Trabajo
Final de Grado

Facultad de Ingeniería en
Telecomunicaciones

Tema:

SEGURIDAD EN REDES DEFINIDAS POR SOFTWARE

Andrés Ballesteros

L. Ezequiel Bujedo

Tutor: Enrique G. Banchio

Año 2016

Dedicatoria

A la paciencia y confianza en mí de mis padres.

L. Ezequiel Bujedo

Al apoyo que siempre me dieron mi familia y amigos

Andrés Ballesteros

Agradecimientos

A mi familia que siempre me dio su apoyo.

A mis compañeros de la facultad que me acompañaron todos estos años.

.

L. Ezequiel Bujedo

A mi familia por su apoyo y confianza
A mis amigos y compañeros por darme ayuda y fuerza

Andrés Ballesteros

Resumen

Las redes definidas por software son la evolución de las redes convencionales. Pueden aportar nuevas funcionalidades en las diferentes áreas donde se las pueden aplicar. Su seguridad es un aspecto muy importante pero poco estudiado hoy en día.

Existen muchas herramientas que sirven para probar su seguridad en forma análoga a la seguridad de redes convencionales. El objetivo de este trabajo, es explicar los aspectos a tener en cuenta sobre la seguridad de este nuevo tipo de redes y marcar la diferencia con sus predecesoras.

Andrés

Ezequiel

Glosario

SDN: Redes definidas por software o SDN (del inglés Software Defined Network)

ONF: La Open Network Foundation (ONF), una organización guiado por usuarios con el objetivo de fomentar el desarrollo de la SDN

API: Application Programming Interface. Interafz de aplicacion

DDoS:Distributed Denial of Service. Es un tipo de ataque informatico que causa que un servicio o recurso sea inaccesible para los usuarios

Framework: Infraestructura digital. Conjunto estandarizado de conceptos prácticas y criterios para enfocar una problematica.

LAN: Local Area Network. Red de area local

MPLS: Multi-Protocol Label Switching

MAC: Media Access Control

IP: Internet Protocol

VLAN: Virtual Local Area Network

ARP: Address Resolution Protocol

ISP: Internet Service Provider

WAN: Wide Area Network

NCS: Network Control Server

NCA: Network Control Application

BGP: Border Gateway Protocol

NIB: Network Information Base

RAP: Routing Application Proxy

TE: Traffic Engineering

OF: OpenFlow

LSP: Link State Packet

OFC: Open Flow Controller

OFA: OpenFlow Application

NVF: Network Function Virtualization

TLS: Transport Layer Security

PKI: Public Key Infrastructure

TCP: Transmission Control Protocol

EIGRP: Enhanced Interior Routing Protocol

IS-IS: Intermediate System to Intermediate System

OSPF: Open Shortest Path First

AH: Authentication Header

TTL: Time to Live

GTSM: Generalized TTL Security Mechanism

ACL: Access Control List

SNMP: Simple Network Management Protocol

SSH: Secure SHell

OOB: Out of Band

IPS: Intrusion Prevention System

IDS: Intrusion Detection System

SIEM: Security information Event Manager

OpenVAS: Open Vulnerability Assessment System

ICMP: Internet Control Message Protocol

DCI: Data Center Interconnect

NVGRE: Network Virtualization using Generic Routing Encapsulation

STT: Stateless Transport Tunneling

SO: Sistema Operativo

Tabla de contenidos

Índice

Índice de Tablas

Índice de Imágenes

Índice de Figuras

Capítulo 1

Introducción

1.1 Definición de redes definidas por software

Redes definidas por software o SDN (del inglés Software Defined Network) es un conjunto de técnicas relacionadas con el área de redes computacionales, cuyo objetivo es facilitar la implementación e implantación de servicios de red de una manera determinista, dinámica y escalable, evitando al administrador de red gestionar dichos servicios a bajo nivel. Todo esto se consigue mediante la separación del plano de control (*software*) del plano de datos (*hardware*).

