uc3m | Universidad Carlos III de Madrid

Grado en Ingeniería en Tecnologías de Telecomunicación

2017-2018

Trabajo Fin de Grado

"Desarrollo de solución SD-WAN basada en SDN"

Memoria

Álvaro Jiménez Moreno

Tutor/es

José Luis Iglesias Martínez

Antonio de la Oliva Delgado

Leganés, 2018

[Incluir en el caso del interés de su publicación en el archivo abierto]

RESUMEN

La forma de trabajar en Internet, tanto de usuarios como de empresas, ha cambiado en los últimos años. Se está pasando de un modelo basado en proveedores de datos que residían en partes localizadas de la red a otro en el cual cualquier punto de la red puede suponer un tráfico masivo de datos.

Este cambio de modelo está obligando a que los operadores comiencen a plantear soluciones escalables que soporten las tendencias futuras de tráfico en Internet.

Las empresas proveedoras de Internet están buscando un modelo lo suficientemente flexible que permita por un lado, asegurar sus negocios, y por otra, soportar cambios rápidos en las redes actuales y dar cabida a las necesidades de los usuarios de forma rápida y eficiente.

Actualmente, las tecnologías disponibles están dejando de ser efectivas ya que son, en su mayoría, sistemas rígidos que no tienen la suficiente capacidad para soportar las exigencias futuras de los clientes.

Aquí es donde entra en escena SDN, un nuevo paradigma que cambia completamente la forma de entender las redes. Hasta ahora el despliegue de las redes estaba, en su mayoría, basado en *switches*, que se encargaban de encaminar todo el tráfico a partir de su propia tecnología. Con SDN, sacamos el plano de control del hardware, obteniendo un mayor manejo del acceso a la red, manteniendo el plano de datos basado en la infraestructura del *switch*. Además, otro punto fuerte de SDN es su alta capacidad de virtualización.

Así, el modelo basado en hardware deja de ser factible, ya que cada componente requiere una configuración individual, con el fin de afrontar cambios en la red.

Este cambio de filosofía viene a solucionar el problema de flexibilidad existente actualmente ya que permite tener una visión global de la red desde un punto y gestionar la red desde él a través de algoritmos más inteligentes basados en *software*.

En este proyecto se empleará SDN, en concreto el protocolo *OpenFlow* para el despliegue y desarrollo de una maqueta de red virtualizada con el fin de evaluar esta tecnología.

Palabras clave: SDN; OpenFlow; ADIF; *Switch*; Maqueta; Red.

GLOSARIO

SDN	Red definida por *software*.
Tabla de flujos	Un conjunto de reglas de encaminamiento del tráfico.
OpenFlow	Protocolo emergente y abierto de comunicaciones que, a través de *software,* determina encaminamiento de tráfico a través de una serie de *switches.*
Open v*Switch*	*Software* para simular *switches* virtuales con el protocolo *OpenFlow*.
Controlador SDN	Equipo encargado del plano de control centralizado de la red. Encamina tráfico a través de los *switches.*
Ovs-ofctl	Interfaz de línea de comandos en el *Open vSwitch* que soporta e implementa el protocolo *OpenFlow* y su *API*.
GRE	Protocolo de establecimiento de túneles a través de internet.
IPsec	Conjunto de protocolos cuya función es asegurar las comunicaciones sobre el Protocolo de Internet (IP).
MPLS	Conmutación de etiquetas multiprotocolo.
API	Interfaz de Programación de Aplicaciones.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1. INTRODUCCIÓN

No hay duda en que, tanto la tecnología como internet, son dos elementos muy presentes en nuestra vida cotidiana. Comunicación, información, entretenimiento, seguridad, banca, orientación, compra, estadística, multimedia, etc. Todos estos elementos, a día de hoy, han sido en su mayoría trasladados a internet, por lo que su importancia presenta un incremento continuo.

Tal es la importancia que ha adquirido Internet, que es esencial asegurar un servicio continuo y de calidad, tanto a clientes particulares, como a empresas. Actualmente, la mayoría de los negocios empresariales tiene n su base en internet, por lo que cualquier tipo de fallo en la infraestructura puede suponer grandes pérdidas.

Para soportar esta gran demanda futura de Internet, los operadores han empezado a trabajar e investigar formas que les permita asegurar el acceso a la red y así responder a la necesidad de sus clientes.

Generar esta respuesta no es fácil, ya que se deben utilizar soluciones que permitan generar un modelo de negocio rentable para los operadores y que tengan en cuenta las tendencias de precios y los hábitos de los usuarios de Internet.

Por ello, es importante empezar a investigar nuevas tecnologías y filosofías de red que faciliten esta transición respecto a Internet. Estas nuevas tecnologías y filosofías deben ser afrontadas estrictamente, ya que dar acceso a millones de personas a Internet supone una tarea compleja.

En este proyecto se empleará SDN, en concreto el protocolo *OpenFlow*, para el despliegue y desarrollo de una maqueta de red virtualizada con el fin de evaluar esta tecnología y su viabilidad en la red de ADIF.

2. RED IP DE ADIF

2.1. Estructura y servicios ofrecidos.

La red IP de Sistemas de Información de ADIF es la red que soporta los servicios TIC de ADIF y la mayor parte de los de Renfe. Los servicios que actualmente presta la red IP Corporativa de ADIF son consecuencia de la evolución de una antigua red de Teleproceso, que principalmente ofrecía los siguientes servicios:

- SNA con terminales punto a punto.
- Accesos Remotos Conmutados.
- Conexiones X25.
- Interconexión con otras redes.
- Redes locales.

La implantación de tecnología TCP/IP permitió integrar servicios de comunicaciones disjuntos sobre una misma red y con mejores prestaciones. En la actualidad, la red IP Corporativa de ADIF ha evolucionado hacia una red multiservicio sobre distintas tecnologías de acceso. Los principales servicios prestados sobre esta red son:

- Transmisión de datos de alta velocidad.
- Control de acceso a la red.
- Soporte multiprotocolo.
- Aplicaciones intranet, extranet e internet.
- Correo electrónico.
- Videovigilancia.
- Videoconferencia.
- Telefonía IP.
- Megafonía.
- Teleindicadores.

La Red IP de Sistemas de Información tiene cerca de 3.500 puntos de presencia y más de 25.000 usuarios conectados. Está interconectada tanto con otras redes propias de ADIF (Multiservicio, AVE, etc.) como con redes externas (Red SARA, Vodafone, Renfe Operadora, etc.) e internet.

La red IP de ADIF está concebida como una estructura jerárquica con la siguiente topología de nodos:

- Centros de usuario: En adelante CU, aproximadamente unos 2.000, distribuidos por la geografía española. Representan los puntos de acceso a la red para los usuarios.
- Centros intermedios: En adelante CI, agregan el tráfico de los centros de usuario. Forman parte de la red troncal. Al menos uno por provincia.
- Centros de Zona: En adelante CZ, concentran el tráfico de los centros intermedios. En total existen nueve, forman parte de la Red Troncal.
- CPD o nodos singulares: Son centros de zona ubicados en los CPDs de ADIF: Delicias y Villaverde, que son destino de la gran mayoría del tráfico generado por los centros de usuario. Estos nodos forman parte de la red troncal.

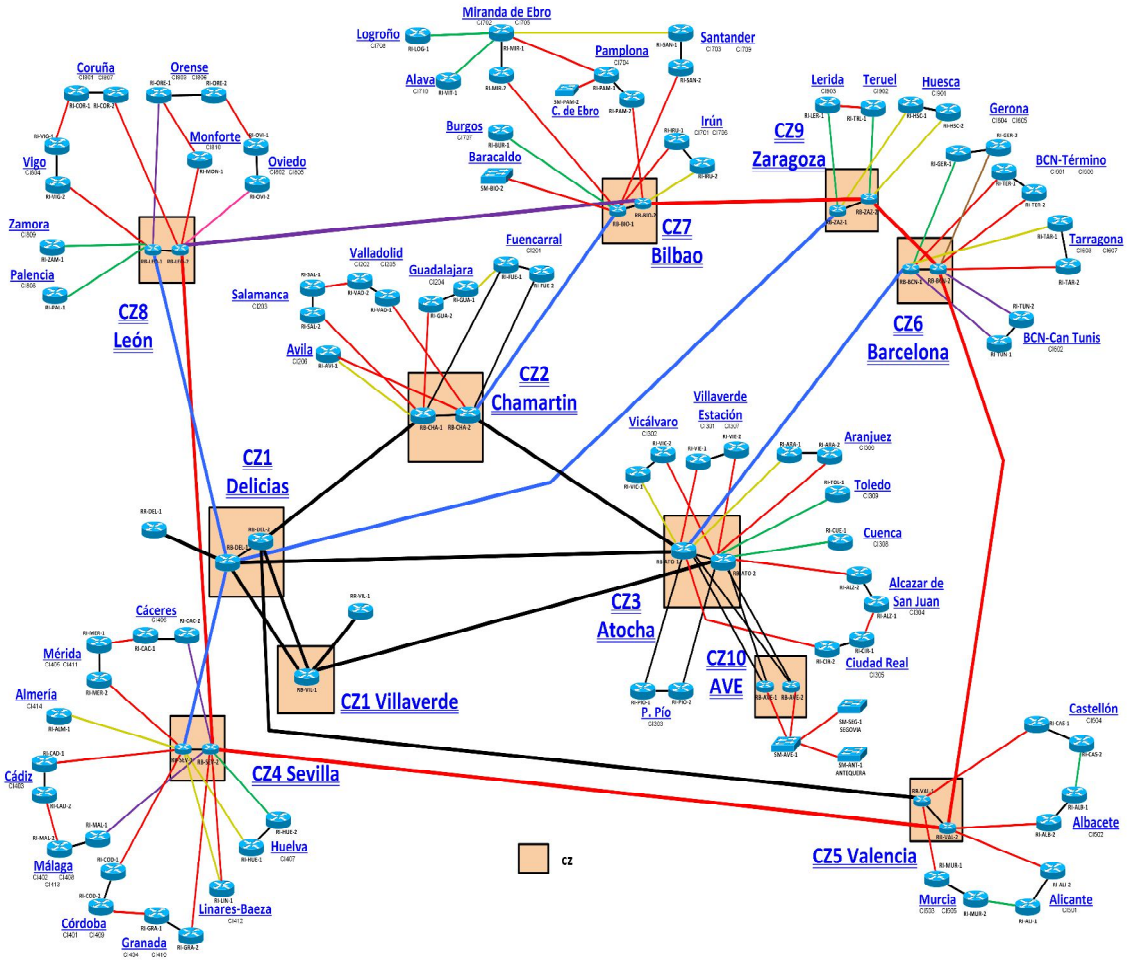


Fig. 2.1. Red IP Corporativa de ADIF [1].

2.2. *Backbone*

Como ya se ha hablado en el apartado 3.1 y visto en la figura 3.1, la red IP de Sistemas de Información de ADIF tiene una estructura jerárquica que ha ayudado a su crecimiento durante los años siguientes a su implantación.

En 2004, dado el crecimiento que estaba teniendo la red y la necesidad de obtener una mayor diferenciación de servicio para las distintas unidades de negocio, empresas externas y organismos gubernamentales decidieron implementar un protocolo recién formulado: MPLS.

Con la implantación de dicho protocolo en el *backbone*, se conseguía unificar y jerarquizar aún más la red, dando la posibilidad de facilitar servicios de mayor calidad

como VPNs. Siguiendo las especificaciones marcadas por el IETF, se estableció un diseño preliminar siguiendo la estructura marcada en la siguiente figura:

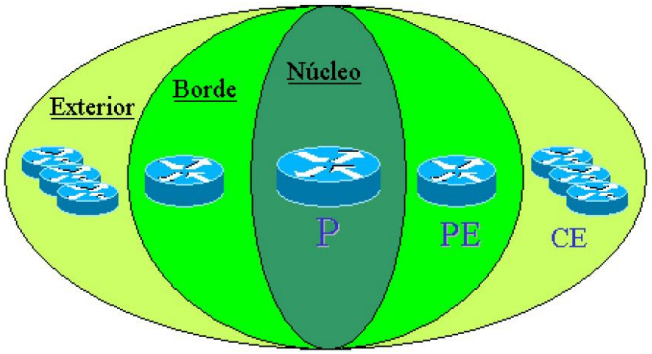


Fig. 2.2. *Backbone* de la red IP de ADIF con la integración de MPLS [1].

El núcleo representa la parte central del *backbone*, constituido principalmente por *routers* de agregación con gran cantidad de líneas de alta capacidad, que harán lo que se denomina "función de P" (*Provider* - Proveedor). Estos *router* no enrutan tráfico IP, si no tráfico ya etiquetado como MPLS. En particular podemos identificar la función de P desarrollado por equipos situados en los CZ.

El borde es la parte del *backbone* que rodea al núcleo y está compuesto por *routers* de acceso que constituyen el borde de la red, haciendo "función de PE" (*Provider Edge* - Proveedor de Borde). A estos *routers* se conecta, por una parte los *routers* del núcleo, y por otra, los *routers* externos a la red (típicamente CEs). Estos *routers* PE tienen la función de encapsular y desencapsular el tráfico IP con cabeceras MPLS; enrutar tráfico hacia los equipos de la capa exterior y tráfico MPLS hacia los equipos del núcleo. La función de PE es desarrollada sobre todo por los equipos que actúan como CI, pero existen casos de equipos en CZ que desarrollan de manera simultánea las funciones de P y PE.

La capa exterior está compuesta por los equipos que no forman parte del *backbone* en sí, haciendo "función de CE" (*Customer Edge* – Borde del Cliente). Estos *routers* serán completamente ajenos al MPLS puesto que sólo enrutarán tráfico IP. Se identifican

como los equipos marcados como CU. La estructura descrita habilita servicios de VPNs sobre MPLS, apoyándose en BGP multiprotocolo para la articulación de este servicio. En este sentido, todos los *routers* en funciones de *Provider Edge* mantienen *peerings* de BGP multiprotocolo entre ellos, en una estructura completamente mallada.

2.3. Conexión física entre nodos.

De igual manera que la red tiene una estructura jerárquica, los circuitos que la soportan también tienen una estructura jerárquica.

Los circuitos de red troncal utilizan tecnología SDH y DWDM, y son utilizados para unir los CZ entre ellos, los CZ con los CI y los CI entre ellos.

Para la interconexión de la red troncal con los CU se utilizan diversas tecnologías. Además, la elección del medio físico para la conexión depende de diversos factores, como puede ser la distancia desde la ubicación de la red troncal a la ubicación de CU o las características geográficas. La práctica usual es establecer una conexión principal acompañada de una de *backup*. Se permite casi cualquier combinación de las siguientes tecnologías:

- Fibra óptica monomodo (mm) utilizando para ello el estándar 1000BASE-LX/LH.
- Fibra óptica multimodo (MM). A evitar, ya que no está soportada por los equipos habituales de mantenimiento. Se utiliza el estándar 1000BASE-SX.
- UTP categoría 5, 5e ó 6, donde la distancia lo permite (< 100 m). En este caso 100/1000BASET.
- Circuitos PDH de 2 Mbps. Trama E1 estructurada o sin estructurar entregada en interfaz V.35 o G.703.
- Circuitos PDH/SDH de $n \times 2$ Mbps, o VC3. Entrega en Ethernet 10/100BASE-T.
- Cable de pares o de cuadretes. En este caso se utilizan módem G.SHDSL para poder llevar circuitos Ethernet a más de 100 metros sobre este medio.
- Conexión a red móvil 3G/4G.
- Conexión a Red de Operador de Telecomunicaciones externo mediante circuitos punto a punto o servicios VPN de nivel 2 y 3 (tipo Macrolan de Telefónica o

VPLS de Vodafone). En este caso el interfaz será 100/1000BASE-T o 1000BASE-LX/LH.

Actualmente, el *backup* de los CU se están realizando mediante conexión 3G/4G. Se hace mediante tarjetas SIM de Vodafone (vigente proveedor de comunicaciones móviles) incorporadas a los *routers*. En la figura 3.3 se puede encontrar un ejemplo de un CU con *backup* 3G/4G. Está previsto utilizar accesos "residenciales" a internet para proveer la conexión de *backup*, y en casos excepcionales, la principal.

Los equipos y terminales finales de usuario casi siempre se conectan en Ethernet (IEEE 802.3) 10/100BASE-T mediante un cable UTP de categoría 5, 5e O 6 a un *switch* denominado LU (LAN de Usuario). El *WiFi*, donde está disponible, es también un medio de acceso utilizado. Estos *switches* se conectan en diferentes topologías, preferiblemente en árbol o sino en cascada con un número máximo de saltos hacia el equipo que da la salida a la red troncal, que se denomina RU (Router de Usuario) o SU (*Switch* de Usuario). La conexión de los LU a los RU/SU se hace mediante las siguientes tecnologías:

- Cable de *stack* propio de cada suministrador.
- Fibra óptica MM monomodo utilizando para ello el estándar 1000BASE-LX/LH.
- Fibra óptica MM multimodo. A evitar, ya que no está soportada por los equipos habituales de mantenimiento. Se utiliza el estándar 1000BASE-SX.
- UTP categoría 5, 5e o 6, donde la distancia lo permite (< 100 m). En este caso 100/1000BASET.
- Cable de pares o de cuadretes. Como en el caso de los circuitos de red de acceso se utilizan módem G.SHDSL para poder llevar circuitos Ethernet a más de 100 metros sobre este medio.

En los centros de usuario suelen convivir usuarios que pertenecen a diferentes VPN, estando configuradas en la red actualmente:

- ADIF Corporativa
- ADIF Seguridad
- Renfe

En estos casos los usuarios están separados en diferentes VLANs y cada VLAN es asignada a una VRF independiente en el subinterfaz del PE al que se conecta el CU.

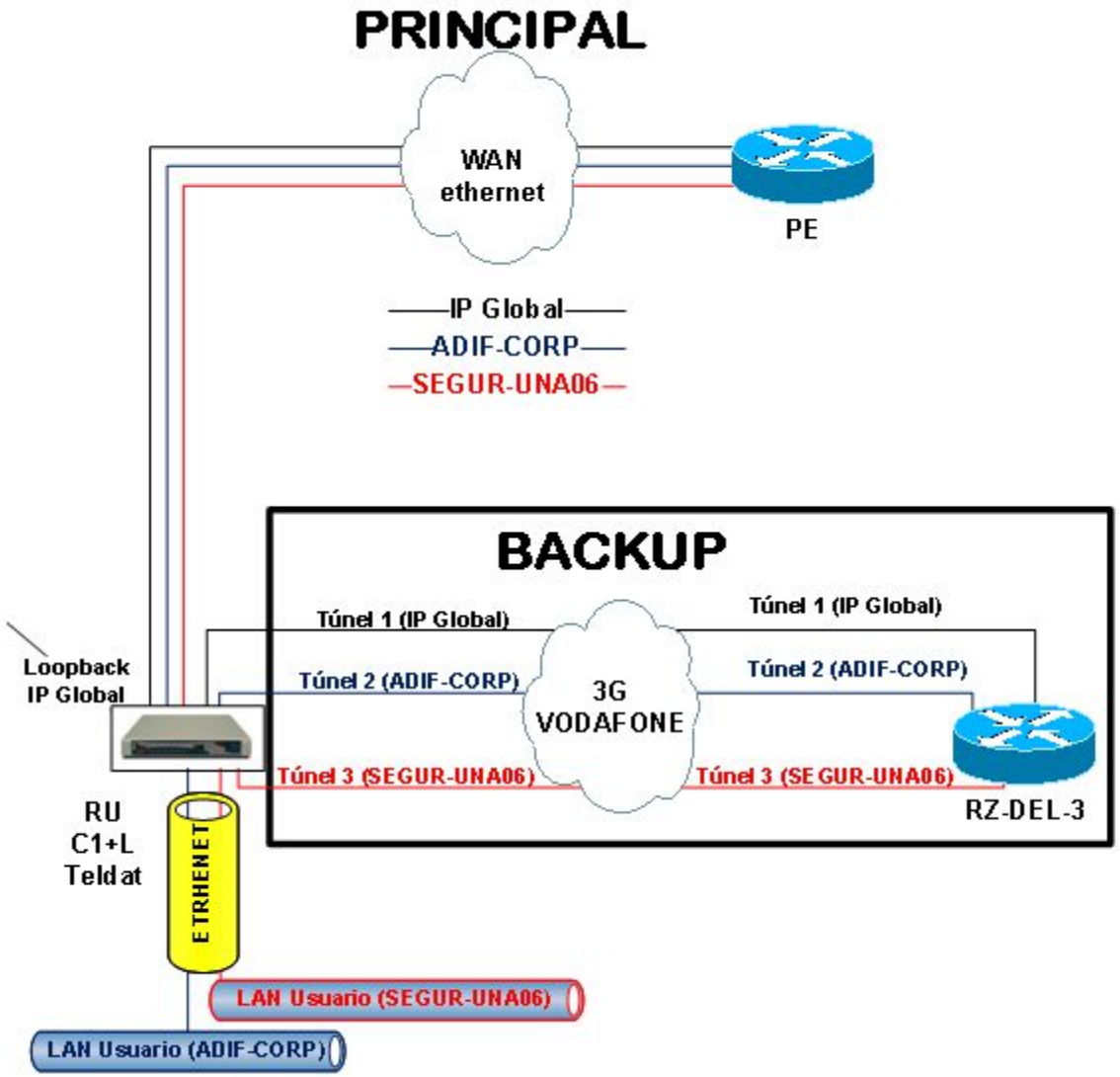


Fig. 2.3. Centro de Usuario con 3 VPNs y con backup 3G [1].

3. MOTIVACIÓN Y OBJETIVOS

3.1. Motivación del proyecto.

3.1.1. Internet, en continuo crecimiento.

Las redes de Internet están evolucionando mucho en los últimos años. Las necesidades y costumbre de los usuarios han cambiado y Internet cada vez está más integrado en nuestra vida.

Antaño, las redes se utilizaban principalmente para servicios de correo electrónico, webs sencillas basadas en HTML o descarga de pequeños datos (fotografías, documentos,..), por lo que el volumen de tráfico era cuantitativamente menor.

Esta situación ha ido cambiando y se prevé que cambie aún más debido a la tendencia que lleva Internet de que la mayoría de tráfico sea multimedia. Como se puede apreciar en las gráficas inferiores, se prevé un aumento del porcentaje de usuarios que accede a internet hasta un 58%, así como un aumento del número de dispositivos, velocidad y, por tanto, de tráfico en las redes.

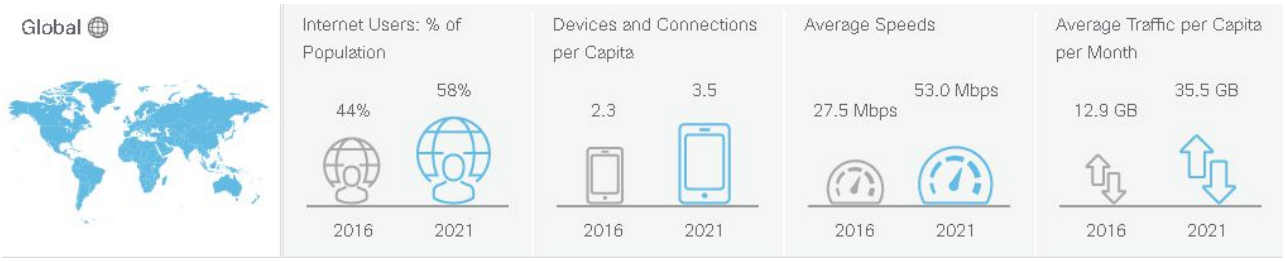


Fig. 3.1. Cisco Visual Networking Index – Reresentación del uso y características de Internet (2016-2021) [2].

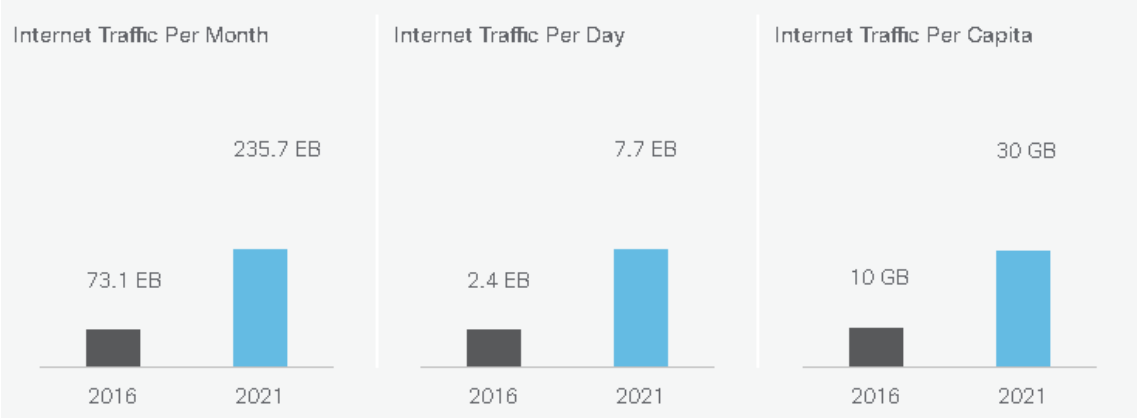


Fig. 3.2. Cisco Visual Networking Index – Representación del tráfico de Internet (2016-2021) [2].

Este aumento gradual del tráfico supone la necesidad de una mejora de las redes y del control de las mismas. A mayor volumen de tráfico, mayor posibilidad de saturación de enlaces, de caídas o la imposibilidad de los enrutadores de manejar tal cantidad de tráfico. Frente a esta situación, se plantean varias soluciones:

- Realizar cambios en la red mediante la adquisición de nuevos equipos capaces de soportar este aumento del volumen de tráfico, lo cual conllevaría una gran inversión por parte de los operadores, así como la costosa modificación y reconfiguración de la red para integrar este nuevo hardware. Además, esto, a su vez, puede implicar un aumento del precio de los servicios de Internet.
- Invertir en mejorar la eficiencia de las redes actuales a través de la investigación de nuevas técnicas y/o protocolos de enrutado de datos. Esto puede suponer un coste progresivo, no tan agresivo como con la compra de nuevo hardware. Y más importante, implementar una modificación de *software* no resultaría tan agresiva como cambiar todos o la mayoría de equipos de una red.

