14] Open Networking Foundation ONF. "Software-Defined Networking: The New Norm for Networks. Rapport technique", abril 2013.

[15] OMF Corporation, "What is Mininet?". https://www.opennetworking.org/mininet/. [Online].

[16] ns- 3 Corporation, "What is ns-3?". https://www.nsnam.org/about/what-is-ns-3/. [Online].

[17]EstiNet Technologies, "Deliver Next-Generation Automated Networking Solutions to You

". https://www.estinet.com/es/?page_id=20671. [Online].

[18] Barona L., Valdivieso L. y Guamán D., "Una Nueva Alternativa a Mininet: Emulación de una Red Definida por Software usando VNX". https://journal.espe.edu.ec/ojs/index.php/cienciaytecnologia/article/view/102. [Online]

[19] InHo Cho, "Introduction to Mininet". https://github.com/mininet/mininet/wiki/Introduction-to-Mininet. [Online]

[20] Shakya S., “Running miniedit”. https://suganshakya.wordpress.com/2015/10/10/running-miniedit/. [Online].

[21] Mininet Community, “Mininet VM Setup Notes”. http://mininet.org/vm-setup-notes/. [Online].

[22] Github Community, Lantz. “Mininet VM Images”. https://github.com/mininet/mininet/wiki/Mininet-VM-Images. [Online].

[23] Pardo D., “¿Qué es NetFlow? ¡Aprovecha y descíbrelo de una vez por todas!”. https://pandorafms.com/blog/es/que-es-netflow/. [Online].

[24] IBM Knowledge Center, “sFlow”. https://www.ibm.com/support/knowledgecenter/es/SS42VS_7.3.0/com.ibm.qradar.doc/c_qradar_adm_sflow.html. [Online].