La Open Network Foundation (ONF), una organización guiada por usuarios con el objetivo de fomentar el desarrollo de la SDN, lo describe como:

> "La separación física del plano de control de la red del plano de reenvío, y donde el plano de control controla varios dispositivos"

El término *Programabilidad* hace referencia a la flexibilidad del plano de control, que es directamente programable en software e incluye un canal de control para intercambiar datos con los dispositivos de red. Una interfaz de aplicación (API) permite a las aplicaciones comunicarse con el plano de control para poder acceder y modificar los dispositivos de red. La consecuencia de este nuevo principio de diseño incrementa la flexibilidad en la gestión de la red. Por analogía a los teléfonos móviles, esta evolución puede ser comparada con el cambio de los teléfonos con funciones restringidas a la flexibilidad de los smartphones con un sistema operativo y la habilidad de instalar varias aplicaciones. El concepto de las redes tradicionales de computadora fue basado en dispositivos que contenían ambos planos, el de control y el de datos. Para distinguir estas redes del paradigma SDN, nos vamos a referir a ellas como redes convencionales.

1.2 Problemática

El concepto de un plano de control centralizado, junto con el canal de control que se utiliza para intercambiar información con los dispositivos de red, introduce nuevos temas de seguridad que necesitan ser caracterizado. Desde la perspectiva de un atacante, el controlador de red es atractivo debido a su importante papel, por lo que requiere de mecanismos específicos de protección. Este es un ejemplo de un caso en el que las soluciones de seguridad de red son más un aspecto de la aplicación y menos dependiente de soluciones de hardware especializados. El alcance de estos conceptos, y

escenarios concretos de aplicación, aún carecen de una comprensión completa en la comunidad de investigación

La seguridad, como en todo tipo de redes, es un aspecto crucial en redes SDN, para proteger la disponibilidad, integridad y privacidad de la información que transporta. La seguridad en estas redes, aunque existen enfoques competentes, todavía está en definición, ya que estos enfoques no convergen en una idea común. A pesar de estas diferencias, es claro, que las soluciones deben crear un entorno escalable, eficiente y seguro. Además, la seguridad debe ser simple de configurar (debido al dinamismo de la red) y efectiva (para asegurar que pueda desplegarse en cualquier parte). En la arquitectura, hay varios elementos que deben ser protegidos y acciones a llevar a cabo:

- **Asegurar y proteger el controlador.** Debido a que es el centro de control y decisión de la red, debe ser cuidadosamente controlado. Además, si se produce por ejemplo un ataque DDoS al controlador y cae víctima de este, la red cae con él, por lo que debe estar protegido con los mecanismos de seguridad necesarios para afianzar su disponibilidad continua.
- **Privacidad e integridad.** La protección de las comunicaciones en la red es crítica, por lo que se debe asegurar el controlador, las aplicaciones que carga y los dispositivos de red que gestiona y autenticar, estableciendo unos mecanismos que permitan que son quien dicen ser y además funcionan correctamente.
- **Crear un *framework* de política robusto.** Es necesario estar seguro de que el controlador (y demás dispositivos) están haciendo lo que queremos que hagan, por lo que habrá que realizar comprobaciones continuas de los sistemas en general.
- **Llevar a cabo análisis forenses de la red.** Para poder determinar, en caso de un ataque, quién lo realizó, reportarlo y proteger la red de cara al futuro

1.3 Principales Objetivos

- Estudiar la tecnología de redes definidas por software, sus componentes, protocolos y funcionamiento.
- Asimilar conceptos sobre la configuración de los equipos y controladores.
- Realizar análisis de seguridad de dicha tecnología, puntos críticos, vulnerabilidades, y posibles ataques.
- Realizar una selección del controlador más apto para dicha tarea teniendo en cuenta el factor seguridad.
- Definir a partir de lo investigado la opción de configuración de controlador más segura

Capítulo 2

OpenFlow

2.1 Resumen Redes definidas por software basadas en openflow

Esta tesis se centra en las SDN basadas en OpenFlow. Esto parece razonable, ya que la investigación más actual y el software de código abierto se centra alrededor de este protocolo. Como resultado, está siendo cada vez más desplegado en entornos de producción. Por ejemplo, Google ha utilizado el concepto de SDN basados en OpenFlow para su red troncal interna llamada G-Scale, que es una red internacional muy importante. La experiencia de Google fue muy prometedora, permitiendo un rico despliegue de características y simplificación de manejo de la red. En la industria, la mayoría de los proveedores de equipos de red proporcionan dispositivos con soporte OpenFlow, por ejemplo, Cisco, Hewlett Packard , Juniper, y NEC. La arquitectura de SDN y OpenFlow se muestra en la Figura[21] y consta de tres capas.