3.1.2. Tráfico multimedia.

Como se ha mencionado anteriormente, la mayor parte del tráfico de internet hoy en día es contenido multimedia. Como ejemplo, podemos observar el aumento de tráfico de vídeo previsto para 2021:

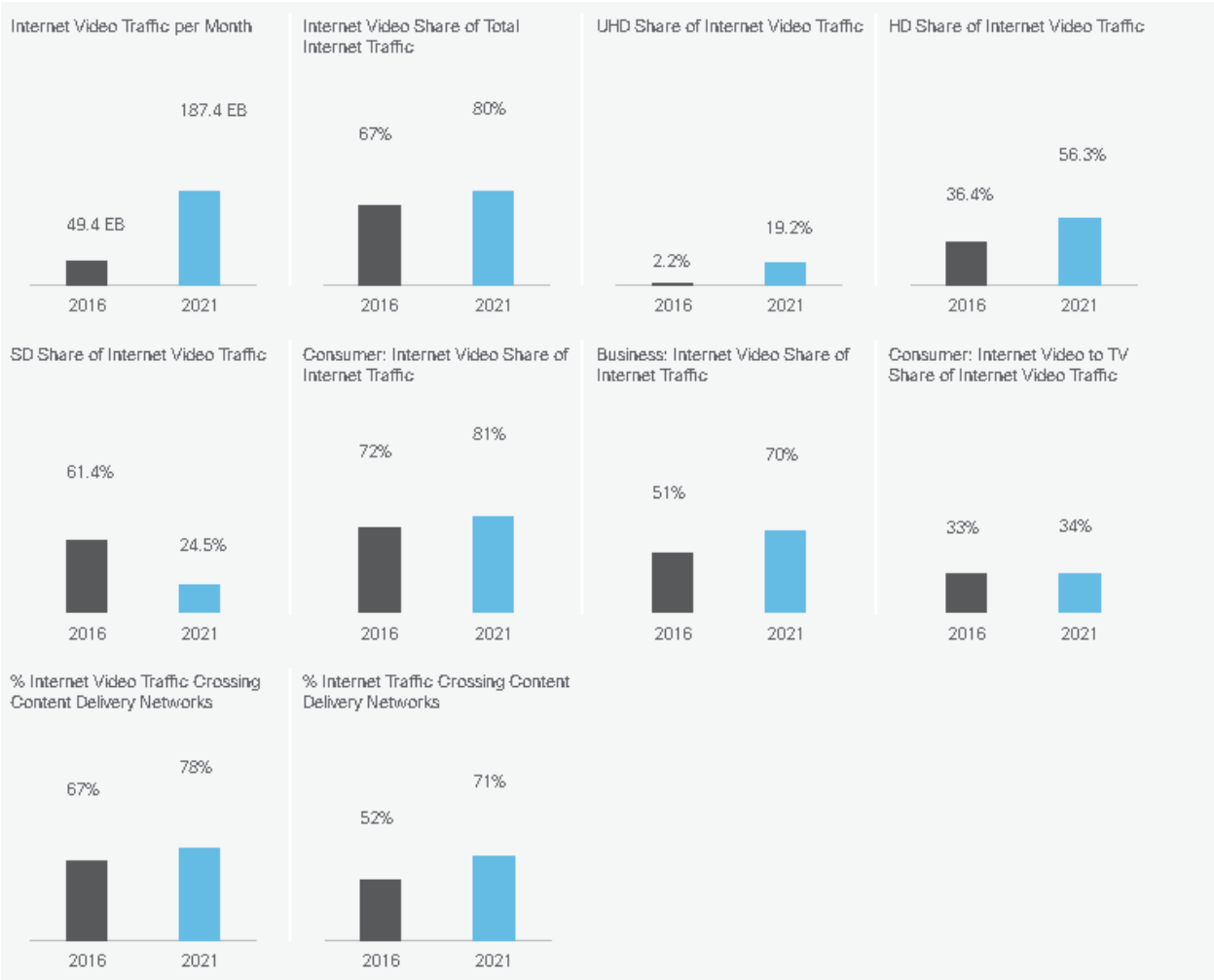


Fig. 3.3 Cisco Visual Networking Index – Representación del tráfico de vídeo de Internet (2016-2021) [2].

El comportamiento de este tipo de tráfico difiere del de otras aplicaciones más tradicionales [3]:

- Requerimiento de un gran ancho de banda: las transmisiones de vídeo, por ejemplo, requieren de mayor ancho de banda que las transmisiones de audio. Además, este requerimiento puede no ser constante durante toda la conexión.
- Flujos de datos muy largos: el tráfico multimedia suele tener flujos continuos de minutos e incluso horas, haciendo un uso más intenso de la red, no como en el caso de otros tipos de tráfico como el web, que suele ser intermitente.
- Gran número de conexiones: puede haber varias conexiones multimedia abiertas simultáneamente y cada una de estas conexiones puede requerir de varios flujos

de datos. Además, con la llegada de las multi conferencias o las emisiones en streaming ha surgido la necesidad de abrir conexiones entre varios orígenes y destinos lo que aumenta la demanda de recursos.

- Sensibilidad al *jitter* o latencia: las tramas de vídeo, por ejemplo deben tener un periodo de llegada regular. Tramas que llegan demasiado pronto requieren mayor capacidad de almacenamiento, al tener que ser guardadas por el destino. Por otro lado, tramas que llegan con retraso, deben ser descartadas, dismunuyendo la calidad del vídeo.
- Exigencia de una gran calidad de servicio: el usuario cada vez exige mayor calidad de visionado de video, tanto en plataformas multimedia como Youtube, por ejemplo, como en servicios de videoconferencia o streaming. Por ejemplo, Skype requiere, como mínimo, 1,2 Mbps de ancho de banda para videollamadas en alta definición aunque 1,5 Mbps es lo más recomendable [4], el cual aumenta para videoconferencias entre más usuarios.

Por tanto, la necesidad de redes más rápidas y eficientes con el fin de soportar todas estas necesidades relativas al contenido multimedia es evidente.

3.1.3. Demanda de los usuarios.

Es evidente que la mayoría de los cambios que se llevan a cabo en las redes se ven fuertemente influenciados por las exigencias de los usuarios finales, siempre y cuando esto genere un beneficio para el operador o empresa. Podemos observar una tendencia clara:

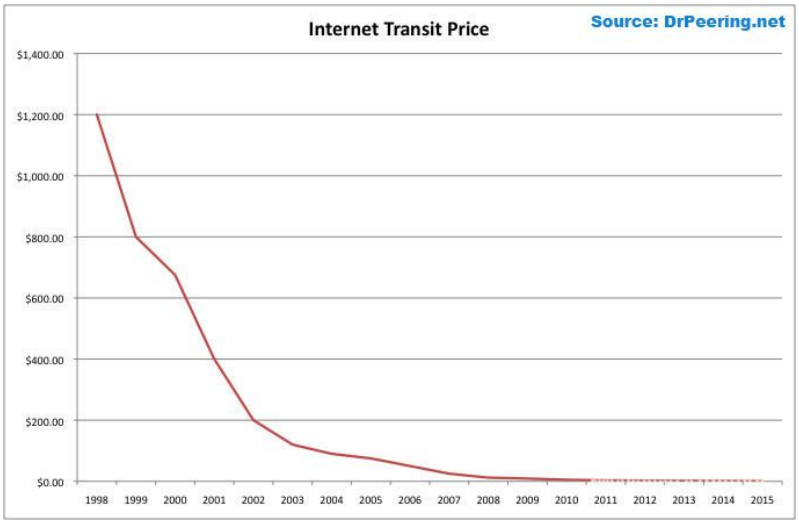


Fig 3.4 Evolución del precio de internet en los últimos años (hasta 2015) [5].

Se puede observar que, desde 2010, el precio de tránsito de internet tiende a 0. Esto se debe principalmente a la apertura de internet al público.

Además la demanda de conexión por parte del usuario final también ha cambiado. Actualmente, existe una demanda de conexiones de 50-100 Mbps para servicio residencial. Algunos operadores ya han decidido preparar nuevos servicios de 200 Mbps [5] para adelantarse a la futura demanda que, probablemente, existirá.

Por tanto, la tendencia es clara, ofrecer un servicio de menor precio y mayor calidad.

3.1.4. Redes actuales.

Las redes han evolucionado cualitativamente en los últimos años permitiéndonos obtener grandes velocidades de acceso y transferencia de datos. Actualmente, disponemos redes que permiten ofrecer hasta 200Mbps en hogares y conexiones de hasta 100Gbps para empresas con servicios como MacroLan de Telefónica [6].

Esta evolución se debe a la introducción de nuevas tecnologías, como la fibra óptica, la cual aumenta notablemente la velocidad de conexión. Sin embargo, pese a este aumento de la velocidad, aún existen limitaciones, sobre todo en los nodos de interconexión:

- El hardware encargado del encaminamiento funciona con electrónica convencional lo que conlleva a que la velocidad de encaminamiento sea inferior a la velocidad de llegada del tráfico.
- La mayor parte de los protocolos actuales no tienen una visión completa de la red por lo que no utilizan sus recursos al completo.

3.1.5. ¿Existe una necesidad de cambio?

Por tanto, tras lo expuesto hasta ahora, podemos realizar la siguiente enumeración relativa a internet:

- Volumen de tráfico en continuo crecimiento. Mayor uso de internet a nivel doméstico y aumento de aplicaciones y servicios multimedia para el usuario.
- Demanda de conexiones más rápidas, seguras y robustas.
- Necesidad de unas características y condiciones específicas para un correcto funcionamiento de los servicios multimedia (*jitter* bajo, mayor ancho de banda,…).
- Abstracción en cuanto a los protocolos de enrutamiento se refiere.

Todo esto sugiere una evolución de las redes de comunicaciones, sobre todo orientada al aprovechamiento de los recursos y al manejo de la misma.

Además, esta evolución debe conservar la tendencia de ofrecer cada vez un servicio de mejor calidad a un menor precio para el usuario, haciendo menos viable soluciones que requieran grandes inversiones iniciales que conlleven a una subida gradual de los precios.

3.2. Objetivos.

El objetivo de este proyecto consiste en proporcionar una solución SDN (Red definida por *software*) de la red de comunicaciones de ADIF mediante *software* libre, en nuestro caso *OpenFlow*, difiriendo de otras soluciones comerciales como puede ser la de Cisco o Teldat. Se desplegará una maqueta virtualizada en tres equipos sobre la cual se realizarán pruebas de rendimiento y encaminamiento a fin de realizar comparaciones finales. Mediante esta solución, se dispondrá de una segunda opción de encaminamiento y gestión de acceso a la red, diferente al *routing* clásico, entre los servicios proporcionados por el Centro de Procesamiento de Datos (CPD) de ADIF y todas las sedes remotas de usuario.

SDN se ha propuesto como una solución cuyo fin es aumentar la eficiencia en el uso de las redes, pasando de un modelo de control distribuido a uno centralizado en un punto (controlador).

A lo largo del trabajo, se explicará en qué consiste SDN y cuál es la idea principal de las redes basadas en él, así como el funcionamiento, más concretamente, del protocolo *OpenFlow*, para posteriormente realizar el despliegue de la maqueta virtualizada.

4. PLANTEAMIENTO DEL PROBLEMA, ESTADO DEL ARTE, REQUISITOS, RESTRICCIONES Y MARCO REGULADOR.

4.1. Planteamiento del problema.

Como se ha comentado, el principal problema reside en el aumento del tráfico, sobre todo el multimedia, lo que conlleva a un aumento de la demanda y, por consiguiente, a una renovación y/o mejora de las redes de comunicaciones actuales. ¿Cuáles son las tendencias que han desencadenado esta demanda?

- Cada vez mayor demanda de una mejor calidad de vídeo y audio por parte de los usuarios.
- Cada vez mayor implantación de internet en el ámbito doméstico. En España, actualmente un 71,2% de los hogares posee acceso de banda ancha fija, 17,2 puntos porcentuales más que en 2010 (54%).

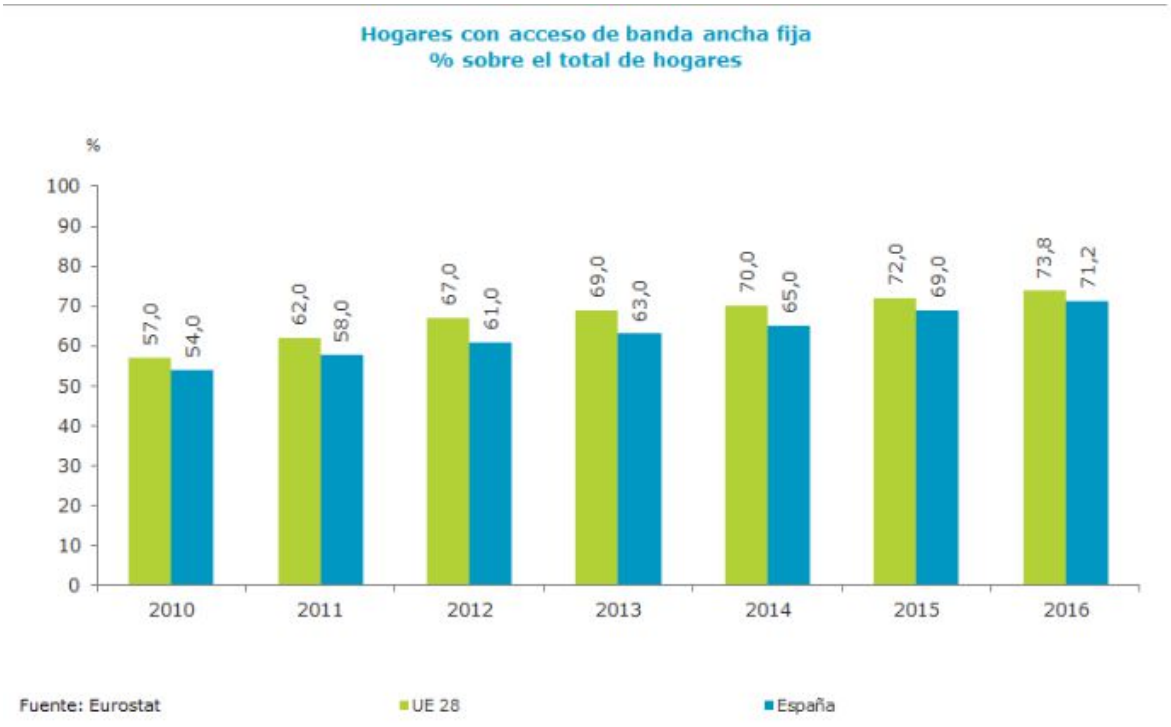


Fig. 4.1 Porcentaje de hogares con acceso de banda ancha fija en España y la Unión Europea [7].

- Cada vez mayor accesibilidad a la red debido, principalmente, a la llegada de *smartphones, tablets, Smart TVs*, etc.

- Incremento anual del porcentaje de ancho de banda usado para el acceso a contenido multimedia, sobretodo vídeo.

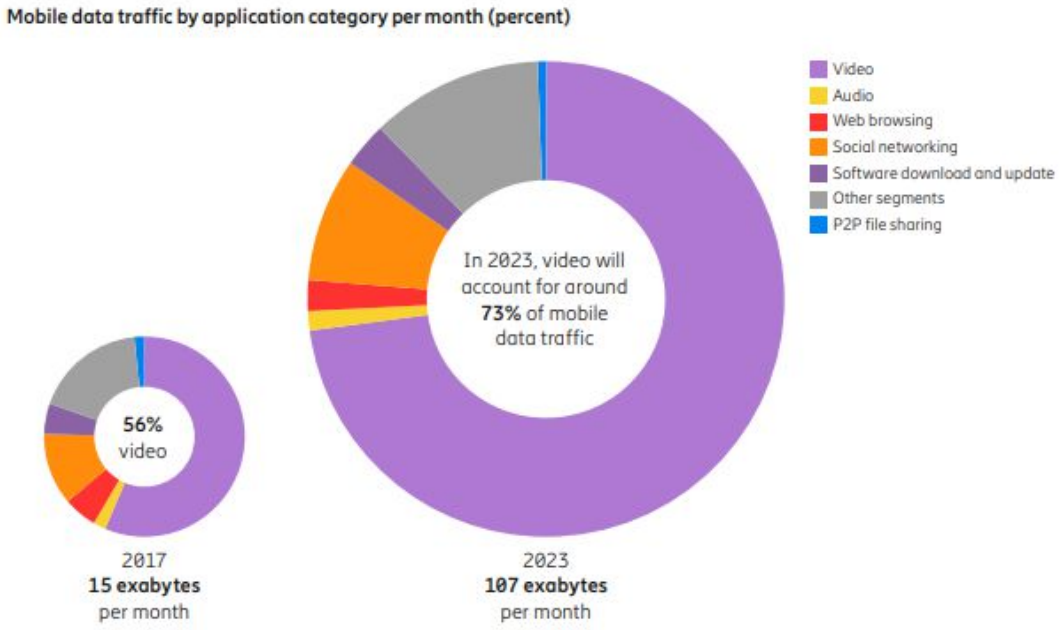


Fig. 4.2 Tráfico de datos móviles por categoría (2017- 2023) – *Ericsson Mobility Report* [8].

En el caso de que las previsiones se cumplan, ¿se podrá ofrecer una calidad de servicio buena al usuario? ¿se podrá hacer sin costes adicionales para el usuario? Lo que parece irremediable es la necesidad de un cambio en el enfoque y manejo de las redes de comunicaciones actuales.

4.2. Estado del arte.

4.2.1. Introducción.

Las redes definidas por *software* suponen un nuevo paradigma introducido en el mundo de la interconexión de computadoras, el cual promete un cambio fundamental en el modo en el que la configuración de las redes y el manejo del tráfico a tiempo real es tratado. A pesar de ser un término relativamente actual, se puede rastrear la historia de SDN en las raíces de varios mecanismos de control de redes y de ingeniería de tráfico desarrollados a través de los años. El objetivo de derivar los sistemas de control de las redes centralizadas siempre ha buscado mejorar el rendimiento general de la red e

introducir mayor grado de control en, por lo menos, un segmento particular de una red mucho más grande. SDN es visto, por muchos de la industria, como una culminación de estos objetivos.

La Open Networking Foundation (ONF), consorcio que lleva a cabo varios trabajos en áreas del desarrollo SDN, define el término SDN como 'la separación física del plano de control del plano de encaminamiento y en la cual un plano de control controla varios dispositivos'. La infraestructura de SDN trata de hacer que el plano de datos sea totalmente programable y separado de la lógica de control y, por consiguiente, elimina el intenso trabajo de configuración de hardware. El paradigma introduce una estructura de control centralizada la cual configura dinámicamente y dirige todo el hardware subyacente basándose en los requerimientos finales del usuario. Los desarrolladores de *software* pueden utilizar la abstracción de la red a partir del plano de control para definir modelos de utilización de recursos de la red sofisticados y optimizar la estructura de la red acorde a los requerimientos de una red en crecimiento. La facilidad resultante en el manejo de diversas aplicaciones en la red de acuerdo con las condiciones del tráfico a tiempo real proporciona beneficios sustanciales a los operadores e introduce actualizaciones tecnológicas y de negocio de una manera continua. Así, SDN está atrayendo sustancialmente la atención de profesionales tanto académicos como de la industria .Sin embargo, a pesar de las ventajas y la promesa de una gestión más simplificada de las redes, SDN encuentra dificultades en su implementación práctica lo cual obstaculiza su funcionalidad y rendimiento en el camino entre los centros de datos y la nube.

4.2.2. ¿Qué son las redes definidas por *software*?

Existen varias visiones acerca de qué son las redes definidas por *software*. Según Nick McKeown, pasa por una refactorización de la funcionalidad, es decir, definir las redes definidas por *software* por la disposición de su funcionalidad; actualmente protocolos cerrados en 'cajas'. Cada 'caja' consta de su sistema operativo, su *API* (interfaz de programación de aplicaciones) y su hardware para manejar paquetes. SDN trata de abrir estas 'cajas', consiguiendo un único sistema operativo extensible, a ser posible, de código abierto (open-source) y una *API* general bien definida [9].

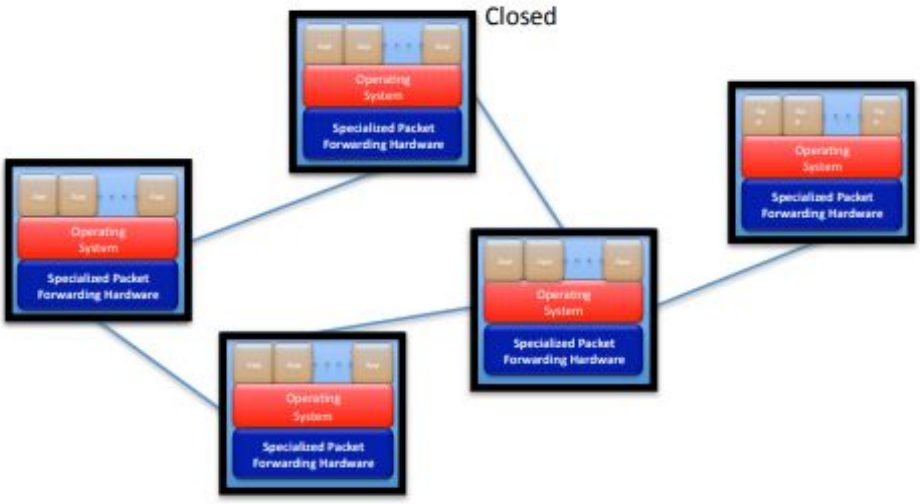


Fig. 4.3 ‘Cajas’ de protocolos cerrados [9].

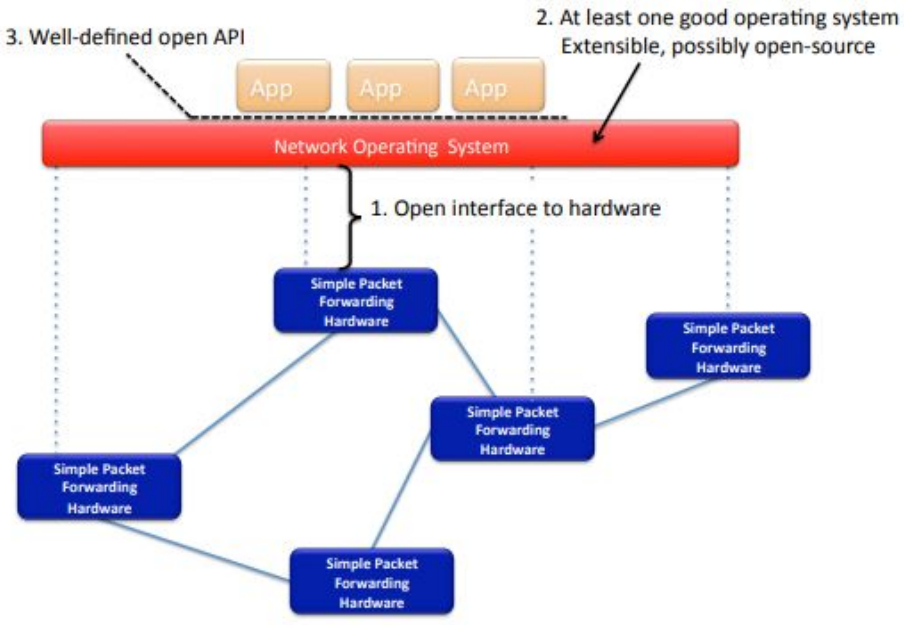


Fig. 4.4 Apertura de las ‘cajas’ de protocolos mediante SDN según Mckeown [9].

Scott Shenker define SDN a partir de las abstracciones que proporciona al *software* (y a la gente que lo escribe) [9].

La visión de Shenker se basa principalmente en la necesidad de abstraer el plano de control de la red con el fin de resolver problemas de arquitectura y habilitar una evolución de las mismas. Las redes funcionan porque se puede manejar la complejidad pero lo que se debe hacer es extraer la simplicidad con las abstracciones correctas.

Por tanto, se puede decir que la programación causó la transición de las redes tradicionales basadas principalmente en hardware a las redes definidas por *software*. Además, las abstracciones simplifican esta programación, haciendo que sea más sencilla de escribir, mantener y razonar.

Otra manera de ver SDN es únicamente mediante la flexibilidad que nos ofrece. Las redes definidas por *software* ofrecen 'ejes' en cuanto a diseño e implementación. Por lo que cualquier red de este tipo es aquella que nos otorga la flexibilidad para elegir cualquier punto en esos 'ejes'. Por ejemplo, centralizado vs distribuido, *microflow* vs flujos agregados, reactivo vs proactivo, físico vs virtual,…

Fig. 4.5 'Ejes' de diseño de SDN [9].

Son estos ejes los que se podrán variar a la hora de diseñar redes definidas por *software*.