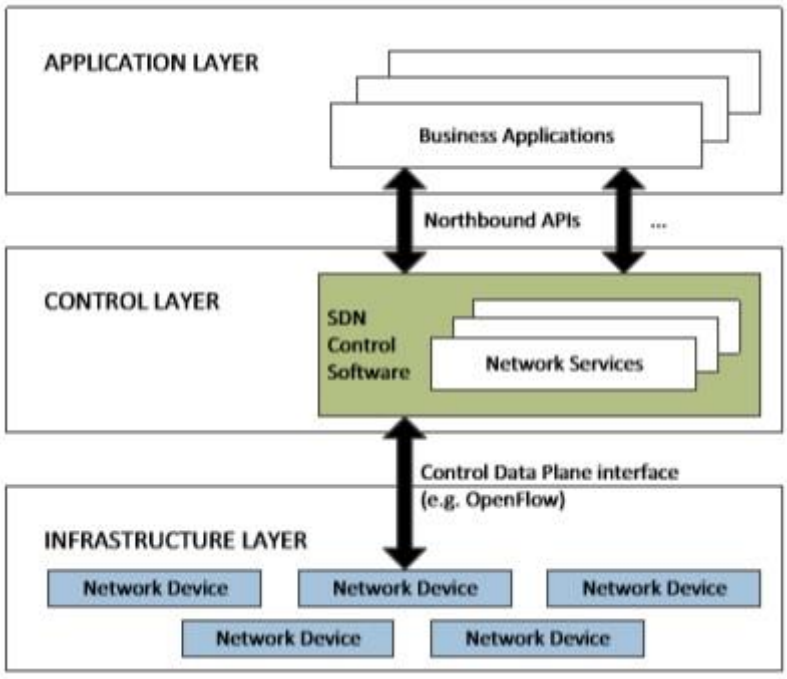


Figura 21 Arquitectura Openflow

2.1.1 Capa de Infraestructura

Esta capa se utiliza para el reenvío de paquetes basado en dispositivos de red que soportan el protocolo Openflow. Hardware especializado, incluyendo la memoria para almacenar información relacionada al flujo, se

utiliza para este propósito, lo que permite el procesamiento rápido de paquetes.

Dispositivos de red comunes que pertenecen a esta capa son switches y routers. Un switch normalmente funciona en la capa de enlace (L2). Dado que la funcionalidad de un SDN basados en OpenFlow es definido por un controlador de red por software, nos referimos a todos los dispositivos de red como switches.

2.1.2 Capa de control

El controlador de red, a menudo referido como el sistema operativo de red, que comprende el plano de control y tiene una vista centralizada de la red. Esto puede ser aprovechado para tomar decisiones de reenvío que son basadas en el estado de todos los dispositivos de red conectados. El controlador de red se comunica con los dispositivos de red a través de la interfaz *southband*de openflow. Un segundo tipo de API, la interfaz *northbound* es usada para intercambiar datos entre los servicios de red y las aplicaciones de negocios. Un ejemplo típico de servicios de red incluye enrutamiento de paquetes capa 2 y MPLS. Además, múltiples controladores de red se pueden implementar en una red para diferentes propósitos, por ejemplo para evitar un nodo que está fallando o para separar tareas.

Inspirado por el primer controlador open-source, NOX, escrito en C++, institutos de investigación y compañías desarrollaron controladores de red para una larga gama de propósitos. En la comunidad open-source, los controladores típicos son por ejemplo POX (Python), Beacon (Java) y Floodlight (Java). Este último es soportado por la compañía Big Switch Network, que comercializa también un controlador comercial de red, el controlador Big Switch

2.1.3 Capa de aplicación

La capa de aplicación permite a las businessapplications modificar e influenciar la forma en que la red se comporta para proveer servicios a los clientes. Esto requiere de la definición de una API, para permitir a desarrolladores terceros que desarrollen y vendan aplicaciones a los operadores de red. El desarrollo de esa API todavía no fue definida por la ONF, pero es requerida para garantizar la interoperabilidad entre businessapplications y los controladores de red de diferentes proveedores. Frenetic es un lenguaje de programación de red que puede ser usado para definir políticas de alto nivel. Este provee una abstracción de alto nivel de las funciones de la red para ocultar el procesamiento de paquetes a bajo nivel al programador. El proyecto propuso recientemente un nuevo lenguaje basado en Python llamado Pyretic. Este provee una abstracción del modelo de paquete y una álgebra de alto nivel de políticas y objetos de red.