4.2.3. Requerimientos para SDN.

El requerimiento fundamental de una infraestructura la cual cumpla con una serie de requerimientos operacionales, disponga de una fácil programabilidad, de un despliegue dinámico y de un fácil abastecimiento, mientras facilita aplicaciones innovadoras, dictó un nuevo paradigma capaz de satisfacer estos requerimientos. Algunas preocupaciones las cuales desembocaron finalmente en la infraestructura SDN se detallan a continuación:

- Automatización: un nivel elevado de automatización el cual reduzca el gasto operacional general y facilite un diagnóstico de problemas efectivo, disminuyendo la inactividad no planeada, facilitando la ejecución de políticas, y provisionando recursos de la red.
- Manejo dinámico de recursos: cambiar dinámicamente el tamaño de la red y actualizar la topología y los recursos asignados, tarea que será facilitada mediante la virtualización de la red.
- Orquestación: orquestar el control de un conjunto completo de aplicaciones como en centros de datos o en redes grandes de campus.
- Soporte de control por una sola instancia del programa: con la creciente proliferación de servicios basados en la nube, los ocupantes prefieren un control completo de sus direcciones, topología, enrutamiento y seguridad y es por ello por lo que se separa la infraestructura de los ocupantes de los servicios anfitriones.
- *APIs* abiertas: usuarios con la posibilidad de acceder a programas modulares, ofreciendo abstracción frente a las redes, definiendo tareas a partir de las *API* y no específicamente preocupados sobre los detalles de implementación, es decir, la comunicación entre dos nodos puede ser suministrada sin la especificación de un protocolo exacto.
- Gran programabilidad: un requerimiento fundamental de SDN es poder cambiar el comportamiento y la configuración de dispositivos a tiempo real ante condiciones de tráfico predominantes.

- Seguridad integrada: la habilidad de integrar dispositivos de seguridad dentro de la estructura de la red, conduce a una mayor precisión a la hora de detectar incidentes de seguridad y a una simplificación de su manejo.
- Manejo de recursos integrado: además de los dispositivos de seguridad, pueden ser integrados diversos servicios como balanceadores de potencia y monitores de recursos cuando sean requeridos.
- Rendimiento mejorado: una infraestructura de control capaz de incorporar soluciones nuevas en cuanto a ingeniería de tráfico, cálculo de capacidad, balance de carga,…
- Virtualización de la red: la habilidad de provisionar recursos sin preocupaciones sobre la localización física de componentes individuales como routers o *switches*.
- Visibilidad y monitorización a tiempo real: mejorar la monitorización a tiempo real y la conectividad entre dispositivos.

Una visión centralizada de la red distribuida a través del plano de control de SDN proporciona una orquestación y una automatización de los servicios más eficientes. Mientras que los protocolos tradicionales pueden reaccionar después de que los servicios estén en línea, SDN puede prever requerimientos adicionales de la red y tomar medidas proactivas para alocar recursos. Además, las aplicaciones de red basadas en SDN, distribuyen políticas definidas por el usuario muy granuladas sobre flujos de tráfico por aplicación [10].

4.2.4. Arquitectura de SDN.

La arquitectura básica de SDN está basada en abstracciones, bastante similares a los métodos formales de ingeniería del *software*. Una arquitectura de red típica basada en SDN divide procesos como la configuración, el alojamiento de recursos, la priorización del tráfico y el encaminamiento del mismo en el hardware subyacente en tres planos básicas: aplicación, control y datos. Cada uno de los planos tiene unos límites bien definidos, un papel específico y unas *APIs* concretas para comunicarse con los planos

adyacentes. Los componentes principales de la infraestructura son, por tanto, los siguientes:

- Plano de datos: el plano de datos es un conjunto de componentes de red los cuales pueden ser *switches*, routers, equipamiento de red virtualizado, firewalls,…El único propósito del plano de datos es encaminar el tráfico de la forma más eficiente posible, basándose en una serie de reglas de encaminamiento creadas por el plano de control. La arquitectura SDN elimina toda la inteligencia relativa al encaminamiento y la configuración aislada de cada dispositivo, moviéndolas al plano de control. La comunicación entre los planos de control y datos se realiza a través de las APIs.

- Plano de control: el plano de control es el responsable de tomar decisiones acerca de cómo debe ser enrutado el tráfico a través de la red desde un nodo en concreto hasta otro, basándose en los requerimientos del usuario final y en las políticas de red resultantes. El componente principal del plano de control es el controlador SDN. Un controlador SDN traduce los requerimientos de la red como la necesidad de priorizar tráfico, control de acceso, manejo del ancho de banda, QoS,…en reglas de encaminamiento relevantes las cuales están conectadas con los componentes del plano de datos. En función del tamaño de la red, puede existir más de un controlador. El hecho de introducir la programabilidad a través del plano de control, hace posible modificar tablas de flujos en elementos individuales a tiempo real, basándonos en el rendimiento de la red y en los servicios requeridos. El controlador da una visión clara y centralizada de la red subyacente siendo una herramienta de manejo muy potente.

- Plano de aplicación: el plano de aplicación consta de aplicaciones de red específicas. Una visión abstracta de la red es presentada a las aplicaciones a través de las APIs. El nivel de abstracción incluye parámetros de red como el retardo, la tasa, la disponibilidad,…dando a las aplicaciones una visión más amplia de la red. Las aplicaciones comunicarán al controlador, estableciendo los requerimientos para el establecimiento de las conexiones, y será el

controlador el que configurará los elementos individuales de la red en el plano de datos para lograr un encaminamiento del tráfico eficiente.

El manejo centralizado de los elementos de la red proporciona ventajas adicionales a los administradores dándoles estadísticas vitales de las condiciones existentes de la red con el fin de adaptar la calidad de servicio y customizar la topología de la red según sea necesario. Por ejemplo, durante periodos de de alta utilización de la red, cierto ancho de banda que consume servicios de vídeo, streaming, grandes transferencias de datos, etc, pueden ser balanceados utilizando canales dedicados. En otras situaciones, como por ejemplo una emergencia (alarmas de incendios, evacuaciones de edificios,…), servicios como VoIP pueden tomar preferencia frente a otros servicios.

4.2.5. Seguridad en SDN.

El creciente interés en SDN por parte de la comunidad ha iniciado un debate significativo en cuanto a los retos de seguridad inherentes a la infraestructura de SDN.

El control centralizado hace que el esquema sea vulnerable a ataques directos al plano de control lo cual puede alterar toda la red. La inteligencia existente en este plano de control centralizado ofrece a los hackers la oportunidad de buscar vulnerabilidades de seguridad en el controlador y tomar el control de toda la red. Por otro lado, la información generada a partir del análisis del tráfico o por una detección anómala en la red puede ser regularmente transmitida al controlador SDN, teniendo así una visión global de la red y una mayor eficiencia en cuanto a seguridad.

El canal de control entre el controlador y los dispositivos de la red tiene que ser lo suficientemente seguro para rechazar comportamientos anómalos, al igual que para la comunicación entre controlador y aplicación. Establecer efectivamente confianza entre los dispositivos de la red y las aplicaciones es considerada una preocupación clave. Análisis de vulnerabilidad, estudios de mitigación y una infraestructura de seguridad estandarizada ha sido el foco de múltiples deliberaciones, centradas sobretodo en la seguridad del canal de comunicación entre el controlador y el *switch* y en las comunicaciones intercontrolador. Seungwon Shin y Guofei Gu, por ejemplo, evalúan la viabilidad de realizar ataques de huella (fingerprint). Esta evaluación ataca los equipos SDN como los *switches* y enfoca a los demás elementos con ataques de denegación de

servicio en el controlador a través del canal de control y los elementos del plano de datos explotando las tablas de flujos [11]. Ambas entidades son significativamente vulnerables. De manera similar, Ruslan L. Smeliansky trató el protocolo de seguridad en consideración a la infraestructura y a los servicios de *software*, concluyendo que los planos control-datos y control-control necesitan un endurecimiento sustancial con el fin de mitigar las amenazas de seguridad [12]. Algunas de las soluciones apuntan a replicar los controladores SDN y las aplicaciones de red para proporcionar redundancia y una realización de operaciones protegidas. Otras investigaciones promueven una movilidad de las entidades o servicios. Las funcionalidades del controlador, por ejemplo, pueden ser continuamente desplazadas a través de varios elementos de la red haciendo que los ataques que apunten directamente al controlador sean más costosos para aquellos que busquen una vulnerabilidad en el plano de control. Lisa Schehlmann discute las mejoras potenciales en los costes en cuanto al manejo de la red, así como en la detección de ataques y mitigación usando la infraestructura SDN por sí sola como una barrera potencial en cuanto a las vulnerabilidades de seguridad [13]. SDN habilita la incorporación de ciertas funcionalidades de seguridad a través del desacoplo de la red de control y la lógica de encaminamiento, donde el tráfico puede ser filtrado usando identificadores de paquetes a través de firewalls dedicados y sistemas de detección y prevención de intrusiones. Adicionalmente, se pueden añadir capas de seguridad por encima de capas SDN ya existentes, así como la introducción de agentes en el plano de datos con el fin de introducir más granularidades para filtrar el tráfico específicamente en redes heterogéneas.

Además de las aplicaciones específicas, la monitorización a tiempo real tiene que ser lo suficientemente robusta como para ofrecer detección de eventos anómalos en la red. La información monitorizada no solo proporciona una visión interna del tráfico, sino que también satisface requerimientos técnicos y legales.

Las aproximaciones de seguridad se enfocan, por tanto, en asegurar la red por sí misma a partir de alarmas de seguridad en los elementos de la red como *switches* y controladores o incluyendo utilidades de seguridad orientadas a SDN en las entidades funcionales como los servidores de aplicaciones y los clusters de almacenamiento.

Organizaciones como *OpenFlowSec*, se centran particularmente en los desafíos en cuanto a seguridad expuestos por SDN y los dispositivos *OpenFlow*. Los

desarrolladores han considerado del mismo modo la implementación de una referencia o fuente en cuanto a características de seguridad en diferentes capas de la pila *OpenFlow*. Más allá de la arquitectura básica de SDN, el despliegue de una seguridad robusta es todavía un área que requiere gran estudio. Se cree que, además, sin un incremento significativo en cuanto a la seguridad en SDN se refiere, el paradigma solo conseguirá adaptarse en infraestructuras privadas y en despliegues autónomos de organizaciones.

4.2.6. *OpenFlow*.

El protocolo *OpenFlow*, mantenido y actualizado por ONF es el primer y más destacado interfaz de comunicación SDN. Usando *OpenFlow*, las reglas de encaminamiento pueden ser añadidas o eliminadas de las tablas de flujo de los *switches* para hacer la estructura de la red más sensible a las demandas de servicio. *OpenFlow* fue desarrollado en las primeras etapas de SDN y cuyo fin es intercambiar mensajes de control entre el controlador SDN y los componentes de la red pertenecientes al plano de datos.

4.2.6.1. ¿Por qué se desarrolló *OpenFlow*?

Los principales motivos por los cuales se desarrolló *OpenFlow* pasan por una innovación de las redes de comunicaciones, además de ofrecer una alternativa a los protocolos existentes, mediante un modelo computacional basado en *software*. Antes del desarrollo de las redes definidas por *software*, éstas eran cada vez más rápidas, pero no necesariamente mejores. Además, la velocidad de cambio de las redes resulta pequeña en comparación con las innovaciones desplegadas en otras áreas (sistemas operativos, sistemas distribuidos, compiladores,…) y son, en gran parte, las mismas que en años anteriores (Ethernet, IP, WiFi,…).

Las innovaciones de las redes de comunicaciones proporcionadas por *OpenFlow* se basan en una disponibilidad total, así como en eliminar las complicadas pruebas de rendimiento de las redes. Además, podemos dividir la red en una parte de producción y en otra parte experimental.

En cuanto a los costes operacionales, *OpenFlow* hace innecesaria la adición de *routers* o *switches* a la red, lo cual hacen que estos costes disminuyan. Además, cada dispositivo añadido a la red no debe ser configurado individualmente. En menor medida, las actualizaciones de firmware y los cambios en general, sí pueden afectar al OPEX. Por otro lado, las tareas operacionales son más escalables y es más sencillo añadir

servidores y configurar políticas en la red. La configuración mediante el interfaz de línea de comandos pasa a ser un controlador *OpenFlow* centralizado.

Actualmente, el modo de añadir o crear nuevas funcionalidades se basa en la definición de un protocolo nuevo. Además, los estándares implementados por los vendedores, crecen exponencialmente, siendo cada vez más complejos, lo cual aumenta el coste y disminuye la estabilidad. ¿De verdad son necesarios estos estándares?

4.2.6.2. ¿Cómo funciona *OpenFlow*?

Como ya se ha mencionado, *OpenFlow* ofrece una serie de ventajas que facilitan la implementación y desarrollo de redes de comunicaciones, pero, ¿cómo funciona?

Por un lado, el plano de control se mueve fuera del *switch*, mediante el empleo de un controlador. Este controlador se conecta a uno o varios *switches* lo que crea una visión más centralizada de toda la red.

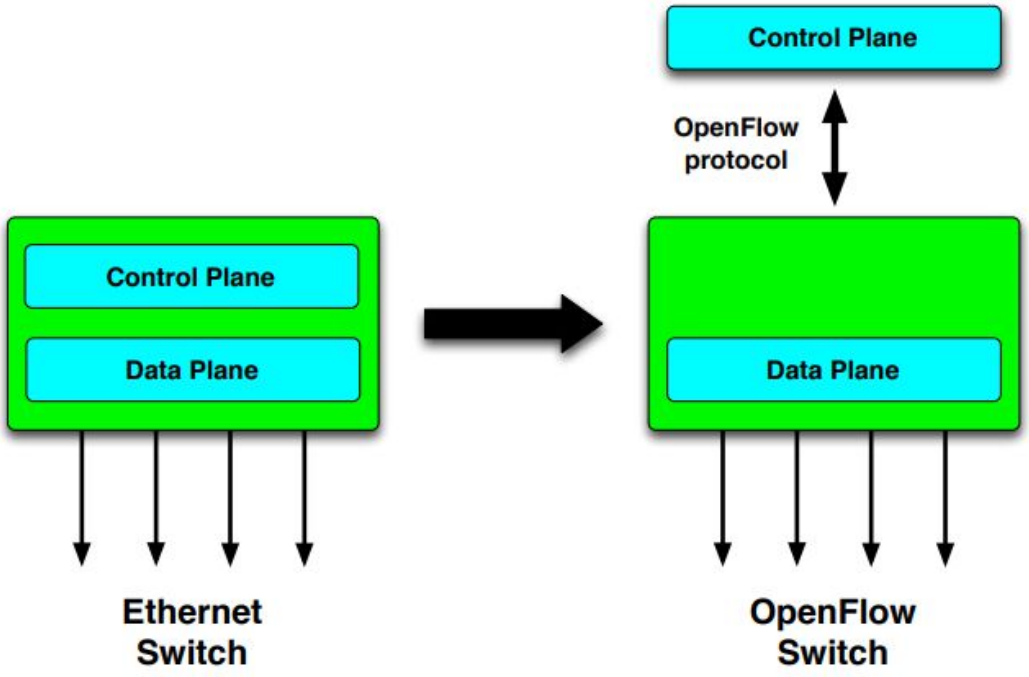


Fig. 4.6. Separación de los planos de datos y control mediante el protocolo *OpenFlow* [14].

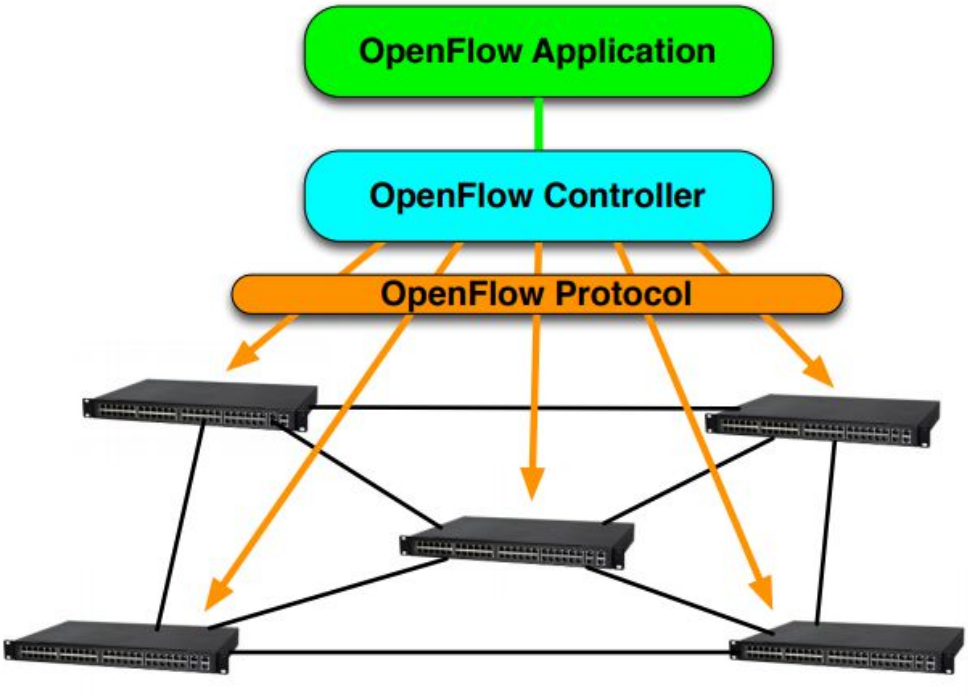


Fig. 4.7 Conexión del controlador *OpenFlow* con diversos *switches* [14].

Mediante este controlador, se podrá introducir una serie de reglas en cada *switch*, con el objetivo de modelar el tráfico según convenga. Una vez establecida la conexión entre el *switch* y el controlador, se podrán intercambiar intercambiar paquetes con este, añadir entradas en su tabla de flujos, solicitarle datos estadísticos del tráfico, ver sus tablas de flujos o sus propias características y parámetros, etc.

El componente esencial de *OpenFlow* son las tablas de flujos. Cada flujo estará formado por los campos de la cabecera, mediante los cuales se emparejarán paquetes (*match*), contadores, que llevarán las cuenta de los paquetes emparejados y acciones, que serán aquellas llevadas a cabo una vez se produzca un emparejamiento.

Como se ha mencionado, el emparejamiento se basa en los campos de la cabecera de cada paquete. *OpenFlow* ofrece un gran número de campos a utilizar. Entre ellos se pueden destacar: puerto de entrada, direcciones Ethernet de origen y destino, tipo de Ethernet, identidad o prioridad VLAN, direcciones origen y destino IPv4, prioridad Ipv4, tipo de servicio Ipv4, puertos de origen y destino TCP o UDP, tipo/código ICMP,…También se podrán utilizar las direcciones origen y destino Ipv6 en versiones de *OpenFlow* 1.2 o superiores.

En cuanto a los contadores, hay varios tipos: por tabla, por flujo, por puerto, por cola, por grupo, por conjunto de grupos,… Los más utilizados serán los contadores por tabla, flujo y puerto. Con los contadores por tabla, se podrá llevar la cuenta de las entradas activas, las búsquedas llevadas a cabo en la tabla para cada paquete y el número de emparejamientos para cada paquete. Mediante los contadores por flujo, se controlará el número de paquetes y *bytes* recibidos y la duración del flujo en la tabla, en segundos y nanosegundos y finalmente, con los contadores por puertose verá el número de paquetes, bytes, descartes y errores (alineamiento de trama, desbordamiento y CRC) transmitidos y recibidos, así como el número de colisiones.

Por último, las acciones se llevarán a cabo una vez que se produzca un emparejamiento de un paquete con una entrada de la tabla de flujos. Estas pueden ser de encaminamiento, encolamiento, descarte o de modificación de algún campo. La acción más utilizada es la de encaminamiento, mediante la cual, una vez recibido un paquete por un puerto de entrada, se encaminará por un puerto de salida. Este puerto de salida puede ser el del controlador, el puerto local del *switch*, el puerto de entrada (reenvío del paquete), o enviarlo por todos los puertos estándar, sin incluir el puerto de entrada. En el caso de querer seguir procesando el paquete de entrada, se podrá enviar a otra tabla de flujos, y no necesariamente a un puerto de salida. También, opcionalmente, se podrá enviar, sin usar el canal *OpenFlow*, por un puerto o por todos (flood). En caso de querer modificar un campo del paquete, antes de realizar alguna otra acción, se podrá modificar o añadir la identidad VLAN, eliminar la cabecera VLAN, modificar las direcciones de origen y destino Ethernet, Ipv4, Ipv6, así como los puertos origen y destino TCP y UDP,…

Las entradas en las tablas de flujos pueden ser proactivas o reactivas. Las proactivas son aquellas insertadas antes de la llegada de los paquetes. Las reactivas son aquellas insertadas después de la llegada de los paquetes. Por defecto, si un paquete no es emparejado en la tabla de flujos, es enviado al controlador, el cual analizará los campos del paquete para añadir una entrada en la tabla para el tráfico con las mismas características que el paquete analizado [14].

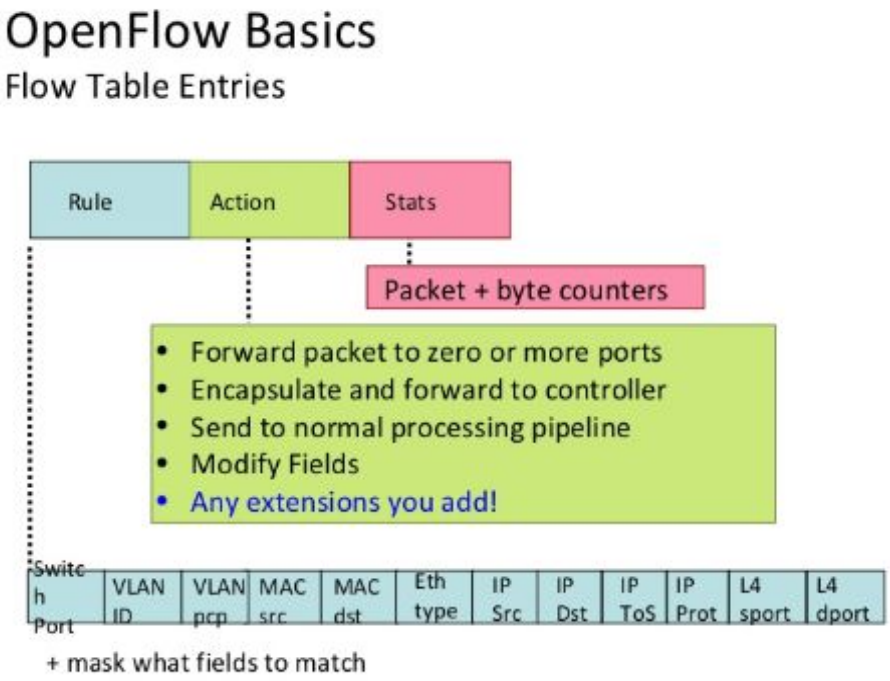


Fig. 4.8 Funcionamiento de las tablas de flujo de *OpenFlow* [14].

Sin embargo, una vez visto el funcionamiento genérico de *OpenFlow*, es fácil observar que no es suficiente. Añade la capacidad de modificar, experimentar,…pero resulta difícil añadir funcionalidades a la red. *OpenFlow* es solo un protocolo de manejo de tablas de encaminamiento.

4.2.6.3. Estandarización de *OpenFlow* (ONF).

El *OpenFlow Switch*ing Consortium fue creado en 2008 con el objetivo de popularizar el protocolo *OpenFlow* y mantener la especificación del mismo (*OpenFlow Switch* Specification). Cualquier discusión acerca de la estandarización de *OpenFlow* se lleva a cabo en este consorcio.

En Marzo de 2011, Deutsche Telecom, Facebook, Google, Microsoft, Verizon y Yahoo crearon la Open Networking Foundation (ONF) con el fin de promocionar las redes definidas por *software* (SDN). Cualquier actividad relacionada con el proceso de estandarización de las especificaciones de SDN y *OpenFlow* se lleva a cabo en la ONF.

La ONF ha creado grupos de trabajo que conducen la estandarización técnica de SDN/*OpenFlow*, manteniendo charlas técnicas, realizando estudios de compatibilidad, o preparando borradores que, de ser aprobados, pasan a formar parte de la especificación estandarizada [15].

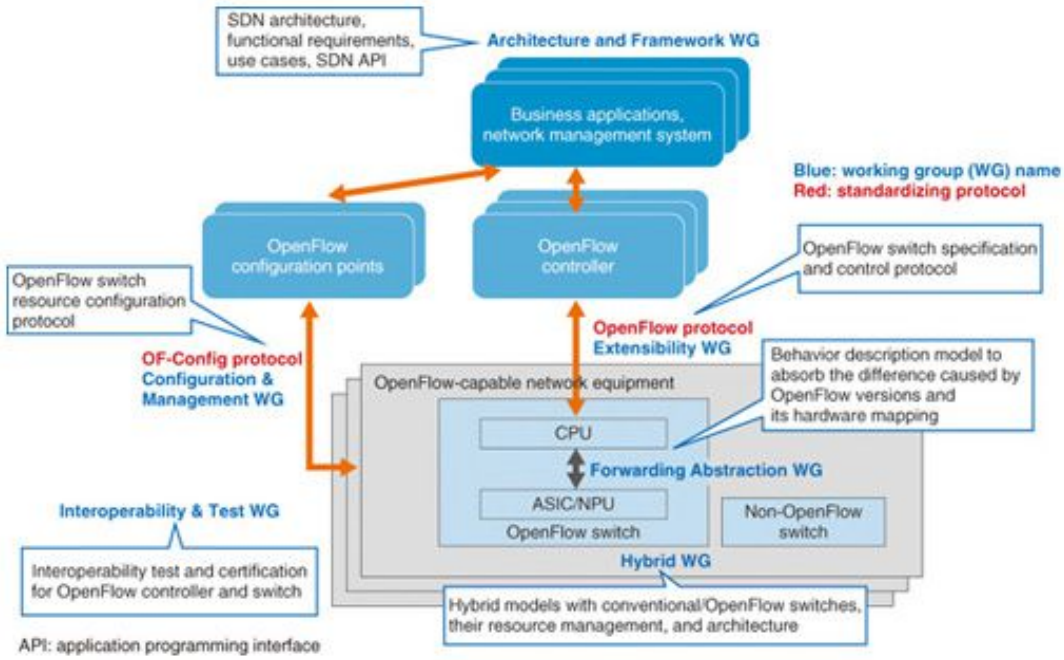


Fig. 4.9. Grupos de trabajo y áreas de estandarización de la ONF [15].

4.2.6.4. Ejemplos del uso de *OpenFlow*.

- Red de datos de Google.

Google introdujo *OpenFlow* en su red *G-Scale* en 2010, por la cual circula tráfico de su centro de datos (intra e inter), con menores acuerdos de nivel de servicio (SLAs), por lo que resulta perfecta para probar *OpenFlow*. Emularon esta red WAN en un simulador, utilizando *switches* con 128 puertos de 10 GE(10 Gb/s) con soporte *OpenFlow* y varios controladores.

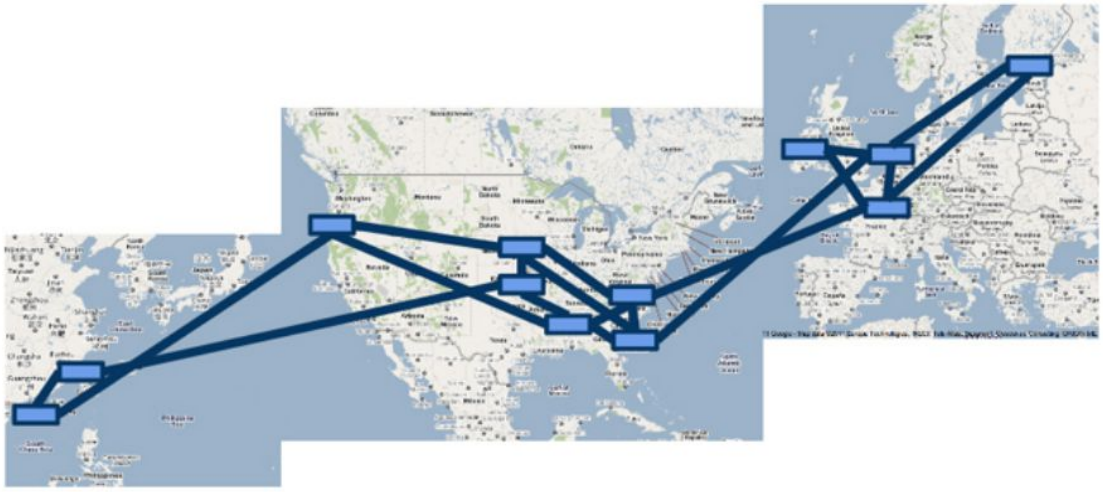


Fig 4.10. Google *G-Scale OpenFlow* WAN (Red de datos) [14].

Como conclusiones, Google vio que el desarrollo de *software* para un servidor de alto rendimiento con herramientas modernas (debuggers, etc) es mucho más fácil y sencillo y produce *software* de mayor calidad que el desarrollo de un sistema embebido (router/*switch*) con un CPD lento y con poca memoria. Además, experimentaron un uso de casi el 100% de los enlaces.

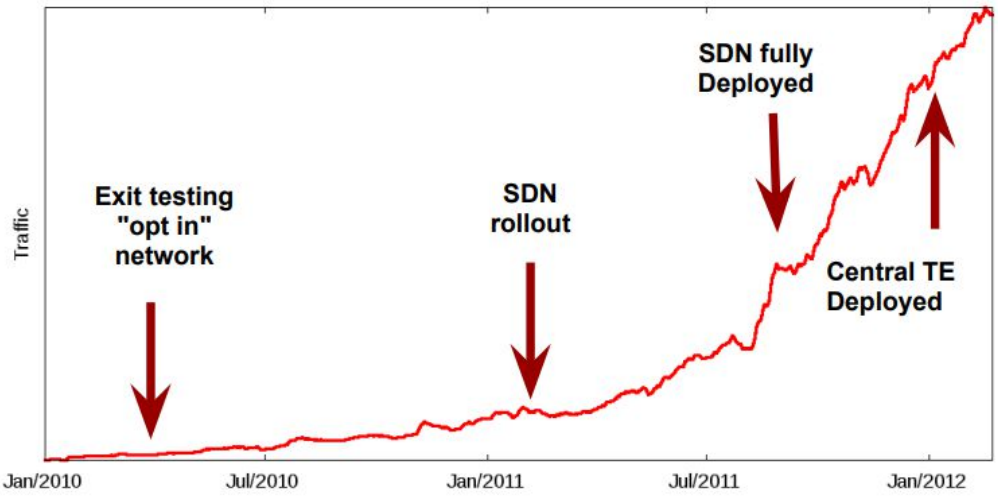


Fig 4.11. Uso de la red *G-Scale* de Google [14].

- Plug-n-Serve.

El objetivo de Plug-n-Serve es balancear la carga en los enlaces de una red desestructurada. *OpenFlow* nos ofrece un control completo del tráfico dentro de la red, la visibilidad de las condiciones de la misma y la habilidad de utilizar hardware básico, entre otras, por lo que se utilizarán estas características para realizar esta tarea.

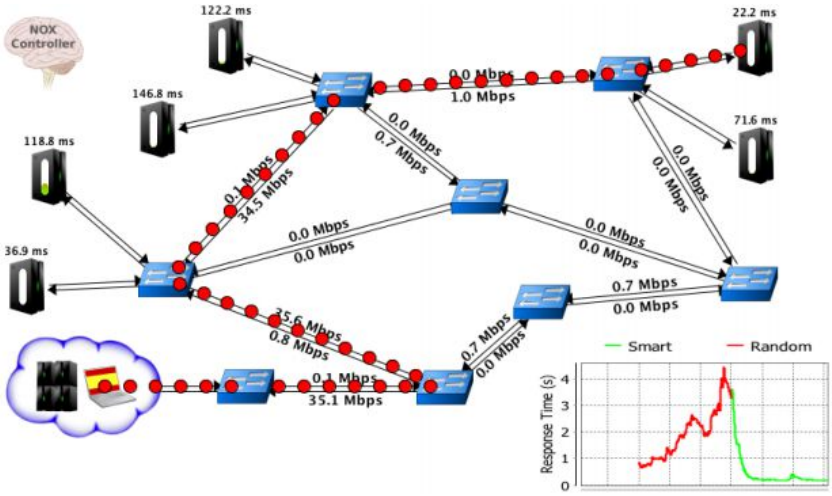


Fig. 4.12. Simulación de *Plug-n-Serve* [9].

- Elastic Tree.

El principal objetivo de *Elastic Tree* es reducir la potencia en las redes pertenecientes a centros de datos. Mediante la flexibilidad que ofrece *OpenFlow*, se deshabilitarán enlaces y *switches* y se elegirán rutas óptimas, eligiendo optimizadores para el balanceo de potencia, la tolerancia a errores y el ancho de banda, y utilizando las estadísticas de las rutas y puertos.

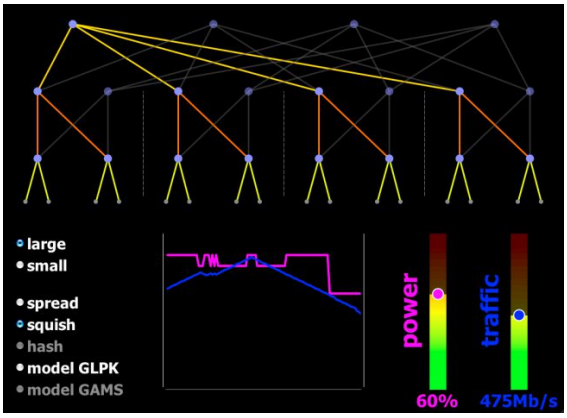


Fig. 4.13. Simulación de *Elastic Tree* [9].

4.2.7. Marco regulador.

Por regla general, la transición hacia redes SDN supone un gran esfuerzo para la mayor parte de los proveedores.

El cambio de redes distribuidas a redes centralizadas puede suponer, incluso, un cambio orientado a un modelo de negocio basado en SDN. Supone trasladar todo el control por hardware distribuido a un nodo centralizado de control por *software*.

4.2.7.1. Estándares técnicos.

Como ya se ha explicado, en este proyecto se utilizará el estándar *OpenFlow* como protocolo para implementar SDN. También se dispondrá de un controlador mediante el cual se realizarán todas las labores de control de la red. Este controlador mediante lenguaje Python, más concretamente utilizando la librería *OpenFlow* del controlador Ryu, que será el controlador utilizado en este proyecto [16], aunque en algunos casos se usará la API general de Python.

4.2.7.2. Legislación y regulación.

Existen proyectos en el IEEE (*Institute of Electrical and Electronics Engineers*) directamente relacionados con SDN [17]:

- IEEE P1903.1 – Standard for Content Delivery Protocols of Next Generation Service Overlay Network (NGSON).
- IEEE P1913.1 – *Software*-Defined Quantum Communication.
- IEEE P1915.1 – Security for Virtualized Environments.
- IEEE P1916.1 – Performance for Virtualized Environments.
- IEEE P1917.1 – Reliability for Virtualized Environments.
- IEEE P1921.1 – *Software*-Defined Networking Bootstrapping Procedures.
- IEEE P1930.1 – SDN based Middleware for Control and Management of Networks.
- IEEE P802.1CF – Recommended Practice for Network Reference Model and Functional Description of IEEE 802 Access Network.

Otro punto a abordar es la existencia del debate de si los servicios deberían ser iguales o no para todos los usuarios, independientemente de la tarifa a pagar por cada usuario.

Gran parte de los usuarios defienden este concepto, conocido como la *Neutralidad de red*, que exige una regulación mediante la cual los proveedores traten el tráfico de los usuarios de forma indiscriminada, sin tarifas extra dependiendo del contenido o servicio al que accedan.

Sin embargo, la existencia de una regulación podría frenar el avance de SDN, ya que el protocolo se basa en una personalización casi total de las redes, concepto que muchos fabricantes no defienden.

No obstante, la implementación de SDN se encontraría bajo cierta regulación y legislación ya existente en las redes tradicionales:

- En primer lugar hay que tener en cuenta la IETF (Internet Engineering Task Force), que se encarga de desarrollar los protocolos de internet [18]. La IETF no es una organización ni un organismo gubernamental, sino un grupo abierto. Además del desarrollo de protocolos, también se encargan del desarrollo de la red y de avanzar tecnológicamente respecto a su funcionamiento y usabilidad. Respecto a SDN, la publicación más destacada del IETF es el RFC 7426 [19].
- La Ley de Protección de Datos de Carácter Personal se mantiene igualmente vigente independientemente de cuanto cambien o se modernicen las redes de comunicaciones [20], así como el nuevo Reglamento Europeo de Protección de Datos vigente desde el 25 de Mayo de 2018, que regula más en detalle aspectos de la evolución tecnológica en cuanto a la privacidad y a la protección de datos [21].
- Al igual que en el caso anterior, la Ley General de Telecomunicaciones se mantiene de igual vigente [22] (conservación de datos, explotación de redes y prestación de servicios, integridad y seguridad de las redes, etc).

Una de las preocupaciones del despliegue de redes definidas por *software* es el cumplimiento de la regulación y legislación establecida. Además, puede suponer un reto extra si la compañía opera en varios sectores o países con diferentes normas reguladoras. SDN no resuelve las barreras existentes en torno al cumplimiento de las

mismas. Sin embargo, facilita su cumplimiento. Existen ciertas observaciones a tener en cuenta respecto a la regulación y legislación existente:

- **Pequeñas empresas bajo las regulaciones de SDN**: La implementación de nuevas tecnologías sometidas a cambios de políticas constantes puede suponer un alto coste para pequeñas empresas o *startups*. Las grandes compañías, por el contrario, no sufren tanto a la hora de cumplir con la regulación, ya que esta hace que las pequeñas empresas tomen otros caminos distintos a tecnologías con políticas cambiantes, aunque preferirían disponer de otras opciones para cumplirla.

- **Reducción de costes**: Como ya se ha explicado, SDN implica una reducción de costes en cuanto a la sustitución de equipamiento, su reparación, su control, etc. Todo ello se debe a la centralización de las funciones de control y a la posible virtualización de la red. Además, se puede utilizar este control por *software* para facilitar el cumplimiento de la regulación a partir de la definición de normas específicas. Trabajar con esta flexibilidad hace que sea más fácil realizar cambios.

- **Manutención del cumplimiento a través de varios sectores:** Las normas reguladoras son diferentes para el sector sanitario que para el sector financiero, por ejemplo, por lo que el cumplimiento de la regulación se hace más complejo. Se podría tener una red dedicada para cada sector, lo que supondría altos costes en la compra de equipamiento y *software* específico para las normas de cada sector.

  Sin embargo, con SDN, se pueden separar los recursos de la red para cada sector individual de tal forma que en cada uno de ellos se cumpla la regulación existente, independientemente del hardware existente en la red [23]:

- **Asegurar la seguridad en la red:** En lugar de realizar actualizaciones de seguridad de forma individual en servidores y redes, se pueden realizar desde una única localización. Es más sencillo y requiere menor tiempo cumplir con las normas regulatorias de seguridad.

- **Manejar el cumplimiento global:** Al igual que en el cumplimiento por sectores, mantener el cumplimiento en múltiples países es una tarea compleja.

Con SDN, no importa dónde están localizados los servidores ya que siempre estarán conectados a una única red. En el caso de querer realizar cambios en el hardware físico, el centro de datos puede realizarlos y posteriormente actualizar las políticas pertinentes para el cumplimiento de la regulación tras esos cambios.

4.2.8. Restricciones.

Las principales restricciones existentes relativas al despliegue de redes definidas por *software* son:

- La necesidad de investigación y mejora. SDN es relativamente joven y aún no está implementado en la mayor parte de las redes de comunicaciones.
- Gran número de usuarios. El requerimiento de un servicio de calidad por parte de los usuarios hace que las nuevas tecnologías deban ser investigadas de forma intensiva antes de su implementación.
- La adaptación de las redes tradicionales hacia esta solución puede suponer altos costes, así como la necesidad de formar e incluso incorporar trabajadores.

Además, la aparición de SDN ha alertado a los fabricantes, los cuales se han visto en la necesidad de crear alternativas a SDN. Es el caso de Cisco, que lanzó en Octubre de 2014 el protocolo OpFlex [24][25], el cual ofrece unos beneficios similares a SDN. Sin embargo, no centraliza las funciones de red en un controlador SDN, sino que se basa en las políticas de red para realizar las funciones de control, continuando con el esquema de control distribuido de las redes tradicionales.

Probablemente esta sea una de las mayores barreras de SDN, ya que los fabricantes seguirán ofreciendo alternativas a la solución a fin de mantener su posición en el mercado.

5. DESPLIEGUE DE LA SOLUCIÓN.

5.1. Introducción.

El despliegue completo de la red se dividirá en dos. En primer lugar, se realizará un despliegue basado en máquinas virtuales, que generarán tráfico que se encaminará a través de los *switches* virtuales, utilizando el controlador. Estas máquinas virtuales se instalarán en el mismo equipo que los *Open vSwitch* (una máquina por cada uno). El objetivo de este despliegue será adquirir una toma de contacto con el protocolo *OpenFlow* y el controlador, así como con las posibilidades que estos ofrecen.

Una vez adquirida esta toma de contacto, se realizará el despliegue final. En este despliegue se utilizarán dos equipos, en este caso ordenadores portátiles, que actuarán como equipos en los extremos, conectados a los *switches* virtuales. A través de estos equipos, se darán diversos servicios, en concreto HTTP (Protocolo de Transferencia de Hipertexto) en los puertos 80 y 8080, FTP (Protocolo de transferencia de archivos) y TFTP (Protocolo de transferencia de archivos trivial).

En ambos despliegues, se manejarán tres VLANs, simulando las tres VPNs que de la red de ADIF (tráfico corporativo, tráfico de Renfe y tráfico de seguridad).

El despliegue se realizará en forma de maqueta virtualizada, por lo que todos los equipos necesarios se simularán o virtualizarán en PCs.

5.2. Esquema de red.

El esquema general de la red a simular es el siguiente:

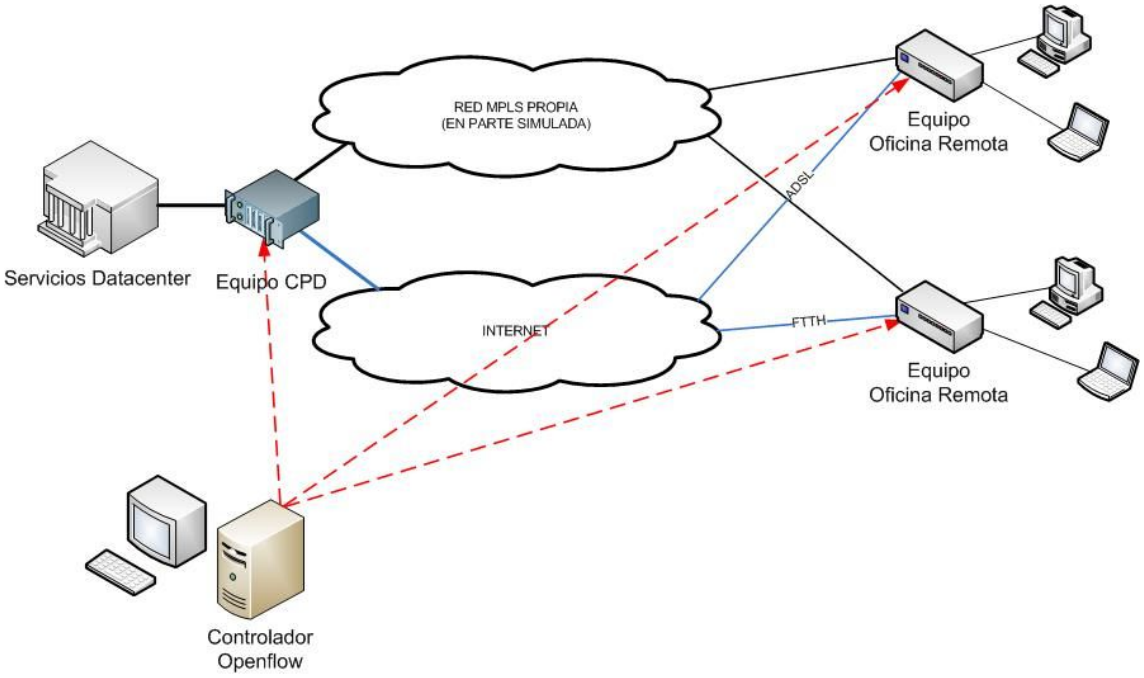


Fig. 5.1. Esquema de red general.

Es necesario tener en cuenta que al encaminar tráfico en los *switches* virtuales, no se tiene en cuenta la red MPLS dado que las funciones de etiquetado o desetiquetado (*push/pop)* las lleva a cabo la red independientemente. Por tanto, simplemente se tendrán en cuenta que tres puertos del *Open vSwitch* (uno por etiqueta VLAN) recaban en la red MPLS y uno por internet por lo que la conexión entre los *switches* virtuales será directa en la maqueta.

Además, por simplicidad, y dado que la maqueta será un entorno totalmente aislado de la red, no se utilizará el plan de numeración de etiquetado VLAN ni de direccionamiento privado aplicado en la red. Se contará con uno propio establecido *ad-hoc*.

5.3. *Software* a utilizar.

Para la virtualización de la red SD-WAN se utilizarán 3 PCs, dos de ellos con sistema operativo *CentOS* 7 para el primer despliegue y Linux *Ubuntu* 16.04 para el segundo despliegue y el restante con sistema operativo *CentOS* 7. Dos de ellos actuarán como *switches* virtuales (*CentOS* y *Ubuntu*), mediante el uso de la herramienta de simulación *Open vSwitch*, y uno de ellos actuará como controlador (*CentOS*), mediante el cual se generarán y modificarán flujos de tráfico en base a unos objetivos. El controlador a utilizar será el *Ryu Controller*, versión 4.22, el cual utiliza lenguaje *Python*. Además, se

instalará *Minicom* 2.6.2 para configurar un *switch* Catalyst 2960 de Cisco mediante el cual se conectarán físicamente los 3 equipos.

5.3.1. Controlador *Ryu*.

Como ya se ha comentado, el controlador SDN actúa como el cerebro de la red, ya que se encargará de las labores de encaminamiento y monitorización de la red con el fin de conseguir con control de la red a tiempo real y una mayor eficiencia en su uso. El controlador que se utilizará en esta solución es el controlador *Ryu*, el cual emplea una estructura de lenguaje en *Python*, basada en su propia librería [26].

Para la instalación del controlador *Ryu* en el equipo con *CentOS* 7, se ejecutarán los siguientes comandos:

```
yum –y install python-pip

pip install ryu
```

5.3.2. *Open vSwitch*.

Se trata de un *switch* virtual multicapa, el cual se utilizará para realizar el encaminamiento del tráfico extremo a extremo. Se emplearán dos de ellos, uno en cada extremo, actuando como puerta de enlace. Para su descarga, se ejecutarán los siguientes comandos en *CentOS* 7:

```
yum –y install wget openssl-devel gcc make python-devel openssl-devel kernel-devel graphviz kernel-debug-devel autoconf automake rpm-build redhat-rpm-config libtool python-twisted-core python-zope-interface PyQt4 desktop-file-utils libcap-ng-devel groff checkpolicy selinux-policy-devel

adduser ovs

su – ovs

mkdir –p ~/rpmbuild/SOURCES

wget http://openvswitch.org/releases/openvswitch-2.8.2.tar.gz

cp openvswitch-2.8.2.tar.gz ~/rpmbuild/SOURCES/
```

```
tar xfz openvswitch-2.8.2.tar.gz

rpmbuild –bb –nocheck openvswitch-2.8.2/rhel/openvswitch-fedora.spec

exit

yum localinstall /home/ovs/rpmbuild/RPMS/x86_64/openvswitch-2.8.2-1.el7.centos.x86_64.rpm –y

systemctl start openvswitch.service

systemctl is-active openvswitch
```

En el primer despliegue se empleará el protocolo *OpenFlow* versión 1.5, mientras que en el segundo despliegue se empleará la versión 1.3. El motivo puede consultarse en el Anexo A.

5.3.3. *PyCharm* IDE.

Con el objetivo de desarrollar el código correspondiente para cada despliegue, se utilizará el entorno de desarrollo *PyCharm*. Para descargarlo, simplemente se accederá al instalador de aplicaciones de *CentOS* y se buscará en el buscador para proceder a su descarga.

Dentro del programa, se podrán realizar *scripts* con extensión ".py". Además, es conveniente crear un fichero "debug.py" con el fin de poder depurar el código en caso de que el funcionamiento no sea el adecuado. La estructura de este fichero es la siguiente:

```
import sys

from ryu.cmd import manager

def main():

    sys.argv.append(--ofp-tcp-listen-port')

    sys.argv.append('6633')

    sys.argv.append('nombre del script 1')

    sys.argv.append('nombre del script N')
```

```
        sys.argv.append(--verbose')
        sys.argv.append(--enable-debugger')
        manager.main()
if __name__ == '__main__':
    main()
```

Con este *script*, se podrán ejecutar como depurar tantos *scripts* como se desee.

5.4. Despliegue con máquinas virtuales.

5.4.1. Introducción.

Como toma de contacto con el protocolo *OpenFlow,* se realizará un despliegue inicial de la red con máquinas virtuales como extremos, con todos los equipos ejecutando *CentOS* 7 como sistema operativo. Este despliegue sigue el siguiente esquema:

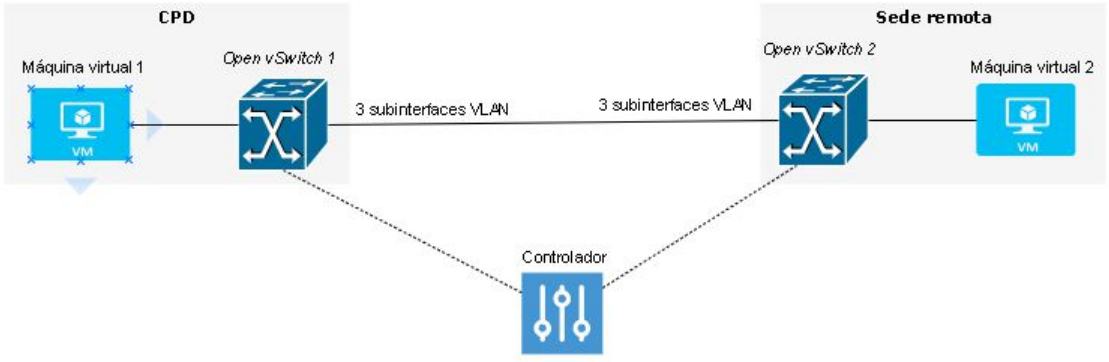


Fig. 5.2. Esquema de red del despliegue con máquinas virtuales.

El objetivo será crear dos máquinas virtuales en cada equipo con *switch* virtual que actúen como extremos de la red y generen tráfico entre ellas, el cual será encaminado en los *switches* virtuales utilizando el controlador. Cada equipo dispondrá de una sola tarjeta de red para este despliegue, dado que en este caso, los extremos de la red están virtualizados.

5.4.2. Instalación de las máquinas virtuales.

Para la instalación de las máquinas virtuales, primero se deberá instalar el *software* necesario para realizar esta tarea. Para ello, se ejecutarán el siguiente comando:

```
yum install qemu-kvm qemu-img virt-manager libvirt libvirt-python libvirt-client virt-i
```

Una vez instalado el *software* necesario, se procede a la instalación de la máquina virtual. Para ello, se alocará la memoria deseada para la máquina mediante el comando *qemu-img* y acto seguido se instalará la máquina virtual. En este caso, se han asignado 5 Gigabytes de memoria a una máquina virtual que ejecutará *CentOS 7 Minimal* [27], sin interfaz gráfica, dado que no será necesaria. Cualquier otro sistema operativo puede ser empleado:

```
qemu-img create –f qcow2 /home/user/VM.qcow2 5G

virt-install –connect qemu:///system –name VM –ram 1024 –vcpus 1 –disk /home/user/VM.qcow2 –cdrom /home/user/CentOS-7-x86_64-Minimal-1708.iso– accelerate –keymap=es
```

En cuanto a las opciones de instalación de la máquina virtual, el usuario decidirá los valores de RAM, CPUs, etc a utilizar. Es muy importante tener en cuenta las características del equipo en el que se trabajará antes de instalar las máquinas virtuales.