2.2 Protocolo OpenFlow

El interés en las SDN ha aumentado considerablemente desde el desarrollo del protocolo OpenFlow, actualmente en la versión 1.4.0 y gestionado por la ONF [ ]. Mientras que la funcionalidad principal se implementó por completo en la versión 1.0, los desarrollos más recientes incluyen soporte para IPv6 y MPLS.

Los principios básicos del protocolo OpenFlow para el reenvío de paquetes se explican a continuación.

2.2.1 Flujos de red

El protocolo OpenFlow se basa en el concepto de la red de flujos para vigilar y controlar el tráfico dentro de una red. Los flujos de red mejoran la escalabilidad mediante el aumento de las velocidades de red y en la actualidad son compatibles con los dispositivos flow-enabled de los principales vendedores. Hay varias definiciones del término *flujo* en la literatura. El IP FlowInformationeXport (IPFIX grupo de trabajo dentro de la IETF [34, 35] describe un flujo de esta manera:

"Un flujo se define como un conjunto de paquetes IP que pasan a un punto de observación en la red durante un intervalo de tiempo determinado. Todos los paquetes que pertenecen a un determinado flujo tienen un conjunto de propiedades comunes. "

Por ejemplo, una comunicación entre dos entidades resulta en dos flujos, como se muestra en la Figura [22]. Las propiedades típicas de un flujo incluye dirección IP origen y destino y también puerto de origen y destino, y el tipo de protocolo (TCP, UDP, ICMP,...).

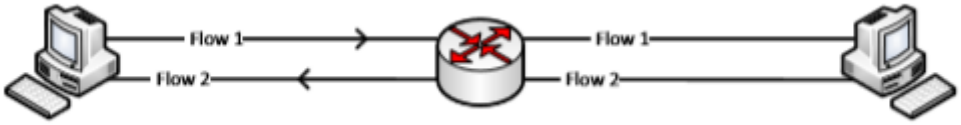


Figura 22 comunicación entre maquinas con Openflow

2.2.2 Tabla de flujo

Con el fin de enviar paquetes basados en el concepto de flujos, cada switch OpenFlow contiene una o varias tablas de flujo que se utilizan para almacenar entradas / reglas de flujo pre definidas. Por ejemplo, la dirección MAC se puede utilizar para identificar hosts en una red con switches como se muestra en la Figura 23. Cada entrada de flujo contiene los siguientes componentes:

Match Fields | Priority | Counters | Instructions | Timeouts | Cookie

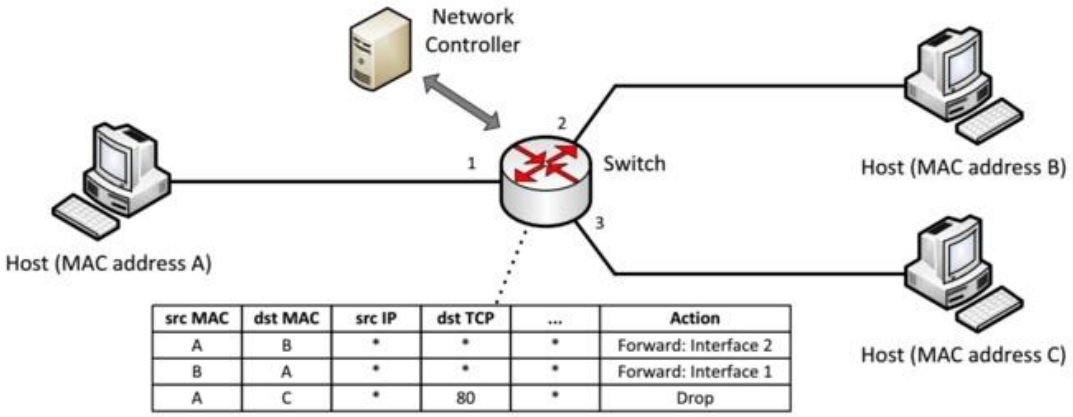


Figura 23 direccionamiento por dirección MAC

Para todas los paquetes entrantes que llegan a un switch, se realiza una búsqueda en la tabla de flujo haciendo coincidir la cabecera del paquete con el *Match fields* de todas las entradas. Cada switch OpenFlow soporta el emparejamiento de todos los campos mostrados en la tabla [22] que también contiene un valor específico o wildcard

Match Fields
Ingress Port
Ethernet source address
Ethernet destination address
Ethernet type
IPv4 or IPv6 protocol number
IPv4 source address
IPv4 destination address
IPv6 source address
IPv6 destination address
TCP source port
TCP destination port
UDP source port
UDP destination port

Tabla 22 Match Fields

En dependencia de la versión de OpenFlow y los dispositivos de red, deben estar disponibles campos adicionales, por ejemplo para VLANs, MPLS o ARP. Flujos de entrada que contienen *campos dewildcards*también son conocidos *macroflujos*. Este concepto permite el rango de paquetes que pueden ser emparejados a un flujo en particular
a ser definido. Si un paquete puede ser emparejado a varios flujos de entrada, se elige la entrada con la coincidencia más precisa (prioridad).

Mientras se busca una coincidencia, se aplican instrucciones asociadas al paquete. Acciones típicas son forward, drop, modify y enqueue. Mientras la última de estas también va a reenviar el paquete, es puesto en una queue donde el comportamiento de reenvío puede ser configurado, por ejemplo a una

cuenta de requerimientos de calidad de servicios. Además de especificar las acciones de reenvío de paquetes, las instrucciones también puede modificar la cabecera del paquete (por ejemplo push cabecera MPLS), si la función es soportada por los dispositivos de red.

El tiempo de vida de una entrada de flujo en una tabla es especificado por el tiempo de expiración de flujo (Timeouts*)*, que es definido por un tiempo de espera libre y un tiempo de espera fijo. El tiempo de espera libre indica el tiempo después de que una entrada de flujo fue removida cuando no hubo coincidencias de paquetes, mientras que el tiempo de espera fijo define cuando una entrada de flujo va a ser removida independientemente del número de paquetes que hayan coincidido.

Las estadísticas respectivas, que son definidas en el campo *Counters,* contienen el número de paquetes/bytes recibidos, la duración del flujo (segundos y nanosegundos), y la razón por la que fue removido. Además de los contadores que son definidos por flujo, también hay contadores por tabla de flujo (por ejemplo número de entradas activas), por puerto y fila. El uso de campo de *Cookie* no está definido, pero puede ser usado por el controlador para varios propósitos, por ejemplo como una bandera o identificador.

Las redes basadas en OpenFLow pueden ser clasificadas por cómo operan, en OpenFlow *nativo* o en *modo híbrido*. Como un dispositivo de red puede ser considerado como una combinación de dos switches en uno, e incrementa la flexibilidad y la posibilidad de crear diferentes escenarios para los operadores de red

El concepto de un plano de control centralizado, junto con el canal de control que se utiliza para intercambiar información con los dispositivos de red, introduce nuevos temas de seguridad que necesitan ser caracterizado. Desde la perspectiva de un atacante, el controlador de red es atractivo debido a su importante papel, por lo que requiere de mecanismos específicos de protección. Este es un ejemplo de un caso en el que las soluciones de seguridad de red son más un aspecto de la aplicación y menos dependiente de soluciones de hardware especializados. El alcance de estos conceptos, y escenarios concretos de aplicación, aún carecen de una comprensión completa en la comunidad de investigación El concepto de un plano de control centralizado, junto con el canal de control que se utiliza para intercambiar información con los dispositivos de red, introduce nuevos temas de seguridad que necesitan ser caracterizado. Desde la perspectiva de un atacante, el controlador de red es atractivo debido a su importante papel, por lo que requiere de mecanismos específicos de protección. Este es un ejemplo de un caso en el que las soluciones de seguridad de red son más un aspecto de la aplicación y menos dependiente de soluciones de hardware especializados. El alcance de estos conceptos, y escenarios concretos de aplicación, aún carecen de una comprensión completa en la comunidad de investigación .El concepto de un plano de control centralizado, junto con el canal de control que se utiliza para intercambiar información con los dispositivos de red, introduce nuevos temas de seguridad que necesitan ser caracterizado. Desde la perspectiva de un atacante, el controlador de red es atractivo debido a su importante papel, por lo que requiere de mecanismos específicos de protección. Este es un ejemplo de un caso en el que las soluciones de seguridad de red son más un aspecto de la aplicación y menos dependiente de soluciones de hardware especializados. El alcance de estos conceptos, y escenarios concretos de