Una vez instaladas en ambos equipos, se iniciarán. Para ello, se ejecutará el siguiente comando:

```
virt-manager
```

Acto seguido, se abrirá una ventana en la cual aparecerá la máquina virtual con el nombre que se le haya asignado al instalarse. Haciendo doble *click* en la misma, se abrirá la ventana con el interfaz gráfico de la máquina virtual.

5.4.3. Configuración de las máquinas virtuales.

El objetivo de este despliegue es encaminar tráfico etiquetado entre las máquinas virtuales a través de los *switches* virtuales. Para ello, en cada máquina virtual CentOS se crearán 3 subinterfaces de tipo VLAN, que generarán tráfico etiquetado. Se utilizarán las VLAN 10,11 y 12, por ejemplo:

TABLA. 5.1. SUBINTERFACES DE TIPO VLAN DE LAS MÁQUINAS VIRTUALES, ASÍ COMO SU DIRECCIONAMIENTO					
Máquina virtual extremo CPD			**Máquina virtual extremo sede remota**		
Interfaz	Dirección IP	VLAN id	Interfaz	Dirección IP	VLAN id
eth0.10	172.16.10.1/24	10	eth0.10	172.16.10.2/24	10
eth0.11	172.16.11.1/24	11	eth0.11	172.16.11.2/24	11
eth0.12	172.16.12.1/24	12	eth0.12	172.16.12.2/24	12

Para lograr esta configuración, se crearán los ficheros */etc/sysconfig/network-scripts/ifcfg-eth0.10*, */etc/sysconfig/network-scripts/ifcfg-eth0.11* y */etc/sysconfig/network-scripts/ifcfg-eth0.12* de la siguiente forma:

```
DEVICE=eth0.10
ONBOOT=yes
BOOTPROTO=static
IPADDR=172.16.10.1
PREFIX=24
NETWORK=172.16.10.0
VLAN=yes
```

Fig. 5.3. Configuración del fichero de un subinterfaz de tipo VLAN.

La configuración de los demás ficheros es idéntica, cambiando el nombre del interfaz por eth0.11 y eth0.12 y el direccionamiento IP según la tabla 5.1.

Además, esta máquina virtual, al instalarse, estará configurada por defecto para enlazarse al equipo físico en el que está instalada a través de un interfaz tipo *tap*. Sin embargo, se configurará para que al iniciarse la máquina virtual, se enlace con el *switch* virtual directamente, añadiendo este interfaz *tap* como puerto del *switch* virtual. Para ello, se ejecutará el siguiente comando para acceder al fichero de configuración de la máquina virtual:

```
virsh edit “nombre de la máquina virtual”
```

Dentro del fichero de configuración se buscará el controlador de red de la máquina y se cambiarán sus parámetros de la siguiente forma:

```
</controller>
<interface type='bridge'>
  <mac address='52:54:00:66:06:8f'/>
  <source bridge='cpd'/>
  <virtualport type='openvswitch'>
    <parameters interfaceid='c745389b-9f82-497e-ab0d-1fde67b48bdc'/>
  </virtualport>
  <model type='virtio'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
</interface>
```

Fig. 5.4. Configuración del controlador de red de la máquina virtual para conectarla al *switch* virtual.

De esta forma, cada vez que se arranque la máquina virtual, se conectará directamente al *switch* virtual.

5.4.4. Configuración de los *switches* virtuales.

Con el objetivo de encaminar el tráfico etiquetado a través del *switch* virtual, se deberán crear otros tres subinterfaces de tipo VLAN en los equipos físicos, que posteriormente se añadirán al *switch* virtual como puertos.

Hay dos formas de crear un *Open vSwitch* y añadir interfaces como puertos del mismo. La primera es configurando los subinterfaces de tipo VLAN de la misma forma que en la configuración de las máquinas virtuales y añadiendo estos subinterfaces al *switch* virtual mediante el comando *ovs-vsctl*, una vez creado:

```
ovs-vsctl add-bridge cpd
ovs-vsctl add-port cpd enp63s0.10
ovs-vsctl add-port cpd enp63s0.11
ovs-vsctl add-port cpd enp63s0.12
```

O creando el fichero */etc/sysconfig/network-scripts/ifcfg-cpd* y editando los ficheros */etc/sysconfig/network-scripts/ifcfg-enp63s0.10*, */etc/sysconfig/network-scripts/ifcfg-enp63s0.11* y */etc/sysconfig/network-scripts/ifcfg-enp63s0.12* de la siguiente forma:

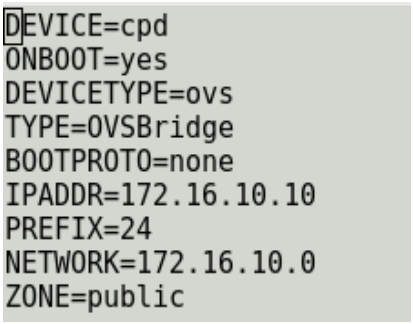


Fig. 5.5. Configuración permanente del *Open vSwitch.*

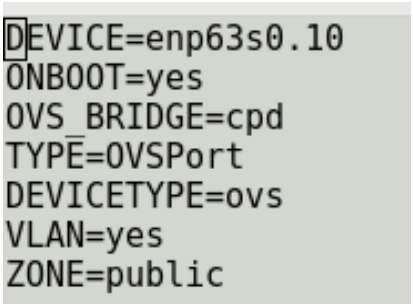


Fig. 5.6. Configuración permanente del subinterfaz de tipo VLAN como puerto del *Open vSwitch*.

La configuración de las subinterfaces asociadas a las VLANs 11 y 12 es idéntica, cambiando el nombre del subinterfaz y el nombre del *switch* virtual al cual queremos asociar nuestro subinterfaz. Cabe destacar que en ambos despligues, los *switches* virtuales tendrán los nombres "cpd" para el extremo asociado al CPD y "srem" para el extremo asociado a la sede o sedes remotas.

5.4.5. Configuración del *switch* físico.

Para la interconexión de los equipos se utilizará un *switch* Cisco Catalyst 2960. Dado que el tráfico se genera etiquetado en las máquinas virtuales, se deben configurar todos los puertos del *switch* como *trunk* con el fin de que encamine por esos puertos todo el tráfico etiquetado. En los equipos con *Open vSwitch* instalado, circulará tráfico de todas las VLANs posibles. Además, en este caso, se aislará el tráfico del controlador con los *switches* virtuales con otra VLAN, con identificador 44, por lo que se creará un subinterfaz para esta VLAN en cada equipo; en este caso sin añadirlo al *switch* virtual, a los cuales se asignará un direccionamiento IP, en la misma subred:

TABLA 5.2. CONFIGURACIÓN DE LOS EQUIPOS PARA AISLAR EL TRÁFICO DEL CONTROLADOR EN UNA VLAN.

Equipo	**Interfaz**	**Identificador VLAN**	**Dirección IP**
Open vSwitch 1	enp63s0.44	44	172.16.44.10/24
Open vSwitch 2	enp63s0.44	44	172.16.44.20/24
Controlador	enp63s0.44	44	172.16.44.1/24

Para realizar esta configuración, se instalará *Minicom* y se configurará para establecer conexión con el *switch* a través del puerto serie:

```
apt-get install minicom

minicom –s
```

En el menú de configuración se selecciona la configuración del puerto serie. Dentro de este submenú, se configurarán todas las opciones de la siguiente manera:

```
+-----------------------------------------------------------------------+
| A - Dispositivo Serial           : /dev/ttyS0                         |
| B - Localización del Archivo de Bloqueo : /var/lock                   |
| C - Programa de Acceso           :                                    |
| D - Programa de Salida             :                                  |
| E - Bps/Paridad/Bits             : 9600 8N1                           |
| F - Control de Flujo por Hardware: No                                 |
| G - Control de Flujo por Software: No                                 |
|                                                                       |
|    ¿Qué configuración alterar?                                        |
+-----------------------------------------------------------------------+
```

Fig. 5.7. Configuración del puerto serie con *Minicom* para la conexión con el *switch* Cisco.

Nótese que en “Dispositivo Serial”, se debe escribir el nombre del puerto serie del equipo a utilizar. En caso de no conocerlo, ejecutar el siguiente comando:

```
dmesg | grep tty
```

Una vez establecida la conexión con el *switch*, se observará el símbolo del sistema “*Switch*>”. Es en este interfaz donde se realizará la configuración del mismo. Primero

se crearán todas las VLANs a utilizar en el *switch* y posteriormente se configurarán los puertos pertinentes. En este caso serán 3 puertos a configurar, uno por equipo:

Switch>enable

Switch#configure terminal

Switch(config)#vlan 10

Switch(config)#name vlan10

Switch(config)#vlan 11

Switch(config)#name vlan11

Switch(config)#vlan 12

Switch(config)#name vlan12

Switch(config)#interface fa0/1

Switch(config-if)#*switch*port mode trunk

Switch(config-if)#*switch*port trunk allowed vlan 10,11,12,44

Switch(config-if)#exit

Switch(config)#interface fa0/2

Switch(config-if)#*switch*port mode trunk

Switch(config-if)#*switch*port trunk allowed vlan 10,11,12,44

Switch(config-if)#exit

Switch(config)#interface fa0/3

Switch(config-if)#*switch*port mode trunk

Switch(config-if)#*switch*port trunk allowed vlan 44

Switch(config-if)#end

5.4.6. Conexión del controlador.

Como se ha explicado en el apartado anterior, la conexión del controlador con los *switches* virtuales se realiza en la VLAN 44, aislando este tipo de tráfico. Para realizar esta conexión, se ejecutan los siguientes comandos en los equipos con *Open vSwitch:*

```
ovs-vsctl set-controller cpd tcp:172.16.44.1:6633
ovs-vsctl set bridge protocols=OpenFlow15
ovs-vsctl set controller cpd connection-mode=out-of-band
```

Donde, es este caso, "cpd" es el nombre del *switch* virtual creado.

Se podrá comprobar la conectividad del controlador ejecutando el siguiente comando en el equipo que actúa como controlador:

```
ryu-manager –verbose
```

Tras su ejecución, deberán aparecer las características de ambos *switches* virtuales, incluída su dirección IP asociada al subinterfaz de tipo VLAN con identificador 44 configurado anteriormente.

En caso de no establecerse conexión con los *switches*, se deberá hacer un PING desde el controlador a los *switches.* Si el PING falla, se trata de un problema de conectividad de nivel 3 por lo que se deberá observar si el direccionamiento asignado a los subinterfaces se encuentra en la misma subred o si las rutas están correctamente definidas. De ser así, se deberá comprobar de nuevo la configuración del *switch* físico, comprobando que la VLAN 44 se encuentra asignada a los 3 puertos configurados como *trunk.* Si el PING funciona, se tratará de un problema de conectividad de nivel 4. Lo más probable es que, en este caso, sea el *firewall* de *CentOS* el que esté bloqueando el puerto de escucha de los equipos con *Open vSwitch.* Para ello, se deberá dar permiso en el *firewall* al puerto de escucha 6633 o desactivar el *firewall*, dado que en este proyecto no será necesario.

5.4.7. Desarrollo del código del despliegue.

Una vez conectados los *switches* virtuales con el controlador, se programará el código necesario para encaminar el tráfico entre las máquinas virtuales. El objetivo de este

despliegue es realizar un PING de una máquina virtual de un extremo a la del otro extremo, y procesar este tráfico en los *Open vSwitches*. El PING se realizará desde los subinterfaces de tipo VLAN por lo que habrá que tratar el tráfico a partir de su identificador. Por tanto, en función del indentificador VLAN de los paquetes (10,11 o 12), se encaminará el tráfico por el subinterfaz añadido como puerto al *switch* correspondiente (enp63s0.10, enp63s0.11 o enp63s0.12). El diagrama de flujo del código es el siguiente:

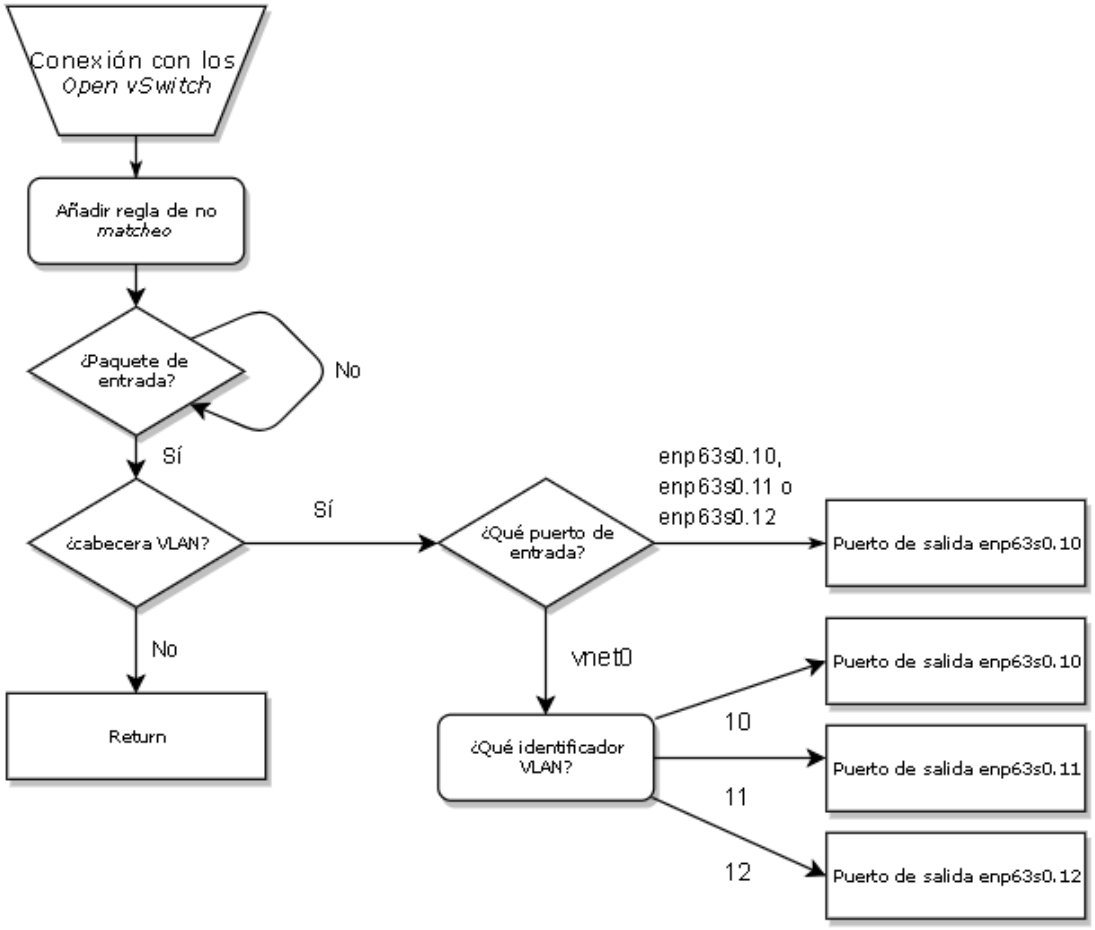


Fig. 5.8. Diagrama de flujo del código para el encaminamiento del tráfico etiquetado.

Cabe destacar que en el código, los puertos se identifican numéricamente, no por nombre. Para ver el identificador de cada puerto, se ejecutará el siguiente comando:

```
ovs-ofctl –O OpenFlow13 dump-ports-desc cpd
```

Es necesario mencionar también, que los paquetes ARP (Protocolo de resolución de direcciones), intercambiados inicialmente con el fin de descubrir las direcciones MAC

de los equipos, están etiquetados, por lo que no es necesario tratar este tipo de tráfico de forma diferente dado que su encaminamiento es idéntico al intercambio ICMP del PING.

5.4.8. Funcionamiento del despliegue.

Tras conectar el controlador ejecutando el *script* correspondiente al diagrama de flujo anterior, se realiza un PING desde una de las máquinas virtuales a la dirección IP del subinterfaz de tipo VLAN de la otra máquina virtual. En este caso, se realiza un PING en las VLANs 11 y 12. Tras establecer conexión con la máquina virtual correspondiente, pueden observarse las reglas añadidas a los *switches* virtuales utilizando el siguiente comando:

```
ovs-ofctl –O OpenFlow13 dump-flows cpd
```

Tras ejecutar el comando en el *Open vSwitch* que actúa como sede remota, por ejemplo, se observan las siguientes reglas:

```
[root@localhost ~]# ovs-ofctl -O OpenFlow15 dump-flows cpd
 cookie=0x0, duration=69.092s, table=0, n_packets=75, n_bytes=7398, priority=1,i
n_port="enp63s0.11",dl_vlan=11,dl_dst=52:54:00:66:06:8f actions=output:vnet0
 cookie=0x0, duration=69.088s, table=0, n_packets=74, n_bytes=7212, priority=1,i
n_port=vnet0,dl_vlan=11,dl_dst=52:54:00:bf:b7:2d actions=output:"enp63s0.11"
 cookie=0x0, duration=57.370s, table=0, n_packets=62, n_bytes=6100, priority=1,i
n_port=vnet0,dl_vlan=12,dl_dst=52:54:00:bf:b7:2d actions=output:"enp63s0.12"
 cookie=0x0, duration=57.351s, table=0, n_packets=61, n_bytes=6054, priority=1,i
n_port="enp63s0.12",dl_vlan=12,dl_dst=52:54:00:66:06:8f actions=output:vnet0
 cookie=0x0, duration=120.267s, table=0, n_packets=186, n_bytes=11936, priority=
0 actions=CONTROLLER:65535
```

Fig. 5.9. Reglas añadidas a la tabla de flujos del Open v*Switch* del CPD para encaminar tráfico etiquetado a nivel 2.

Se pueden observar cuatro reglas. Dos de ellas se corresponden al tráfico correspondiente a la VLAN 11 y las otras dos a la VLAN 12. Si se analizan las correspondientes a la VLAN 11, se observa que una de ellas corresponde al tráfico entrante en el *switch* con dirección a la máquina virtual (*in_port=enp63s0.11* y *output=vnet0*) y otra al tráfico saliente hacia el otro *Open vSwitch* (*in_port=vnet0* y *output=enp63s0.11*). Las mismas reglas se añaden al hacer PING en la VLAN 10 o la VLAN 11, independientemente de la máquina virtual en la que se origine el tráfico.

5.5. Despliegue completo.

5.5.1. Introducción.

Una vez establecida una toma de contacto con el *OpenFlow* y el controlador *Ryu*, se procederá a realizar el despliegue completo de la red, así como la programación del código correspondiente para encaminar todo el tráfico a contemplar en la red.

En este despliegue, se simulará el esquema de red general del apartado 6.2, el cual, sustituyendo el equipo CPD y el equipo de oficina remota por *switches* virtuales, es el siguiente:

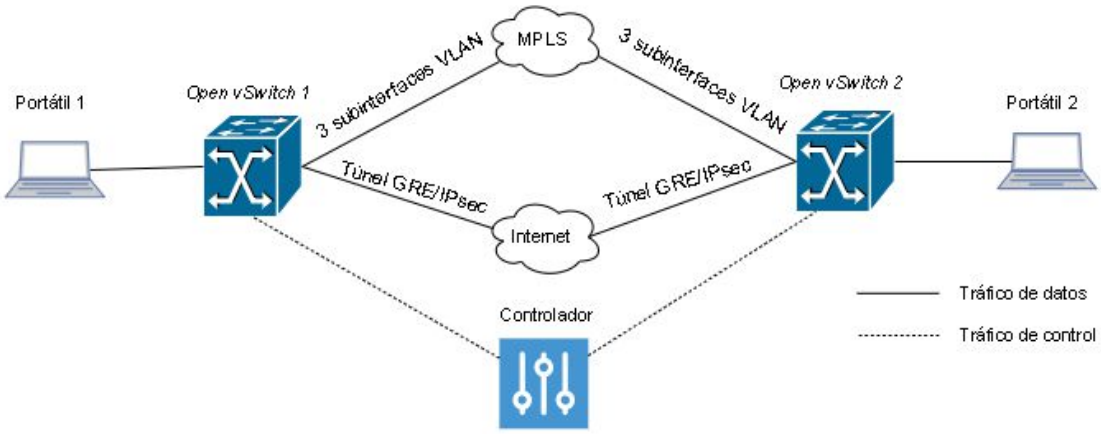


Fig. 5.11. Esquema de red del despliegue completo.

Al igual que en el despliegue con máquinas virtuales, los enlaces entre los *switches* virtuales es directo dado que la red MPLS e internet actúan de manera independiente al encaminamiento con *OpenFlow*, que sí encaminará hacia una red u otra en función del tipo de tráfico que reciba.

En este despliegue no se emplearán máquinas virtuales, sino equipos físicos, en este caso ordenadores portátiles, los cuales se encargarán de generar tráfico entre ellos.

Esto requiere la adición de otro *switch* Cisco Catalyst 2960, uno en cada extremo, que etiquete el tráfico entrante de los portátiles con el fin de simular las 3 VPNs de la red. Además, se añadirá una tarjeta de red a cada equipo que simule *switch* virtual. Así, una de las tarjetas de red irá conectada de forma directa al otro *switch* virtual, mientras que la otra tarjeta de red recibirá el tráfico entrante del *switch* físico.

En este despliegue se añadirá un túnel GRE cifrado con IPsec como puerto al *switch* virtual. Este túnel será el puerto que encaminará el tráfico a internet. Para el encaminamiento hacia la red MPLS se continuarán usando las subinterfaces de tipo VLAN al igual que en el despliegue con máquinas virtuales.

Cabe destacar que, como ya se ha mencionado anteriormente, se utilizará *Ubuntu* como sistema operativo.

El objetivo consiste en encaminar diferentes tipos de servicio a través de los *Open vSwitch*. Estos servicios son HTTP en los puertos 80 y 8080, FTP y TPFT. En función del servicio a ofrecer, de la VLAN de los paquetes y de si el tráfico pertenece al direccionamiento privado de ADIF (192.168.0.0/16, 10.0.0.0/8 y 172.16.0.0/12), se encaminará el tráfico hacia la red MPLS o hacia internet.

5.5.2. Instalación del *software* necesario.

El equipo del controlador se mantiene idéntico respecto al despliegue con máquinas virtuales.

Por otro lado, dado que en este caso se empleará Ubuntu en los equipos que simularán los *switches* virtuales, se reinstalará tanto *Open vSwitch* como el soporte para túneles GRE cifrados con IPsec:

```
apt-get install openv switch-switch
apt-get install openv switch-ipsec
```

5.5.3. Configuración de los *switches* físicos.

Se siguen manteniendo las 3 VLANs, salvo que se emplearán nuevos valores de etiquetado en este despliegue como puede apreciarse en la tabla 6.3. Así, se configurarán 3 puertos de cada *switch* como puertos de acceso para el tráfico de entrada a la red y un puerto como *trunk*, que etiquete el tráfico hacia el *switch* virtual. Además, se seguirá empleando la VLAN 44 de control para el tráfico entre el controlador y los *switches* virtuales.

TABLA 5.3. TABLA DE VLANS A UTILIZAR				
	Tráfico de la red corporativa	Tráfico de Renfe	Tráfico de Seguridad	Tráfico de control
Etiqueta VLAN	101	201	301	44

```
Switch>enable
Switch#configure terminal
Switch(config)#vlan 101
Switch(config)#name corporativa
Switch(config)#vlan 201
Switch(config)#name renfe
Switch(config)#vlan 301
Switch(config)#name seguridad
```

Para configurar los puertos de acceso:

```
Switch#configure terminal
Switch(config)#interface fa0/1
Switch(config-if)#switchport mode 54orks54
Switch(config-if)#switchport 54orks54 vlan 101
Switch(config-if)#exit
Switch(config)#interface fa0/2
Switch(config-if)#switchport mode 54orks54
Switch(config-if)#switchport 54orks54 vlan 201
Switch(config-if)#exit
```

```
Switch(config)#interface fa0/3
Switch(config-if)#switchport mode 55orks55
Switch(config-if)#switchport 55orks55 vlan 301
```

Para configurar el puerto *trunk* conectado al equipo con el *switch* virtual:

```
Switch#configure terminal
Switch(config)#interface fa0/4
Switch(config-if)#switchport mode trunk
Switch(config-if)#switchport trunk allowed vlan 44,101,201,301
```

Ahora se deberán configurar los puertos asociados a la VLAN de control. Se desea que el tráfico vaya etiquetado desde el equipo, por lo que se configurará un interfaz de tipo VLAN en cada equipo.

En Ubuntu, esto se consigue editando el fichero */etc/network/interfaces* de la siguiente manera:

```
auto enp63s0.44
iface enp63s0.44 inet static
address X.X.X.A
netmask 255.255.255.0
vlan-raw-device enp63s0
```

En el equipo con el controlador se seguirá empleando el subinterfaz VLAN creado para el primer despliegue.

En este caso, se emplea el direccionamiento 100.100.100.0/24, aunque cualquier otro direccionamiento puede ser utilizado, siempre y cuando las direcciones asignadas a cada subinterfaz de tipo VLAN pertenezcan a la misma subred, como se explicó en el primer despliegue. Así:

TABLA 5.4. TABLA DE DIRECCIONAMIENTO PARA EL TRÁFICO DE CONTROL.			
Equipo/Sistema Operativo	*Switch* virtual 1/*Ubuntu*	*Switch* virtual 2/*Ubuntu*	Controlador/*Centos*
Dirección IP/Prefijo	100.100.100.10/24	100.100.100.20/24	100.100.100.1/24

Dado que en el equipo con el controlador se utilizará una única tarjeta de red y éste debe establecer comunicación con ambos *Open vSwitches*, se creará un enlace directo entre ambos *switches* Cisco entre dos puertos *trunk* en la VLAN 44, así como la utilización de otro puerto *trunk* en la VLAN 44 para conectar el controlador. Así, el tráfico de control puede circular entre ambos *switches* Cisco, estableciendo una topología *full-mesh,* entre los *Open vSwitches* y el controlador. La configuración es la siguiente:

En el *switch* 1 (al que se conectará el controlador):

```
Switch#configure terminal
Switch(config)#interface fa0/6
Switch(config-if)#switchport mode trunk
Switch(config-if)#switchport trunk allowed vlan 44
Switch(config-if)#exit
Switch(config)#interface fa0/7
Switch(config-if)#switchport mode trunk
Switch(config-if)#switchport trunk allowed vlan 44
```

En el *switch* 2:

```
Switch#configure terminal
Switch(config)#interface fa0/7
Switch(config-if)#switchport mode trunk
Switch(config-if)#switchport trunk allowed vlan 44
```

5.5.4. Configuración de los *switches* virtuales.

La creación de los *switches* virtuales se hará con el *software Open vSwitch* como hasta ahora. Como se ha visto en el primer despliegue, para crear un *switch* virtual de *OpenFlow*, se puede ejecutar el siguiente comando:

```
ovs-vsctl add-bridge cpd
```

Y para observar que se ha creado correctamente:

```
ovs-vsclt show
```

Sin embargo, esta configuración se perderá al reiniciar los equipos, siendo necesario ejecutar de nuevo los comandos. Por tanto, para que la configuración sea permanente, y al tratarse de un sistema operativo *Ubuntu*, se editará el fichero */etc/network/interfaces*.

En este caso se deben añadir tantos subinterfaces de tipo VLAN como VLANs se vayan a manejar, dado que Ubuntu posee un único fichero de configuración en el cual definir todos los interfaces de red. Dado que en este despliegue se disponen de dos tarjetas de red, en una de ellas se recibirá el tráfico etiquetado en los *switches* Cisco procedente de los equipos portátiles mientras que la otra se utilizará como conexión directa entre los *Open vSwitch*.

Es importante saber que si a un subinterfaz de tipo VLAN le llega tráfico etiquetado procedente del interfaz físico al que está asociado este subinterfaz, éste quitará la etiqueta VLAN del paquete, lo cual no es el objetivo del despliegue. Por tanto, solo se definirán subinterfaces VLAN en el enlace directo entre los *Open vSwitches*, dado que así el tráfico etiquetado será enviado y recibido por éstas y, por tanto, no se eliminará la etiqueta. Para el tráfico entrante desde los portátiles, se añadirá el interfaz físico sin modificación al *Open vSwitch* como puerto, recibiendo todo el tráfico etiquetado en él. En este caso, se definirán los subinterfaces *ens4.101*, *ens4.201* y *ens4.301* para la conexión directa entre los *switches* virtuales, y se utilizará el interfaz *enp63s0* para el tráfico entrante de los extremos.

Además, dado que por el enlace directo entre los *Open vSwitches* se definirá un túnel GRE cifrado con IPsec, se dará direccionamiento estático al interfaz *ens4*. Este direccionamiento será el que utilice el túnel GRE para realizar el encapsulamiento del

tráfico. En este caso se dará la dirección 192.168.1.1/24 al interfaz del *Open vSwitch* del CPD y la dirección 192.168.1.2/24 al interfaz del *Open vSwitch* de la sede remota.

Así, la configuración de los interfaces de red será la siguiente:

```
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

allow-cpd enp63s0
iface enp63s0 inet manual
        ovs_bridge cpd
        ovs_type OVSPort

auto ens4
iface ens4 inet static
        address 192.168.1.1
        netmask 255.255.255.0

auto cpd
allow-ovs cpd
iface cpd inet manual
        ovs_type OVSBridge
        ovs_ports enp63s0 ens4.101 ens4.201 ens4.301

auto enp63s0.44
iface enp63s0.44 inet static
        address 100.100.100.10
        netmask 255.255.255.0
        vlan-raw-device enp63s0

allow-cpd ens4.101
iface ens4.101 inet manual
        ovs_bridge cpd
        ovs_type OVSPort
        vlan-raw-device ens4

allow-cpd ens4.201
iface ens4.201 inet manual
        ovs_bridge cpd
        ovs_type OVSPort
        vlan-raw-device ens4

allow-cpd ens4.301
iface ens4.301 inet manual
        ovs_bridge cpd
        ovs_type OVSPort
        vlan-raw-device ens4
```

Fig. 5.12. Fichero de configuración */etc/network/interfaces* para el despliegue final.

Una vez configurados los interfaces a utilizar, se deberá crear el túnel GRE cifrado y añadirlo al *Open vSwitch* como puerto. Para ello se ejecutarán los siguientes comandos:

```
ip tunnel add tun0 mode gre remote 192.168.1.2 local 192.168.1.1 dev ens4 ttl 255

ovs-vsctl add-port cpd tun0 – set Interface tun0 type=ipsec_gre options=remote_ip:192.168.1.2 options:psk=test
```

El direccionamiento, el nombre del *Open vSwitch* y la clave PSK para el cifrado del tráfico a través del túnel puede varias según se desee.

Posteriormente, se reinicia la configuración de red del equipo, se conecta el controlador a ambos *Open vSwitch*, se configura el protocolo *OpenFlow* versión 1.3 en los *Open vSwitch* y se comprueba que la configuración es correcta:

```
systemctl restart networking
ovs-vsctl set-controller cpd tcp:100.100.100.1:6633
ovs-vsctl set bridge protocols=OpenFlow15
ovs-vsctl set controller cpd connection-mode=out-of-band
ovs-vsctl show
```

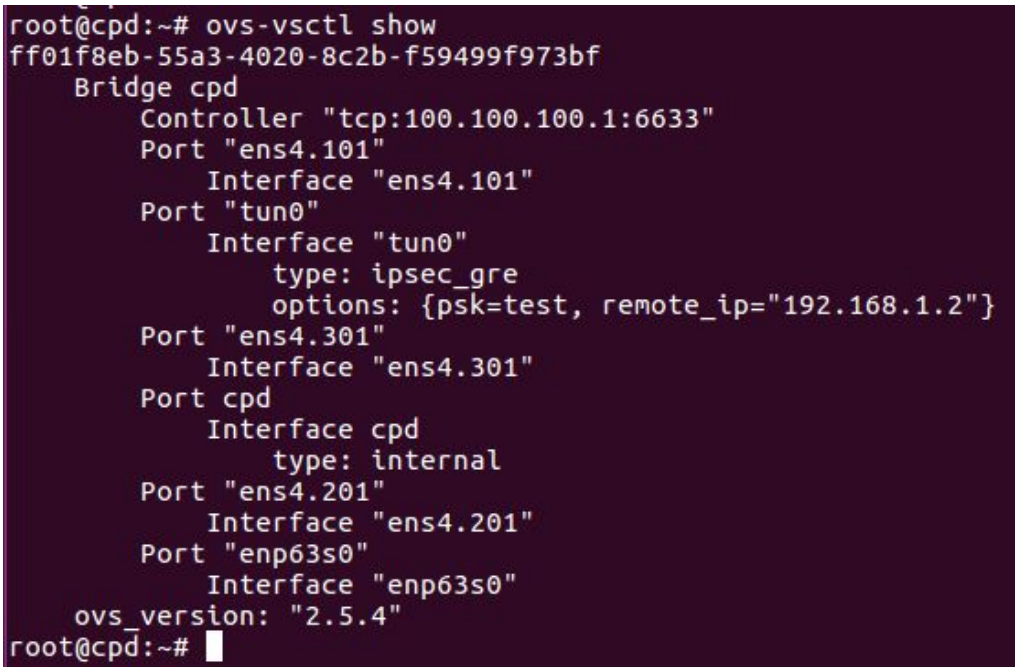


Fig. 5.13. Configuración del *switch* virtual y sus puertos.

Mediante esta configuración, el *switch* virtual podrá recibir paquetes etiquetados por todos los puertos, incluído el túnel GRE cifrado (*tun0)*.

5.5.5. Desarrollo del código para este despliegue.

Como ya se ha explicado en la introducción de este despliegue, la intención es ofrecer diversos servicios entre los extremos de la red, haciendo que un equipo portátil actúe como CPD/servidor y otro como sede remota/cliente. Así, las reglas de encaminamiento a añadir en los *switches* virtuales son las siguientes:

- Si el tráfico pertenece al direccionamiento privado de ADIF (192.168.0.0/16, 172.16.0.0/12 o 10.0.0.0/8), se efectúa el encaminamiento según la siguiente tabla:

TABLA 5.5. TABLA DE ENCAMINAMIENTO SEGÚN EL SERVICIO Y LA VLAN.

	HTTP en puerto 80	**HTTP en puerto 8080**	**FTP**	**TFTP**
Corporativa	MPLS	Internet	Internet	MPLS
Renfe	Internet	Internet	Internet	Internet
Seguridad	Internet	MPLS	MPLS	MPLS

- Si el direccionamiento IP no pertenece al direccionamiento privado de ADIF, el tráfico se encamina hacia internet por el túnel GRE, sin realizar ninguna comprobación más.

Mediante el código de este despliegue se añadirán reglas en ambos *Open vSwitches* en tres diferentes tablas, creando un encaminamiento multitabla, en el cual en cada tabla se *matchearán* una o dos cabeceras del tráfico. El esquema de encaminamiento multitabla y el diagrama de flujo del código son los siguientes:

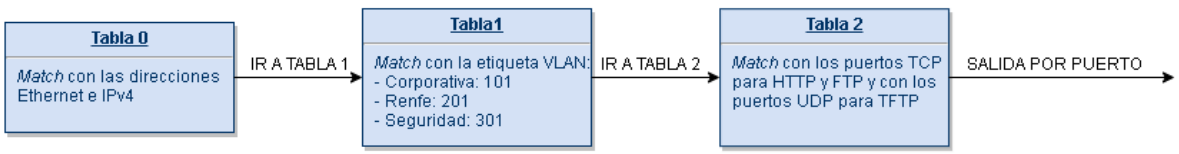


Fig. 5.14. Esquema de encaminamiento multitabla.

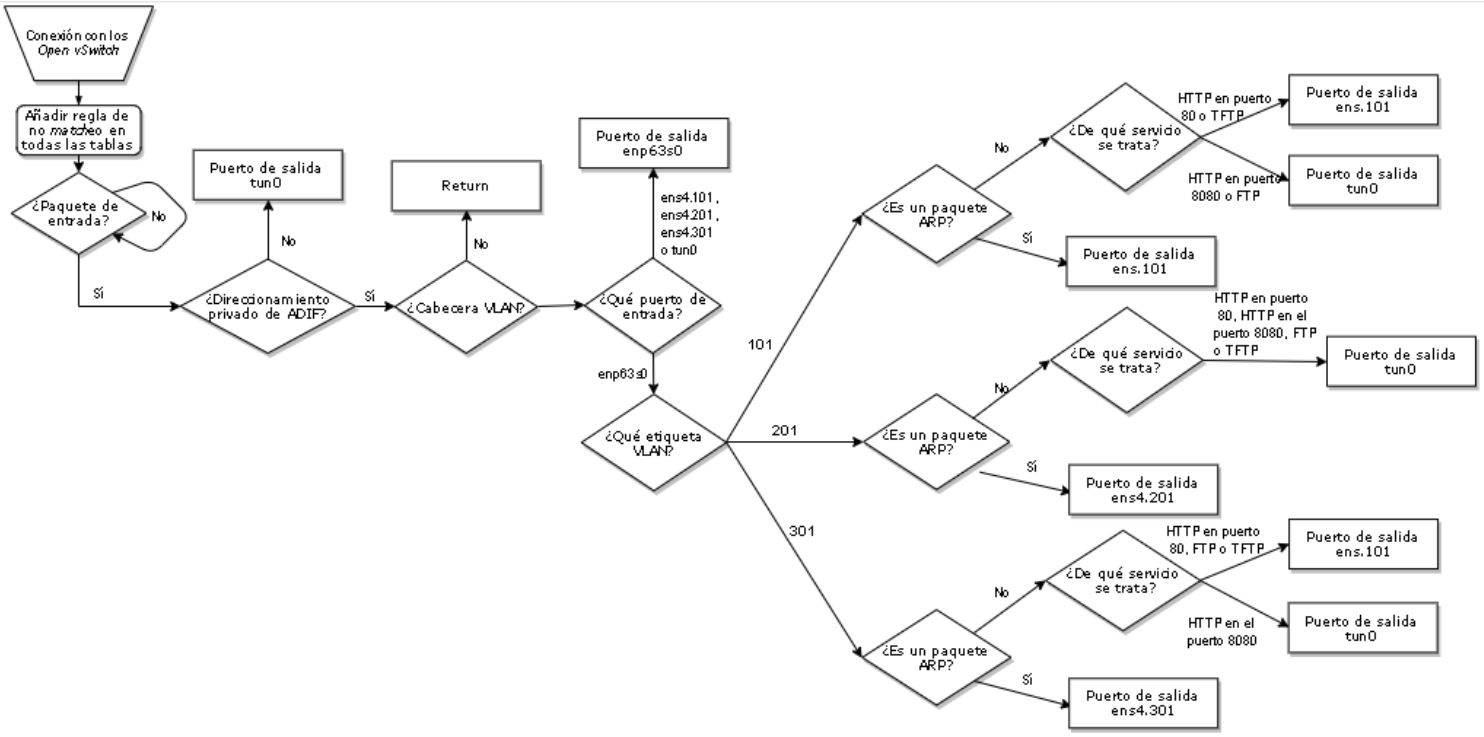


Fig. 5.15. Diagrama de flujo del despliegue final.

Nótese que todo el tráfico ARP se encamina por la salida hacia la red MPLS, dado que éste no lleva cabeceras TCP o UDP, lo cual impide identificar el servicio a dar. Sin embargo, puede encaminarse por el túnel GRE con tan solo cambiar la numeración del puerto de salida en el código.

5.5.6. Funcionamiento del despliegue.

Con el fin de evaluar el código desarrollado y su correcto funcionamiento, se irán conectando los equipos portátiles a cada uno de los puertos de acceso configurados en los *switches* Cisco. Por tanto, se separará la comprobación de funcionamiento en función de la VPN, comprobando la tabla 6.5 a la hora de insertar las reglas correspondientes. Dado que la inserción de las reglas es similar en todos los servicios, no se comprobarán todos éstos para cada VPN.

Para realizar esta tarea, en el cliente se utilizará el navegador *Mozilla Firefox* [28] para realizar solicitudes HTTP y FTP y el programa *tftpd32/64* [29] para solicitudes TFTP. En el servidor, se utilizará *FileZilla Server* para crear el servidor FTP al que se

realizarán las peticiones FTP, de nuevo *tftpd32/64* para crear el servidor TFTP y el servidor *Apache* [30] para crear un servidor HTTP en los puertos 80 y 8080.

5.5.6.1. Configuración del *software* de pruebas.

- Configuración del servidor *Apache*:

Tras la instalar el servidor *Apache*, se configurará para poder recibir solicitudes en los puertos 80 y 8080. Para ello, se accederá a la carpeta raíz del programa y se abrirá el fichero "httpd.conf". Dentro de este fichero se buscará el apartado "Listen" y se escribirán las siguientes líneas:

```
# Listen: Allows you to bind Apache to specific IP addresses and/or ports, instead of the default. See also the <VirtualHost> directive.

# Change this to Listen on specific IP addresses as shown below to prevent Apache from glomming onto all bound IP addresses.

Listen 80

Listen 8080
```

- Configuración del servidor FTP *Filezilla*:

En "Edit/Settings", en la opción "Listen on these ports", poner el puerto de escucha deseado. En este caso se define el puerto 21, que es el puerto de escucha por defecto de FTP. También se deberá de configurar el servidor en modo pasivo. Posteriormente, en "Edit/Users/General se crearán los credenciales del cliente (nombre de usuario y contraseña) y en "Edit/Users/Shared Folders" se definirá el directorio FTP al cual se accederá desde el cliente.

- Configuración del servidor y cliente TFTP con *tftpd32/64*:

En el servidor, se debe definir el directorio TFTP al cual se accederá desde el cliente en la opción "Browse". En "Settings", se selecciona únicamente la opción "TFTP Server", "TFTP Security" a "None", en "TFTP Port" se define el puerto de escucha del servidor. En este caso se define el puerto 69 que es el puerto por defecto de escucha del protocolo TFTP. Es indispensable habilitar la

opción “Bind TFTP to this address” y seleccionar la dirección IP del puerto Ethernet del equipo.

En el cliente la configuración es la misma, salvo que en este caso no hay que definir ningún directorio TFTP y hay que habilitar únicamente la opción “TFTP Client”.

- Configuración de los equipos.

Dado que para eñadir las correspondientes reglas, los equipos deben generar tráfico con direccionamiento privado de ADIF, se asignarán las direcciones 10.1.1.1/8 en el servidor y 10.1.1.2/8 en el cliente.

5.5.6.2. Comprobación de funcionamiento.

- VPN corporativa:

Se realiza una petición HTTP al puerto 80 y una petición FTP, por ejemplo. Para realizar la petición HTTP, en el navegador *Mozilla Firefox* se escribe “http://10.1.1.1” y para realizar una solicitud FTP se escribe “ftp://10.1.1.1” y se introducen las credenciales configuradas en el servidor.

Tras realizar la solicitud HTTP, se deberá observar una imagen de *Apache Haus* con el mensaje “It 63orks”, mientras que tras realizar la petición FTP deberá a parecer el directorio compartido por el servidor FTP. Se comprueban las reglas añadidas a los *Open vSwitches*:

- En el *Open vSwitch* del servidor:

```
root@adif-HP-Compaq-6005-Pro-SFF-PC:~# ovs-ofctl -O Openflow13 dump-flows srem
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=131.253s, table=0, n_packets=54, n_bytes=4372, priority=1,ip,dl_dst=28:d2:44:bb:d8:38,nw_dst=10.1.1.1 actions=g
oto_table:1
 cookie=0x0, duration=131.221s, table=0, n_packets=63, n_bytes=51096, priority=1,ip,dl_dst=00:02:3f:13:36:89,nw_dst=10.1.1.2 actions=
goto_table:1
 cookie=0x0, duration=179.653s, table=0, n_packets=12865, n_bytes=825460, priority=0 actions=CONTROLLER:65535
 cookie=0x0, duration=131.221s, table=1, n_packets=117, n_bytes=55468, priority=1,dl_vlan=101 actions=goto_table:2
 cookie=0x0, duration=179.653s, table=1, n_packets=0, n_bytes=0, priority=0 actions=CONTROLLER:65535
 cookie=0x0, duration=157.987s, table=2, n_packets=27, n_bytes=2536, priority=1,tcp,tp_dst=80 actions=output:2
 cookie=0x0, duration=141.514s, table=2, n_packets=20, n_bytes=1352, priority=1,tcp,tp_dst=21 actions=output:5
 cookie=0x0, duration=157.961s, table=2, n_packets=36, n_bytes=47961, priority=1,tcp,tp_src=80 actions=output:1
 cookie=0x0, duration=141.470s, table=2, n_packets=19, n_bytes=1663, priority=1,tcp,tp_src=21 actions=output:1
 cookie=0x0, duration=179.653s, table=2, n_packets=15, n_bytes=1956, priority=0 actions=CONTROLLER:65535
root@adif-HP-Compaq-6005-Pro-SFF-PC:~#
```

Fig. 5.16. Reglas añadidas al *Open vSwitch* del servidor tras solicitudes HTTP:80 y FTP.

- En el *Open vSwitch* del cliente:

Fig. 5.17. Reglas añadidas al *Open vSwitch* del cliente tras solicitudes HTTP en el puerto 80 y FTP.

Se comprueba de igual forma el funcionamiento del túnel GRE cifrado:

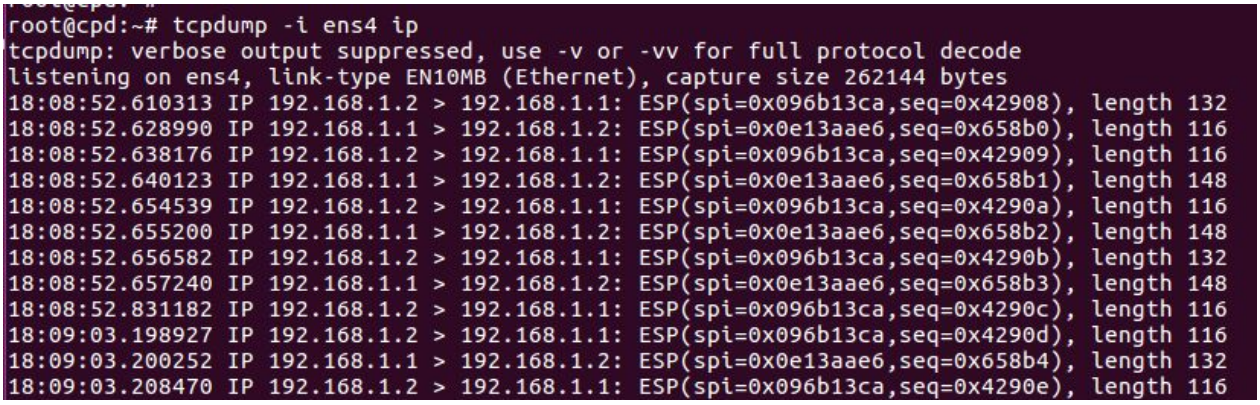


Fig. 5.18. Funcionamiento del túnel GRE cifrado con IPsec.

Como se puede observar, haciendo *tcpdump* para ver el tráfico IP en la conexión directa entre los *OpenvSwitches*, el tráfico se encuentra encapsulado, función principal de los túneles GRE, con el dreccionamiento asignado a la hora de crear el túnel. Además, se puede observar la cabecera del protocolo ESP (Encapsulating Security Payload) de IPsec, el cual proporciona confidencialidad, cifrando el tráfico con clave, la cual se define a la hora de definir el interfaz de tipo túnel como puerto del *Open vSwitch*.

- VPN de Renfe:

Dado que todo el tráfico de Renfe se envía hacia internet, solo se comprueba el funcionamiento de TFTP, dado que el funcionamiento de HTTP y FTP es correcto en el caso del tráfico corporativo. Para realizar una solicitud TFTP, se

abre *tftp32/64* en ambos equipos portátiles. En el cliente se introduce el puerto al cual se realiza la solicitud, en este caso el puerto 69, el fichero a descargar del servidor y el fichero local donde se guardará el fichero remoto. Tras esto, se pulsa en la opción "GET" para proceder a la descarga del archivo del servidor TFTP remoto. Se comprueban las reglas añadidas al *switch* virtual:

- En el *Open vSwitch* del servidor:

```
root@cpd:~# ovs-ofctl -O OpenFlow13 dump-flows cpd
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=40.552s, table=0, n_packets=1, n_bytes=65, priority=1,ip,dl_dst=00:02:3f:13:36:89,nw_dst=10.1.1.2 actions=goto_
table:1
 cookie=0x0, duration=40.527s, table=0, n_packets=2, n_bytes=128, priority=1,ip,dl_dst=28:d2:44:bb:d8:38,nw_dst=10.1.1.1 actions=goto
_table:1
 cookie=0x0, duration=84.351s, table=0, n_packets=182, n_bytes=13709, priority=0 actions=CONTROLLER:65535
 cookie=0x0, duration=40.527s, table=1, n_packets=3, n_bytes=193, priority=1,dl_vlan=201 actions=goto_table:2
 cookie=0x0, duration=84.350s, table=1, n_packets=0, n_bytes=0, priority=0 actions=CONTROLLER:65535
 cookie=0x0, duration=40.595s, table=2, n_packets=0, n_bytes=0, priority=1,udp,tp_dst=69 actions=output:1
 cookie=0x0, duration=84.350s, table=2, n_packets=3, n_bytes=193, priority=0 actions=CONTROLLER:65535
root@cpd:~#
```

Fig. 5.19. Reglas añadidas al *Open vSwitch* del servidor tras solicitud TFTP.

- En el *Open vSwitch* del cliente:

```
root@adif-HP-Compaq-6005-Pro-SFF-PC:~# ovs-ofctl -O Openflow13 dump-flows srem
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=4.617s, table=0, n_packets=1, n_bytes=65, priority=1,ip,dl_dst=00:02:3f:13:36:89,nw_dst=10.1.1.2 actions=goto_t
able:1
 cookie=0x0, duration=4.609s, table=0, n_packets=2, n_bytes=128, priority=1,ip,dl_dst=28:d2:44:bb:d8:38,nw_dst=10.1.1.1 actions=goto_
table:1
 cookie=0x0, duration=49.224s, table=0, n_packets=1388, n_bytes=89664, priority=0 actions=CONTROLLER:65535
 cookie=0x0, duration=4.609s, table=1, n_packets=3, n_bytes=193, priority=1,dl_vlan=201 actions=goto_table:2
 cookie=0x0, duration=49.224s, table=1, n_packets=0, n_bytes=0, priority=0 actions=CONTROLLER:65535
 cookie=0x0, duration=4.678s, table=2, n_packets=1, n_bytes=64, priority=1,udp,tp_dst=69 actions=output:5
 cookie=0x0, duration=49.224s, table=2, n_packets=2, n_bytes=129, priority=0 actions=CONTROLLER:65535
root@adif-HP-Compaq-6005-Pro-SFF-PC:~#
```

Fig. 5.20. Reglas añadidas al *Open vSwitch* del cliente tras solicitud TFTP.

Se pueden observar las reglas añadidas de direccionamiento Ethernet e Ipv4, así como la regla para la VPN de Renfe con identificador 201 y la regla para el tráfico hacia el puerto 69 del servidor. Por esta regla solo pasa la solicitud RRQ (*Read Request*) de establecimiento de conexión, el ACK del servidor al cliente y la posterior transmisión del fichero no se realiza por esa regla, sino que todo ese tráfico pasa por el controlador.