aplicación, aún carecen de una comprensión completa en la comunidad de investigación

Capítulo 3

Controladores

3.1 Resumen de controladores

3.1.1 NOX/POX:

Figura 31 NOX

NOX es un controlador de primera generación, de hecho su primera versión fue el primer controlador de OpenFlow. NOX es de código abierto, es estable y ampliamente usado.

Este controlador en su primera versión o NOX clásico se escribió en C++ y Python, pero este ya no se usa, mientras que su versión más nueva usa solo C++, es más rápido, todavía recibe actualizaciones y lo soportan varios equipos. Como se dijo el usuario programa de mismo solo en C++, y el mismo se puede usar en OpenFlow en sus versiones 1.0 a 1.3.

El funcionamiento de este controlador como el de la mayoría trabaja monitoreando eventos y se programa una serie de tareas a realizar ante cada evento. NOX es recomendado para quienes ya programan C++ y quieren usar instrucciones no muy complejas, además por su simplicidad logra buenos resultados en cuanto a velocidad.

Figura 32POX

POX es básicamente una versión de NOX en Python, con la desventaja de que solo soporta OpenFlow versión 1.0 y al ser Python su performance no es tan rápida como el C++ de NOX. POX es ampliamente usado y actualizado, además es relativamente fácil de leer y escribir el código del mismo, por lo que se recomienda si se sabe programar en Python o, dada su simplicidad, se quiere aprender a hacerlo, siempre y cuando la performance del mismo no sea

crítica. También permite una rápida programación, por lo que es muy usado en demostraciones, experimentación e investigación.

3.1.2 Ryu:

Figura 33RYU

Ruy es un controlador Open Source basado en Python, este soporta las versiones 1.0, 1.2, 1.3 y 1.4, trabaja con extensiones Nicira 1.5 y también con OpenStack.

Sus principales ventajas es que integra OpenStack con OpenFlow 1.2, 1.3 y 1.4. Otra de sus ventajas es que todo el código está disponible gratuitamente bajo la licencia de Apache 2.0

Una de sus desventajas es la baja Performance que tiene.

3.1.3 Floodlight

Figura 34FLOODLIGHT

Floodlight es un controlador Open Source basado en Java que Soporta Openflow 1.0 y Es apoyado por una comunidad de desarrolladores incluidos Ingenieros de Big Switch Networks

Ventajas:

- Buena documentación
- Integración con REST API
- Nivel de producción , OpenStack/Nubes Multi clientes

La desventaja de estos controladores está en su complejidad, y en la dificultad a la hora de aprender a usarlos.

3.1.4 OpenDaylight

Figura 35OPENDAYLIGHT

OpenDaylight Project es un proyecto de código abierto financiado por la fundación Linux, la misma busca acelerar el proceso de aceptación de las redes SDN con una plataforma común a todas las redes y que sea robusta. Este software está escrito en JAVA.

Este controlador que ya está por su tercera versión (Hydrogen, Helium y Lithium en ese orden) tiene la ventaja de haber sido ampliamente aceptado por las industrias, además de integrar OpenStack y Cloud Applications.

La desventaja de estos controladores está en su complejidad, y en la dificultad a la hora de aprender a usarlos.

Este controlador es recomendado para aquellos que conocen JAVA, necesitan una buena performance y soporte. También permite utilizar funciones modulares y soporta OpenStack y Cloud Applications.