- VPN de Seguridad:

Dado que se comprobó el funcionamiento de HTTP en el puerto 80 en la VPN corporativa, se comprobará en este caso en el puerto 8080, así como FTP y TFTP. Tras la solicitud de los servicios, se comprueban las reglas añadidas:

- En el *Open vSwitch* del servidor:

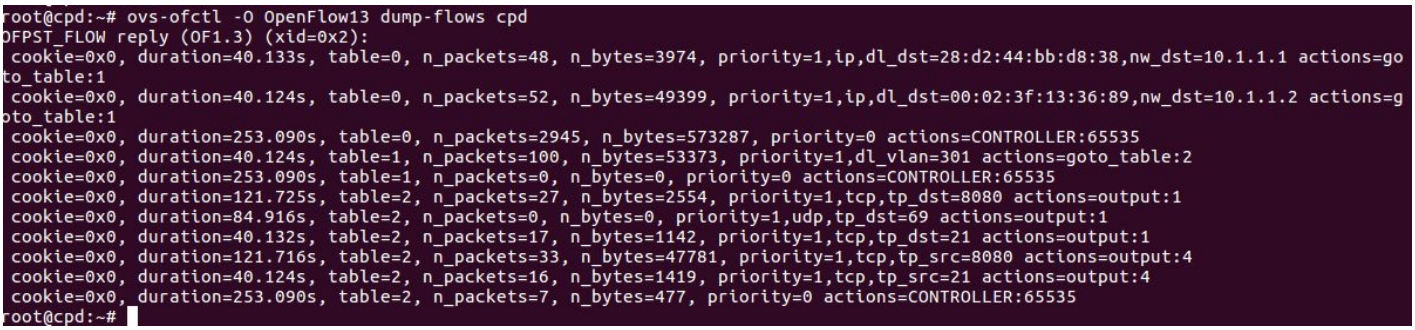


Fig. 5.21. Reglas añadidas al *Open vSwitch* del servidor tras solicitud HTTP en el puerto 8080, FTP y TFTP.

- En el *Open vSwitch* del cliente:

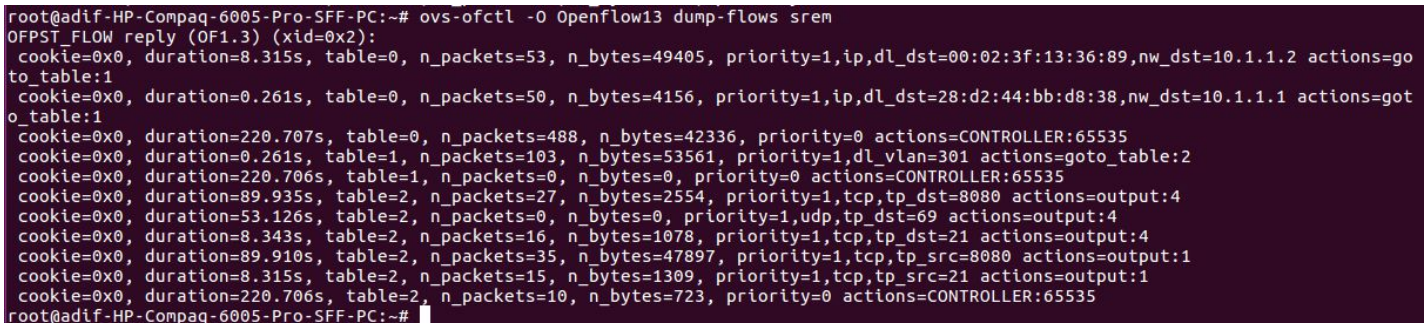


Fig. 5.20. Reglas añadidas al *Open vSwitch* del cliente tras solicitud HTTP en el puerto 8080, FTP y TFTP.

Las reglas son idénticas a las ya vistas en los casos anteriores, salvo que para HTTP, ahora el puerto TCP del servidor es el 8080 y, además, la regla añadida en la tabla 1 es de la VPN de Seguridad con identificador 301.

En el Anexo B se puede ver un análisis más detallado del encaminamiento del tráfico y de las reglas añadidas en función de la tabla 5.5.

6. EVALUACIÓN, RESULTADOS Y POSIBLES MEJORAS.

6.1. Evaluación de la solución.

6.1.1. Despliegue con máquinas virtuales.

En la figura 6.1 puede observarse que al hacer el PING, los paquetes iniciales, los cuales pasan por el controlador, tienen un TTL aproximado de 30ms a 50ms, mientras que los que son encaminados a partir de las reglas añadidas al *switch* virtual tienen un TTL de alrededor 1 ms. Esto implica que todo el tráfico que no coincida con una regla y sea enviado al controlador tendrá un TTL mucho mayor que si se realiza un encaminamiento "normal" y que el tráfico encaminado a través de las reglas del *switch* tendrá un TTL de aproximadamente el doble que con encaminamiento "normal".

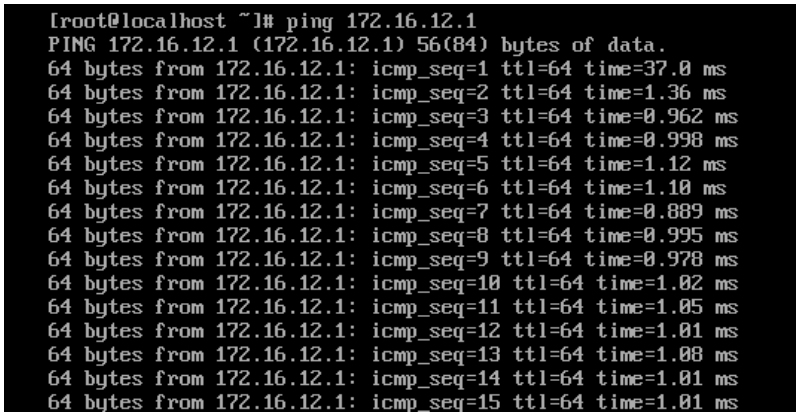


Fig. 6.1. PING de entre máquinas virtuales a través del controlador y las reglas añadidas a los *Open vSwitch*.

6.1.2. Despliegue completo.

El código de este despliegue no contempla la adición de reglas para los puertos abiertos a la hora de realizar transmisión de ficheros con FTP y TFTP, los cuales son diferentes a los puertos 21 y 69, respectivamente, de apertura de conexión. Esto implica que para la transmisión de ficheros, todo el tráfico pasa por el controlador. Como se ha visto en el apartado anterior, el tráfico, al pasar por el controlador, tiene un TTL mayor. En este caso ocurre lo mismo, la transmisión de ficheros es mucho más lenta que si el tráfico se encaminase mediante reglas en los *Open vSwitches*. Sin embargo, no es deseable crear tablas de reglas muy extensas, sobre todo si el sistema se implementa en una Red de Área Amplia (WAN) en la cual es común la existencia de gran cantidad de flujos de tráfico. En el capítulo "Posibles Mejoras", se propondrá una solución a este problema.

Por otro lado, siempre y cuando el tráfico sea encaminado por reglas añadidas al *Open vSwitch*, el rendimiento es aceptable aunque de menor calidad que si se realiza encaminamiento "normal".

6.2. Posibles mejoras.

6.2.1. Utilización de otro controlador.

Respecto al *software* utilizado en la maqueta, podría estudiarse la viabilidad de utilizar otro controlador diferente a *Ryu*. Dado que el objetivo del proyecto es utilizar *software* libre, controladores comerciales como APIC de Cisco, VAN de HP o NSX de Vmware no son opciones válidas.

La elección del controlador a utilizar es, en parte, algo relativo, ya que depende de los objetivos del despliegue, dado que cada controlador ofrece, o no, ciertas funcionalidades que otros no ofrecen, teniendo cada uno de ellos ventajas y desventajas. En este proyecto no se evaluará directamente el rendmiento de un controlador u otro. Sin embargo, en caso de querer realizar esta tarea, se puede recurrir a herramientas como *Cbench* [31] o *Hcprobe* [32] que se encargan de monitorizar y evaluar el rendimiento, la escalabilidad, la disponibilidad y la seguridad de controladores *OpenFlow*.

En la siguiente figura se podrán observar las características que poseen los controladores de *OpenFlow* más conocidos:

TABLA 6.1. TABLA COMPARATIVA DE CONTROLADORES *OPENFLOW* [33].

Controllers / Use-Cases	Trema	Nox/Pox	RYU	Floodlight	ODL	ONOS***
Network Virtualizaiton by Virtual Overlays	YES	YES	YES	PARTIAL	YES	NO
Hop-by-hop Network Virtualization	NO	NO	NO	YES	YES	YES
OpenStack Neutron Support	NO	NO	YES	YES	YES	NO
Legacy Network Interoperability	NO	NO	NO	NO	YES	PARTIAL
Service Insertion and Chaining	NO	NO	PARTIAL	NO	YES	PARTIAL
Network Monitoring	PARTIAL	PARTIAL	YES	YES	YES	YES
Policy Enforcement	NO	NO	NO	PARTIAL	YES	PARTIAL
Load Balancing	NO	NO	NO	NO	YES	NO
Traffic Engineering	PARTIAL	PARTIAL	PARTIAL	PARTIAL	YES	PARTIAL
Dynamic Network Taps	NO	NO	YES	YES	YES	NO
Multi-Layer Network Optimization	NO	NO	NO	NO	PARTIAL	PARTIAL
Transport Networks - NV, Traffic-Rerouting, Interconnecting DCs, etc.	NO	NO	PARTIAL	NO	PARTIAL	PARTIAL
Campus Networks	PARTIAL	PARTIAL	PARTIAL	PARTIAL	PARTIAL	NO
Routing	YES	NO	YES	YES	YES	YES

Como se puede apreciar en esta tabla, en el caso de querer desplegar un esquema SDN sencillo, basado en encaminar diferente tipo de tráfico y diferentes servicios, controladores como el ya usado *Ryu*, *Floodlight* o *Trema* pueden realizar esta tarea. Sin

embargo, se puede observar que el controlador más completo es *OpenDaylight* (ODL). Es el controlador de *OpenFlow* más reciente (creado en 2014), y ofrece posibilidades que ningún controlador es capaz de ofrecer: monitorización de red, aplicación de políticas, balance de enlace, ingeniería de tráfico, interoperabilidad de redes heredadas, integración con *OpenStack*,…

Por tanto, en caso de realizar esquemas más complejos, como en el caso de una red WAN de una empresa, la utilización del controlador *OpenDaylight* puede ser más factible que el uso de un controlador con menos funcionalidades.

6.2.2. Mejoras en el código.

Una vez comprobado el funcionamiento de la solución y del código programado en el controlador, se pueden plantear diversas mejoras en el mismo:

- **Adición y eliminación dinámicas de reglas en los *Open vSwitches.*** En los servicios simulados en la maqueta, las conexiones se abren en puertos concretos. Sin embargo, en FTP y TFTP, a la hora de transmitir ficheros, se abren puertos libres que no se conocen de antemano. El controlador añade reglas para esos puertos, las cuales no se volverán a usar ya que para transmitir el siguiente fichero se abrirá otro puerto distinto, lo cual causaría una tabla con numerosas reglas irrelevantes. Existen dos soluciones:
    - No añadir reglas para puertos diferentes a los ya conocidos, lo cual hará que todo el tráfico sea enrutado por el controlador y los ficheros tarden mucho más en transmitirse.
    - Añadir y eliminar las reglas para estos puertos desconocidos dinámicamente. Esto se consigue guardando los puertos abiertos por cliente y servidor para la transmisión globalmente y, tras una transmisión, una vez llegue tráfico al controlador con otros valores de puerto para otra transmisión, eliminar las reglas de la transmisión anterior, mediante mensaje mensajes de modificación de flujos con la opción *DELETE* y actualizando los valores estáticos de los puertos. De esta forma se consigue realizar la transmisión de los ficheros de manera efectiva sin tener una tabla de reglas muy extensa con reglas irrelevantes en futuras transmisiones.

- **Reencaminamiento por caída de puertos.** Otra de las posibles mejoras es implementar reencaminamiento de tráfico en caso de caída de puertos mediante grupos de tipo *Fast Failover* [34]. Un grupo consiste en uno o más *buckets*. Un *bucket* es una lista de acciones. En los despliegues realizados, en la última tabla se definía la acción de encaminar por un puerto de salida. Sin embargo, en caso de implementar este tipo de reencaminamiento, .se enviará el tráfico a un grupo, que elija una de las acciones del *bucket*. En el tipo de grupo *Fast Failover*, por cada entrada dentro de cada *bucket*, se definen las variables *watch_port* y *watch_group*, que indican que puerto o grupo serán monitorizados para detectar caídas. Así, en este caso, se definirán los grupos pertinentes de tipo *Fast Failover* con un dos *buckets*:

TABLA 6.2. ESTRUCTURA DE GRUPO *FAST FAILOVER*.	
Group_id = X	***type=ff***
bucket = *watch_port:MPLS,actions=output:MPLS*	
bucket = *watch_port:Internet,actions=output:Internet*	

Como primera opción el tráfico se encamina hacia la red MPLS, monitorizando este mismo puerto de salida. En caso de caída del puerto, el tráfico se encaminaría por el segundo *bucket*, es decir, por internet, monitorizando de igual forma este puerto.

7. PLANIFICACIÓN DEL PROYECTO.

7.1. Estructura del trabajo.

Para la realización del trabajo, se han seguido una serie de procesos o fases diferenciadas, con el objetivo de adquirir un aprendizaje gradual de las redes definidas por *software*, mediante documentación y la aplicación práctica de las mismas a partir del despliegue de una maqueta de red.

La estructura de procesos o fases es la siguiente:

- Estudio del funcionamiento de SDN.
- Estudio del protocolo *OpenFlow* y del controlador para crear el plano de control de la red.
- Despliegue de la red de con máquinas virtuales.
- Despliegue de la red completa.
- Evaluación de rendimiento y viabilidad.
- Redacción de la memoria.

Se puede apreciar una parte teórica inicial, necesaria para la adquisición de unos conceptos mínimos previos a la aplicación práctica de SDN y su evaluación.

Como se puede apreciar, el hecho de haber implementado y evaluado el diseño por partes de forma gradual hasta llegar al diseño final, conlleva un desarrollo en espiral, en el cual, en cada vuelta o iteración se han tenido en cuenta los objetivos y planificación, el desarrollo y la evaluación del mismo.

Fig. 7.1. Representación del desarrollo en espiral del proyecto.

7.2. Tareas, subtareas y diagrama de Gantt.

En el apartado anterior se han definido las tareas o procesos generales llevados a cabo durante la realización del proyecto. Estas tareas pueden dividirse en subtareas, llevadas a cabo durante un proceso de tiempo tal y como se expone a continuación:

- Estudio del funcionamiento de SDN (6 días).
    - Búsqueda de información relativa a SDN (3 días) – Tarea 1.
    - Ventajas y aplicabilidad de SDN (3 días) – Tarea 2.
- Estudio del protocolo *OpenFlow* y del controlador para crear el plano de control de la red (39 días).
    - Lectura de la documentación de *OpenFlow* 1.5 (25 días) – Tarea 3.
    - Lectura de la documentación y bibliotecas del controlador (14 días) – Tarea 4.
- Despliegue de la red con máquinas virtuales (27 días).

    - Despliegue del *hardware* e instalación del *software* necesario (1 día) – Tarea 5.
    - Instalación y configuración de las máquinas virtuales y de los *switches* virtuales (2 días) – Tarea 6.
    - Conexión del controlador y configuración del *switch* Cisco Catalyst 2960 (14 días) – Tarea 7.
    - Planificación, desarrollo y depuración del código en Python para este despliegue (10 días) – Tarea 8.
- Despliegue de la red completa (24 días).
    - Despliegue del *hardware* e instalación del *software* necesario (2 días) – Tarea 10.
    - Configuración de los *switches* virtuales y del *switch* Cisco Catalyst 2960 (1 día) – Tarea 11.
    - Planificación, desarrollo y depuración del código en *Python* para este despliegue (21 días) – Tarea 12.
- Evaluación de rendimiento y viabilidad de la maqueta de red completa (2 días) – Tarea 13.
- Evaluación de posibles mejoras (5 días) – Tarea 14.
- Redacción de la memoria (21 días) – Tarea 15.

Este desglose de tareas puede obtenerse en *Microsoft Project* y observarse en la siguiente página:

Id		Nombre del recurso	Trabajo
5		Ingeniero de Telecomunicaciones	992 horas

Identificador	Nombre de tarea	Unidades	Trabajo	Retraso	Comienzo	Fin
2	Búsqueda de información relativa a SDN	100%	24 horas	0 días	jue 01/02/18	lun 05/02/18
3	Ventajas y aplicabilidad de SDN	100%	24 horas	0 días	mar 06/02/18	jue 08/02/18
5	Estudio de la documentación de OpenFlow	100%	200 horas	0 días	vie 09/02/18	jue 15/03/18
6	Estudio de la documentación y bibliotecas del controlador	100%	112 horas	0 días	vie 16/03/18	mié 04/04/18
9	Despliegue del hardware e instalación del software necesario	100%	8 horas	0 días	jue 07/06/18	jue 07/06/18
10	Instalación y configuración de las máquinas virtuales y de los switches virtual	100%	16 horas	0 días	vie 08/06/18	lun 11/06/18
11	Conexión del controlador y configuración del switch Cisco Catalyst 2960	100%	112 horas	0 días	mar 12/06/18	vie 29/06/18
12	Planificación, desarrollo y depuración del código en Python para este despliegu	100%	80 horas	0 días	lun 02/07/18	vie 13/07/18
14	Despliegue del hardware e instalación del software necesario	100%	16 horas	0 días	lun 16/07/18	mar 17/07/18
15	Configuración de los switches virtuales y del switch Cisco Catalyst 2960	100%	8 horas	0 días	mié 18/07/18	mié 18/07/18
16	Planificación, desarrollo y depuración del código en Python para este despliegu	100%	168 horas	0 días	jue 19/07/18	jue 16/08/18
17	Evaluación de rendimiento y viabilidad de la maqueta de red completa	100%	16 horas	0 días	vie 17/08/18	lun 20/08/18
18	Evaluación de posibles mejoras	100%	40 horas	0 días	mar 21/08/18	lun 27/08/18
19	Redacción de la memoria	100%	168 horas	0 días	mar 28/08/18	mar 25/09/18

Fig. 7.2. Desglose de tareas con *Microsoft Project.*

Con este esquema de tareas desglosadas en subtareas, se puede formalizar el siguiente diagrama de Gantt, utilizando *Microsoft Proyect*, y asignando a cada tarea definida en el diagrama los recursos necesarios para su realización:

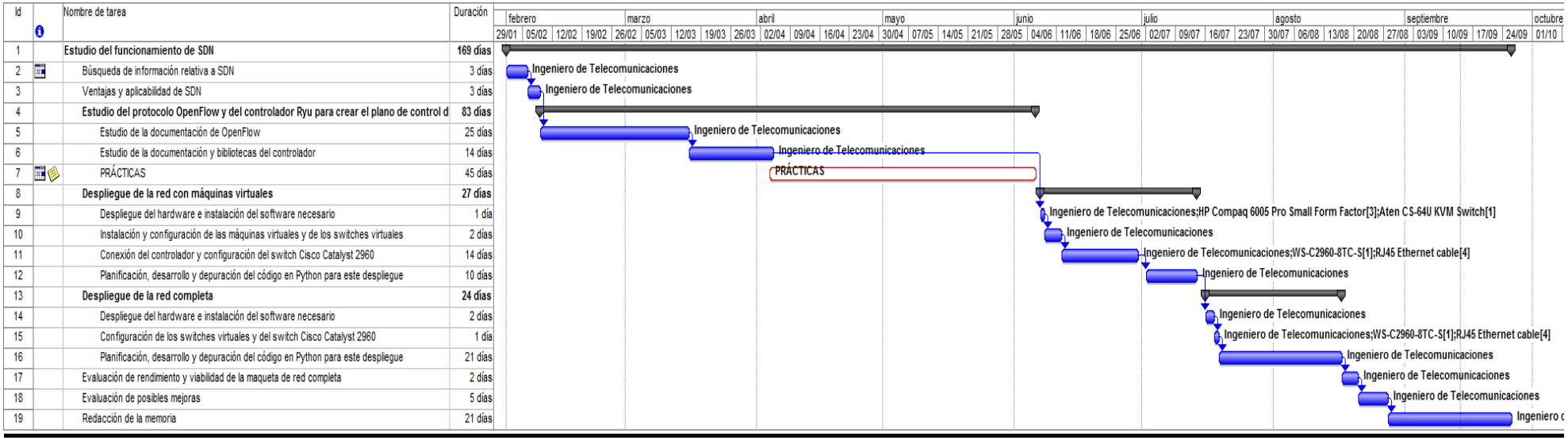


Fig. 7.3. Diagrama de *Gantt* de las tareas llevadas a cabo durante el proyecto, con los respectivos recursos utilizados para su realización.

8. ENTORNO SOCIO-ECONÓMICO.

8.1. Presupuesto del proyecto.

Durante el desarrollo del proyecto se ha utilizado la herramienta *Microsoft Project* con el fin de documentar información de tareas, presupuesto y recursos. En este apartado vendrá deglosado y explicado el presupuesto. En el apartado siguiente, vendrá de forma detallada la planificación del proyecto en un diagrama de *Gantt*. El presupuesto del hardware utilizado se aprecia en la siguiente tabla:

TABLA 8.1. PRESUPUESTO DEL HARDWARE UTILIZADO.

Item	**Precio por unidad**	**Número de unidades**	**Enlace de compra**	**Precio total**
Cisco Catalyst 2960-S 8-PORT *Switch* (Cisco WS-C2960-8TC-S)	880$/767,50€	1	https://www.amazon.com/Cisco-WS-C2960-8TC-S-8_PORT-Catalyst-*Switch*/dp/B002G3IF6A	880$/767,50€
RJ45 Ethernet cable	4.49$/3,90€	3	https://www.amazon.com/AmazonBasics-RJ45-Cat-6-Ethernet-Patch-Cable-3-Feet-0-9-Meters/dp/B00N2VISLW	13,47$/11,7€
HP Compaq 6005 Pro Small Form Factor	180$/157€	3	https://www.amazon.com/HP-Compaq-Performance-Professional-Refurbished/dp/B01LK4APPY	540$/471€
Aten CS-64U KVM *Switch*	56,98€	1	https://www.amazon.es/Aten-AT-CS64U-Perif%C3%A9rico-de-entrada/dp/B0037BU9LE	56,98€

Por otro lado, con *Microsoft Project* se ha desglosado el presupuesto en función de las tareas llevadas a cabo, que se verán en el siguiente apartado de *Planificación* definiendo un gasto de 30€ /hora como ingeniero de telecomunicaciones recién incorporado en plantilla, según se indica en la siguiente tabla:

TABLA 8.2. DESGLOSE SALARIAL ESTIMADO DE ADIF EN FUNCIÓN DE LA CATEGORÍA PROFESIONAL.

		Bruto	Coste empresa	Total Anual	Total Hora	
Mando Intermedio	Mínimo	32.982,00	14.841,90	47.823,90	27,80	
	Medio	36.000,00	16.200,00	52.200,00	30,35	**30**
	Máximo	42.745,00	19.235,25	61.980,25	36,04	
Estructura Apoyo	Mínimo	40.022,00	18.009,90	58.031,90	33,74	
	Medio	47.000,00	21.150,00	68.150,00	39,62	**40**
	Máximo	56.082,00	25.236,90	81.318,90	47,28	

En la tabla se indica una estimación salarial, dado que por sensibilidad del tema no hay estadísticas que indiquen valores medios.

El desglose del presupuesto en función de las tareas llevadas a cabo durante la realización del proyecto, incluyendo el gasto salarial de ingeniero de telecomunicaciones, puede verse en la siguiente página:

Id	Nombre de tarea	Costo fijo	Acumulación de costos fijos	Costo total
2	Búsqueda de información relativa a SD	0,00 €	Prorrateo	720,00 €
3	Ventajas y aplicabilidad de SDN	0,00 €	Prorrateo	720,00 €
5	Estudio de la documentación de Openl	0,00 €	Prorrateo	6.000,00 €
6	Estudio de la documentación y bibliote	0,00 €	Prorrateo	3.360,00 €
7	PRÁCTICAS	0,00 €	Comienzo	0,00 €
9	Despliegue del hardware e instalación	0,00 €	Prorrateo	767,98 €
10	Instalación y configuración de las máq	0,00 €	Prorrateo	480,00 €
11	Conexión del controlador y configuraci	0,00 €	Prorrateo	4.143,10 €
12	Planificación, desarrollo y depuración	0,00 €	Comienzo	2.400,00 €
14	Despliegue del hardware e instalación	0,00 €	Comienzo	480,00 €
15	Configuración de los switches virtuales	0,00 €	Comienzo	1.023,10 €
16	Planificación, desarrollo y depuración	0,00 €	Comienzo	5.040,00 €
17	Evaluación de rendimiento y viabilidad	0,00 €	Comienzo	480,00 €
18	Evaluación de posibles mejoras	0,00 €	Comienzo	1.200,00 €
19	Redacción de la memoria	0,00 €	Comienzo	5.040,00 €
		0,00 €		**31.854,18 €**

Fig. 8.1. Presupuesto del proyecto.

8.2. Impacto socio-económico.

El creciente desarrollo de SDN ha supuesto un gran impacto socio-económico tanto en grandes como en pequeñas empresas.

Económicamente, el modelo SDN supone una reducción de costes al cambiar de un modelo distribuído a uno centralizado, lo cual hace que el provisionamiento de la red sea más sencillo y eficaz. Además, la posibilidad de virtualizar elementos de la red supone un ahorro en hardware y, por tanto, en su mantenimiento y reparación en caso de averías.

Al ser una tecnología cada vez más creciente y valorada, grandes compañías como Cisco o Teldat han invertido en su desarrollo, proporcionando sus soluciones comerciales a empresas que deseen transformar sus redes distribuidas tradicionales en esquemas centralizados.

Estas soluciones comerciales son, en su mayoría, económicamente inaccesibles para pequeñas empresas o *startups*, que quieran implementar este tipo de solución en sus redes de comunicaciones. Por el contrario, el impacto económico será positivo en empresas que vean viable la reducción de costes que supone una transición hacia SDN frente a la inversión necesaria para la adquisición de una solución comercial.

Es aquí donde entra en escena *OpenFlow*. Como se ha comentado, *OpenFlow* es un estándar libre, no comercial, por lo que su desarrollo e implementación está abierta a todos los usuarios. Para empresas pequeñas o *startups* cuyo presupuesto no permita adquirir soluciones comerciales, esto supone un impacto económico muy positivo, debido a la posibilidad de adquirir un equipo de desarrollo del protocolo *OpenFlow*, o de la implementación de soluciones ya existentes basadas en éste.

Por otro lado, ya han surgido pequeñas empresas que se encargan únicamente del desarrollo e implementación de soluciones SDN basadas en *software* libre para ponerlas a disposición de otras empresas interesadas ya que, lógicamente, estas soluciones de *software* libre son económicamente más viables que una solución totalmente comercial.

Socialmente, el avance tecnológico supone un gran impacto. En este caso, so muchas las compañías que poseen un gran número de clientes. La transición a SDN puede suponer grandes cambios en la calidad de servicio ofrecida. Por tanto, es inviable ofrecer a los clientes un servicio de peor calidad al ofrecido mediante redes tradicionales.

SDN, al ser una tecnología reciente, requiere de gran estudio así como de pruebas técnicas. Otro de los objetivos, independientemente del beneficio para las empresas, es ofrecer un servicio de mejor calidad, basado en una mayor eficiencia en el uso de la red y en el control de acceso, reduciendo la caída de enlaces y su recuperación en caso de caída, la congestión de los mismos, etc. Por tanto, una solución cuyo servicio sea de peor calidad tendrá un impacto social negativo, lo cual no es el objetivo de la implementación de una solución de red definida por *software*.

9. CONCLUSIONES

9.1. El futuro de las redes definidas por *software*.

Tras el análisis de SDN llevado a cabo durante la realización de este proyecto mediante el protocolo *OpenFlow*, se puede obtener una visión más clara de lo que supone SDN para las redes de comunicaciones.

Se observa una tendencia clara, basada en la migración de las filosofías a un modelo centralizado y más flexible, con mayor facilidad y capacidad para adaptarse a las necesidades y demandas de los usuarios.

SDN no tiene un éxito asegurado, ni la garantía de ser el modelo predominante en el futuro. Esto se debe a otras alternativas similares a SDN que están apareciendo en el mercado, con ventajas similares pero con un plano de control más distribuído.

La tendencia al cambio es evidente. Este cambio requiere un estudio y avance rápido de las redes dado que a medida que avanzan los años los volúmenes de tráfico aumentan cada vez más.

Una de las características más notables de SDN, la cual se ha comprobado durante la realización del proyecto, es la gran flexibilidad que ofrece. Esta flexibilidad proporciona a los proveedores la posibilidad de estar en cualquier punto de la red y la capacidad de actuar de forma rápida y eficiente ante cambios en la red.

SDN, aún sin tener un éxito garantizado, se encontrará de forma segura entre los modelos predominantes a la hora de desplegar redes de comunicaciones.

9.2. Aprendizaje adquirido.

Al ser una tecnología reciente y en auge, es de gran valor la adquisición de conocimientos sobre SDN en el ámbito de las TIC.

Trabajar en este tipo de tecnologías nuevas y en una empresa como ADIF te permite adquirir grandes competencias y experiencia en cuanto al desarrollo, planificación y ejecución de proyectos reales en un ámbito laboral real.

Además, la realización del proyecto ha permitido una gran ampliación de conocimientos técnicos relacionados con redes de comunicaciones, programación, sistemas operativos Linux, configuración de equipos, capacidad de análisis, etc.

Estos conocimientos facilitan, en mayor o menor medida, futuras exigencias de empresas y suponen una visión general de cómo podría trabajarse en el futuro en cuanto a redes de comunicaciones se refiere.

10. BIBLIOGRAFÍA

- [1] Figuras relativas a documentación y presentaciones del departamento de ja Jefatura de Comunicaciones de ADIF.
- [2] Cisco Visual Networking Index Highlights Tool – Última consulta: 15/09/2018. [En línea]. Disponible en: https://www.cisco.com/c/m/en_us/solutions/service-provider/vni-forecast-highlights.html
- [3] P.A. Cuenca Castillo y R. Puigjaner Trepat, *Tendencias en redes de altas prestaciones*, 1999. [En línea]. Disponible en: http://books.google.es/books?id=F9c1RheLgxsC&lpg=PA115&ots=DUcCF5xKo4&dq=trafico%20multimedia%20jitter&hl=es&pg=PA115#v=onepage&q&f=false
- [4] Ancho de banda Skype – Última consulta: 21/08/2018. [En línea]. Disponible en: https://support.skype.com/es/faq/FA1417/que-cantidad-de-ancho-de-banda-necesita-skype
- [5] *Internet Trasit Prices*. Última consulta: 23/08/2018. [En línea]. Disponible en: http://drpeering.net/white-papers/Internet-Transit-Pricing-Historical-And-Projected.php
- [6] Servicio MacroLan de Telefónica. Última consulta: 23/08/2018. [En línea]. Disponible en: http://www.movistar.es/grandes-empresas/soluciones/fichas/wan-lan
- [7] Observatorio Nacional de las Telecomunicaciones - Indicadores de hogares con acceso a banda ancha fija. Última consulta: 24/08/2018. [En línea]. Disponible en: http://www.ontsi.red.es/ontsi/es/indicador/hogares-acceso-banda-ancha-fija
- [8] F. Jejdling, *Ericsson Mobility Report,* Junio 2018. [En línea]. Disponible en: https://www.ericsson.com/assets/local/mobility-report/documents/2018/ericsson-mobility-report-june-2018.pdf

- [9] B. Heller et al, *SDN for engineers*, Abril 2012. [En línea]. Disponible en: https://web.archive.org/web/20130116072451/http://www.stanford.edu/~brandonh/ONS/*OpenFlow*Tutorial_ONS_Heller.pdf
- [10] T.Bakhshi, *State of the Art and Recent Research Advances in Software Defined Networking,* Enero 2017. [En línea]. Disponible en: https://www.hindawi.com/journals/wcmc/2017/7191647/#B181
- [11] S. Shin and G. Gu, "Attacking *software*-defined networks: a first feasibility study", *Proceedings of the 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking* (HotSDN '13), pp. 165–166, Agosto 2017.
- [12] R. Smeliansky, "SDN for network security", *Proceedings of the International Science and Technology Conference* (Modern Networking Technologies) (MoNeTeC '14), pp. 1–5, Moscú, Rusia, Octubre 2014.
- [13] L. Schehlmann, S. Abt, and H. Baier, "Blessing or curse? Revisiting security aspects of *software*-defined networking", *Proceedings of the 10th International Conference on Network and Service Management* (CNSM '14), pp. 382–387, Río de Janeiro, Brasil, Noviembre 2014.
- [14] R. van der Pol, "Software defined networking and OpenFlow", *SNE Colloquium*, Mayo 2014. [En línea]. Disponible en: https://pdfs.semanticscholar.org/presentation/84f6/b872c72875acd124d32c72ae51e53b527d3e.pdf
- [15] Y. Nakajima, "Standardization progress in Sofware Define Networking/*OpenFlow*", *NTT Technical Review*. [En línea]. Disponible en: https://www.ntt-review.jp/archive/ntttechnical.php?contents=ntr201302gls.html
- [16] *OpenFlow* Protocol API Reference. [En línea]. Última consulta: 17/09/2018. Disponible en: http://ryu.readthedocs.io/en/latest/ofproto_ref.html
- [17] SDN IEEE Standarization. Última consulta: 30/08/2018. [En línea]. Disponible en: https://sdn.ieee.org/standardization
- [18] A. Bruno, "¿Qué es internet y la IETF?", 2-11-2011. [En línea]. Disponible en: https://derechoeinternet.wordpress.com/2011/11/02/%C2%BFque-es-internet-los-protocolos-de-internet-y-la-ietf/

- [19] E. Haleplidis et al, ".RFC 7426 (*Software*-Defined Networking (SDN): Layers and Architecture Terminology)", Enero 2015. [En línea]. Disponible en: https://www.rfc-editor.org/info/rfc7426
- [20] Ley de Protección de Datos de Carácter Personal - https://www.boe.es/buscar/act.php?id=BOE-A-1999-23750
- [21] Reglamento Europeo de Protección de Datos - https://eur-lex.europa.eu/legal-content/ES/TXT/HTML/?uri=CELEX:32016R0679&from=ES
- [22] Ley General de Telecomunicaciones - https://www.boe.es/buscar/act.php?id=BOE-A-2014-4950
- [23] "Regulatory compliance for SDN", *Zenlayer,* 28.12.2017. [En línea]. Disponible en: https://www.zenlayer.com/regulatory-compliance-*software*-defined-networks-sdn/
- [24] J. Duffy, "Cisco reveals OpenFlow SDN killer: OpFlex Protocol for ACI offered to IETF, OpenDaylight", *Networld.* 2-4-2014. [En línea]. Disponible en: https://www.networkworld.com/article/2175716/lan-wan/cisco-reveals-*OpenFlow*-sdn-killer.html
- [25] "OpFlex: An Open Policy Protocol White Paper", Cisco, ID del decumento: 1518187482185567, 17-10-2014. [En línea]. Disponible en: https://www.cisco.com/c/en/us/solutions/collateral/data-center-virtualization/application-centric-infrastructure/white-paper-c11-731302.html
- [26] Ryu Application API. Última consulta: 19/09/2018. [En línea]. Disponible en: https://ryu.readthedocs.io/en/latest/ryu_app_api.html
- [27] CentOS 7 Minimal download. Última consulta: 25/08/2018. [En línea]. Disponible en: https://www.centos.org/download/
- [28] Enlace de descarga de *Mozilla Firefox.* Última consulta: 25/08/2018. [En línea]. Disponible en: – https://www.mozilla.org/es-ES/firefox/
- [29] Enlace de descarga de *tftp32/64.* Última consulta: 25/08/2018. [En línea]. Disponible en: http://www.tftpd64.com/

- [30] Enlace de descarga de Apache Server. Última consulta: 26/08/2018. [En línea]. Disponible en: https://httpd.apache.org
- [31] – CBench *OpenFlow* controller benchmarker. Última consulta: 10/09/2018. [En línea]. Disponible en: https://github.com/trema/cbench
- [32] – Hcprobe framework for testing *OpenFlow* controllers. Última consulta: 10/09/2018. [En línea]. Disponible en: https://github.com/ARCCN/hcprobe
- [33] S. Rao, "SDN Series Part Eight: Comparison Of Open Source SDN Controllers", *thenewstack*, 31-03-2015. [En línea]. Disponible en: https://thenewstack.io/sdn-series-part-eight-comparison-of-open-source-sdn-controllers/
- [34] R. Izard, "How yo work with *Fast-Failover* groups", *Floodloght Controller tutorials for developers,* 22-08-2018. [En línea]. Disponible en: https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/7995427/How+to+Work+with+Fast-Failover+*OpenFlow*+Groups
- [35] "*Open vSwitch/OpenFlow* compatibilty", *Open vSwitch documentation*. Última consulta: 14/09/2018. [En línea]. Disponible en: http://docs.openv*switch*.org/en/latest/faq/*OpenFlow*/

10. ANEXOS

Anexo A

Como se ha observado en la realización del proyecto, se han realizado dos despliegues: uno con máquinas virtuales y el despliegue completo con equipos físicos en los extremos. Se habrá observado que en el primero se utilizó el sistema operativo *CentOS 7*, mientras que en el segundo se usó *Ubuntu 16.04*. Esto se debe a que la extensión de túneles GRE cifrados con IPsec solo existe en sistemas operativos *Ubuntu 12.04*, *Ubuntu 14.04* y *Ubuntu 16.04* y *Debian*. Además, este soporte fue eliminado en la versión 2.6 de *Open vSwitch* por lo que en el segundo despliegue se utilizó la versión 2.5.4, mientras que en el primer despliegue se utilizó la versión 2.8.2. Junto con la versión 2.8.2 de *Open vSwitch,* en el primer despliegue se utilizó la versión del protocolo *OpenFlow* 1.5. En el segundo despliegue, dado que se pasó a utilizar la versión de *Open vSwitch* 2.5.4, fue necesaria la utilización del protocolo *OpenFlow* 1.3, debido a falta de características. En la siguiente figura se pueden apreciar las compatibilidades de *OpenFlow* y *Open vSwitch*:

TABLA. A.1 COMPATIBILIDAD DE VERSIONES DE *OPEN VSWITCH* Y *OPENFLOW* [35].

Open vSwitch	OF1.0	OF1.1	OF1.2	OF1.3	OF1.4	OF1.5	OF1.6
1.9 and earlier	yes	—	—	—	—	—	—
1.10, 1.11	yes	—	(*)	(*)	—	—	—
2.0, 2.1	yes	(*)	(*)	(*)	—	—	—
2.2	yes	(*)	(*)	(*)	(%)	(*)	—
2.3, 2.4	yes	yes	yes	yes	(*)	(*)	—
2.5, 2.6, 2.7	yes	yes	yes	yes	(*)	(*)	(*)
2.8	yes	yes	yes	yes	yes	(*)	(*)

— Indica no compatibilidad, “yes” indica versiones compatibles, (*) indica falta de funcionalidades y (%) indica posible incompatibilidad, *bugs* o implementación poco segura.

Anexo B

En este anexo se detallarán las reglas añadidas a los *Open vSwitches* en cada despliegue:

- En el primer despliegue:

En este caso, la numeración de puertos es la siguiente:

TABLA B.1. TABLA DE NUMERACIÓN DE PUERTOS PARA EL PRIMER DESPLIEGUE.				
ID del puerto	1	2	3	4
Nombre del puerto	enp63s0.10	enp63s0.11	enp63s0.12	vnet0

- Reglas del *Open vSwitch* 1 (sólo tabla 0 y todas prioridad 1 menos la regla del controlador):

TABLA B.2. REGLAS AÑADIDAS AL *OPEN VSWITCH* 1 PARA EL PRIMER DESPLIEGUE.			
in_port = 1	dl_vlan = 10	dl_dst = MAC equipo 1	actions=output:4
in_port = 4	dl_vlan = 10	dl_dst = MAC equipo 2	actions=output:1
in_port = 2	dl_vlan = 11	dl_dst = MAC equipo 1	actions=output:4
in_port = 4	dl_vlan = 11	dl_dst = MAC equipo 2	actions=output:2
in_port = 3	dl_vlan = 12	dl_dst = MAC equipo 1	actions=output:4
in_port = 4	dl_vlan = 12	dl_dst = MAC equipo 2	actions=output:3

Nótese que es innecesario definir la tabla del *Open vSwitch* 2, ya que es idéntica a la del *Open vSwitch* 1. La única diferencia entre ellas serán las MAC destino, cuyo valor será el del interfaz de red de una máquina virtual u otra.

- En el segundo despliegue:

Ahora, la numeración de puertos es la siguiente:

TABLA B.3. TABLA DE NUMERACIÓN DE PUERTOS PARA EL DESPLIEGUE FINAL.					
Identificador del puerto	1	2	3	4	5
Nombre del puerto	enp63s0	ens4.101	ens4.201	ens4.301	tun0

- Reglas del *Open vSwitch* 1 (tablas 0,1 y 2 y todas prioridad 1 menos las reglas del controlador con prioridad 0):

TABLA B.4. REGLAS AÑADIDAS AL *OPEN VSWITCH* 1 PARA EL DESPLIEGUE FINAL.			
table = 0	dl_dst = MAC equipo 2	nw_dst = 10.1.1.2	goto_table:1
table = 0	dl_dst = MAC equipo 1	nw_dst = 10.1.1.1	goto_table:1
table = 1	dl_vlan=101		goto_table:2
table = 2	tcp_dst = 80		actions=output:1
table = 2	tcp_src = 80		actions=output:2
table = 2	tcp_dst = 21		actions=output:1
table = 2	tcp_src = 21		actions=output:5

- Reglas del *Open vSwitch* 2 (tablas 0,1 y 2 y todas prioridad 1 menos las reglas del controlador con prioridad 0):

TABLA B.5. REGLAS AÑADIDAS AL *OPEN VSWITCH* 2 PARA EL DESPLIEGUE FINAL.			
table = 0	dl_dst = MAC equipo 1	nw_dst = 10.1.1.1	goto_table:1
table = 0	dl_dst = MAC equipo 2	nw_dst = 10.1.1.2	goto_table:1
table = 1	dl_vlan=101		goto_table:2
table = 2	tcp_dst = 80		actions=output:2
table = 2	tcp_src = 80		actions=output:1
table = 2	tcp_dst = 21		actions=output:5
table = 2	tcp_src= 21		actions=output:1

Nótese que las reglas añadidas son para una solicitud HTTP en el puerto 80 y para una solicitud FTP en la VPN corporativa con identificador 101. Nótese también el procesamiento multitabla y la bidireccionalidad en la definición de las reglas en la tabla 0, como en las reglas de la tabla 2, en las que el puerto del servidor es el 80 en el caso de HTTP y 21 en el caso de FTP.

En caso de recibir tráfico de la VPN de Renfe, por ejemplo, con identificador 201, se añadiría una regla en la tabla 1 con *dl_vlan* = 201 con la instrucción de ir a la tabla 2, idéntica a la insertada para la VPN con identificador 101.

uc3m | Universidad Carlos III de Madrid

Grado en Ingeniería en Tecnologías de Telecomunicación

2017-2018

Trabajo Fin de Grado

"Desarrollo de solución SD-WAN basada en SDN"

Resumen en inglés

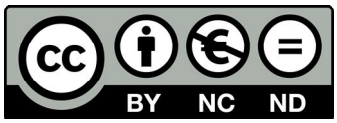
Álvaro Jiménez Moreno

Tutor/es

José Luis Iglesias Martínez

Antonio de la Oliva Delgado

Leganés, 2018



Summary.

Introduction.

Users and companies have changed their working methods related to networking in recent years.

A centralized model where big data providers where located on specific places in the network was changed with a distributed model where you can see providers anywhere in the network.

Nevertheless, internet statistics suggest that in ten years from now, there will be triple internet traffic, mainly multimedia traffic. This concerns most companies as they dont actually know if they might be able to support this traffic demand. Furthermore, monitoring and administration of all this network equipment that nowadays, are mainly managed in a distributed manner, makes it even more complex.

Software defined networking (SDN), has emerged as a new network architecture that increases programmability and centralizes the intelligence of the network, breaking the control plane out of the *switch* itself and delegating this control to a central equipment increasing the flexibility of the network.

Instead of having the same distributed scheme of legacy networks, the SDN architecture centralizes the management of the network infrastructure in one device, the SDN Controller. The main concepts of SDN such as will be reviewed throughout this work, as well as the architecture components including a brief description of its utility.

Moreover, SDN offers a real-time traffic monitoring and network supply which make network control quite easier and efficient. This monitoring and supply is also centralized in one point, so costs of maintaning ore ven repairing all the network hardware are substantially reduced.

The need of an overall framework which fulfilled a range of requirements (programmability, dynamic deployment and provisioning) while facilitating all new

services and applications dictated a new paradigm which should be capable to satisfy this requisites.

Research that delve into this topic have already started. The main problem is to give clients the same or a better quality of service, especially during the process of changing the network arquitecture to SDN.

Protocol description.

In this project, a virtualized SDN network outline will be also deployed using the *OpenFlow* Protocol. The *OpenFlow* protocol extracts the control plane into an *OpenFlow* controller, which determine the path of network packets across a network of *switches*. Its working method is simple. With the controller you can add rules into flow tables in order to match incoming traffic with those rules and forward it. The actual advantage of this is the variety of match fields that can be used to add rules (MAC addresses, IP addresses, IPv6 addresses, VLAN tag, MPLS tag, TCP port, UDP port, ARP fields, etc).

Multiple flow tables with multiple rules can be created and matches can be done through all of the flow tables, part of them or just one. When there is a match, an action previously assigned to the match is done. There are multiple actions permitted: output action (to specify a *switch*'s output port), push VLAN tag, push MPLS tag, pop VLAN tag, pop MPLS tag, set field action (to modify header fields in the packet), etc.

Counters are also available. With *OpenFlow* table, flow and port statistics can be visualized. There are also group and queue but they are not commonly used. Table counters show the number of active entries in the table, table searches done and number of matches. Flow counters manage the number of packets and bytes that matched that flow and flow's duration in the table (in seconds and nanoseconds). Finally, port counters display the number of packets and bytes that went through each port as well as the number of packets dropped, the number of errors (alignment, overflowing and CRC errors) transmitted and recieved and the number of collisions.

Project description.

As said, an *OpenFlow* based network outline will be deployed. Open v*Switch* will be used to virtualize *OpenFlow switches* and the Ryu controller and its API, which uses Python lenguage will be used to process packets sent to the controller and to add rules to the Open v*Switches*.

This deployment is done in order to test its functionality and capabilities to control ADIF'snetwork. It consists in an end to end network. An *OpenFlow* virtual *switch* as a layer 2 gateway for each end. Between each virtual *switch*, there will be two posible paths to route traffic, a normal layer 2 interface and a layer 2 IPsec tunnel interface to encrypt traffic. The non-IPsec path goes through the ADIF MPLS network which provides each own security requeriments so no encryption is needed.

The real network has to be able to manage three kinds of traffic: corporative traffic, Renfe traffic and security traffic. In order to simulate this, traffic is tagged before arriving to the virtual *switch*, using a physical Cisco Catalyst 2960 *switch* so that, depending on this VLAN, traffic will be treated differently.As there are three types of traffic, three VLANs will be used.

With this scheme, the virtual SDN network will be tested by giving HTTP service on ports 80 and 8080 and FTP/TFPT services, with two laptops connected at each end (one as client at the remote user end and the other one as server at the CPD end).

So, to provide an access control to the network, traffic will be sent through each path (MPLS or IPsec Internet) depending on the VLAN tag and the service the client asked for.

Moreover, a multi-table forwarding plane will be programmed using the controller. Ethernet and IPv4 headers will ne matched in the first table, VLAN tag in the sencond one and TCP or UDP ports in the last one. It is in this last table where traffic is sent to an output port to be forwarded.

Once the virtual network and its control plane is fully built, evaluation of the capabilities and advatiges and disadvantages of *OpenFlow* and SDN will be done.

As ADIF is working with other providers of SDN technollogy such as Cisco or Teldat, they want to compare all kind of solutions and value which solution fits better for their network topology and requirements, as well as its economic implications.

Project description.

The deployment of the outline will be divided in two parts:

- In the first part, before using the laptop deployment, virtual machines will be installed and linked to the Open v*Switch* by a tap interface. So an Open v*Switch* and a virtual machine will be installed in the same host. Virtual machines are simulating both ends of the network and will just generate tagged traffic using VLAN subinterfaces. This tagged traffic will be forwarded using rules added to the virtual *switches* using the controller. With this deployment, a brief contact wll be made with the protocol just and its working method, just before deploying the full outline. In this first deployment a Cisco Catalyst 2960 *switch* will be used to interconnect all hosts configuring three ports as trunk ports, to forward tagged traffic from the Open v*Switches* and to isolate controller traffic in another different VLAN.

- The second and final deployment will be done using the laptops mentioned in the introduction (one as client and the other one as server), giving HTTP, FTP and TFPT services between both of them.In this final deployment, two Cisco Catalyst 2960 *switches* will be used, one *switch* for each end. As there are three different VLANs, three access ports will be configured in each *switch* (one port per VLAN) as well as a trunk port (3 VLANs and controller traffic VLAN). One extra netword card will also be installed in each host with Open v*Switch* to makea direct connection between the virtual *switches*. With this configuration, incoming traffic from one of the laptops will go through both Open v*Switches* to the other end laptop without skipping any virtual *switch* forwarding process.

Evaluation.

After all deployments done, it could be said that *OpenFlow* seems to be a good option for small businesses which can't afford comercial solutions. It is also a good option for simple SDN desired network solutions. If a thorough Access control is needed, it is better to get solutions which make traffic engineering, traffic pilitícies, Quality of Service or link balance easier.

Refering to things seen in the outlines. TTL increases noticeably when packets are sent to the controller. When traffic is forwarded using *OpenFlow* rules inserted in the virtual *switches*, TTL doubles compared to normal Linux forwarding methods.

However, it has been corroborated SDN flexibility, as, for instance, changing the forwarding path is simply done by changing the output port in the code, or just adding rules to forward other kind of traffic (ICMP, VoIP, etc).

Conclusions.

After all tests and evaluations done, it could be said that SDN is a clear trend. Most companies desire to migrate their systems to a more flexible and controlled model which can support future traffic demands.

This, however, does not ensure SDN as the dominant networking model as manufacturers are offering new solutions at the same time with quite similar capabilities to SDN.

Nevertheless, there is no doubt that a network evolution is needed to support future demands, specially multimedia traffic. This evolution also has to provide the same or better quality of service and even support new exigences.

It can also be said that flexibility is one of the most important capabilities or the most important one. Data providers could be at any point of the network (lots of cloud and multicast services) so flexibility and ability to provide a real-time network control are the most attractive capabilities of SDN.

It is also notable the capability of provisioning and the real-time actions that can be done in case of network failure.

Acquired learning.

During the development of the project, deep concepts of SDN have been learnt. Its huge flexibility had been checked by deploying a network outline. Due to its innovative nature, acquiring these SDN knowledge is quite valuable nowadays.

Moreover, making this project in a Company like ADIF gives quite competences that an engineer should develop through years in addition to the working experience acquired.

Other knowledges related to communication networks, programming, Linux operating systems or hardware configuration have been also learnt