3.2 Comparación entre los controladores

Veamos una comparación entre los distintos controladores

	NOX	POX	Ryu	Floodlight
Lenguaje	C++	Python	Python	Java
Performance	Rapida	Lenta	Lenta	Rapida
OpenFlow	1.0 (1.1, 1.2, 1.3)	1.00	1.0, 1.1, 1.3	1.00
OpenStack	No	No	Si	Si
Curva de aprendizaje	Moderado	Facil	Moderado	Dificil

Tabla 32 Comparación entre controladores

Existen otros controladores como por ejemplo Pyretic, Frenetic, Procera, RouteFlow, Trema, etc. los mismos no serán desarrollados en esta tesis por su poca utilización.

Capítulo 4

SDN hoy y su seguridad

4.1 SDN HOY

4.1.1 Google B4

Google opera una red enorme, es tan grande que en 2010 un estudio hecho por Arbor Networks concluyó que “si google fuera un ISP, sería el de más alto crecimiento y la tercera prestadora de servicio más grande del mundo. Solo dos proveedores (quienes transportan una gran cantidad de información de google) aportan mayor tráfico.”

En la Open Networking Summit, el ingeniero de google AminVahdat presentó "SDN@Google: Porque y como." ("SDN@Google: Why and How.”). En esta charla compartió cómo google usa una combinación de OpenFlow y software Quagga para optimizar la interconexión de sus centros de datos. También compartió algunos detalles del uso de OpenFlow dentro de sus centros de datos, en lo que llamaron Red SDN “B4”.

Vahdat explicó que el crecimiento del tráfico de algunas zonas está superando la capacidad de la red, y este crecimiento es muy caro, porque la red no escala económicamente como lo hacen las computadoras de almacenamiento. El costo de operación por unidad de estos últimos decrece a medida que aumenta la escala, no así con la red.

Primero, separando el Software del Hardware la compañía puede elegir Hardware basado en parámetros requeridos, mientras es capaz de innovar e implementar con el software. Luego, provee lógica de control centralizada que será más determinante, más eficiente y que tolere mejor las fallas. Por último, la automatización permite a Google separar el monitoreo, operación y manejo de la red.

Al comienzo de este proyecto, Google construyó sus propios Switch, esto se debió a que ningún Hardware del mercado cumplía con sus necesidades al momento de iniciar el proyecto.

Imagen 41 Switches Google B4

Las WAN modernas son críticas en la performance de internet, pero en las mismas la pérdida de paquetes es considerada aceptable, sin distinguir entre las distintas aplicaciones y lo crítico que puede ser el tráfico de algunas de ellas. Dado que los enlaces en redes WAN se utiliza normalmente al 30-40% cualquier pérdida de enlace o fallo de equipos es imperceptible al usuario. Este sobre aprovisionamiento vuelve a la red muy confiable al costo de 2 a 3 veces el ancho de banda requerido y equipos de enrutamiento de muy alto nivel.

Lo que más motivó el cambio a SDN fue poder tener una red hecha a medida para las necesidades de Google, centrando el diseño en el hecho de que las fallas son inevitables y deberían notificarse a las aplicaciones finales y hardware que exporte a interfaces simples para programar tablas de enrutamiento desde un control central.

B4 ya lleva más de 3 años de implementado, y algunas de sus características únicas son sus grandes anchos de banda para pocos equipos, demandas de tráfico muy elásticas y completo control sobre los servidores de frontera, lo que le permite medir y limitar la demanda. La ingeniería de tráfico centralizada permite un uso de los enlaces cercano al 100% y un promedio de uso del 70%. En este tiempo la red ya creció más que las otras, permitiendo cumplir las demandas de ancho de banda de las distintas aplicaciones de manera más eficiente de lo que hubiera sido posible.

La arquitectura lógica de esta red SDN puede verse en tres capas como muestra la figura. B4 sirve a múltiples sitios WAN, cada uno con un clúster de servidores. Dentro de cada switch se dirige el tráfico sin realizar ningún control complejo, y la capa de control consiste de servidores de control de red (Network Control Servers, NCS), que contienen controladores OpenFlow (OFC) y aplicaciones de control de red (NCAs). Estas características permiten ruteo distribuido y control de tráfico centralizado.

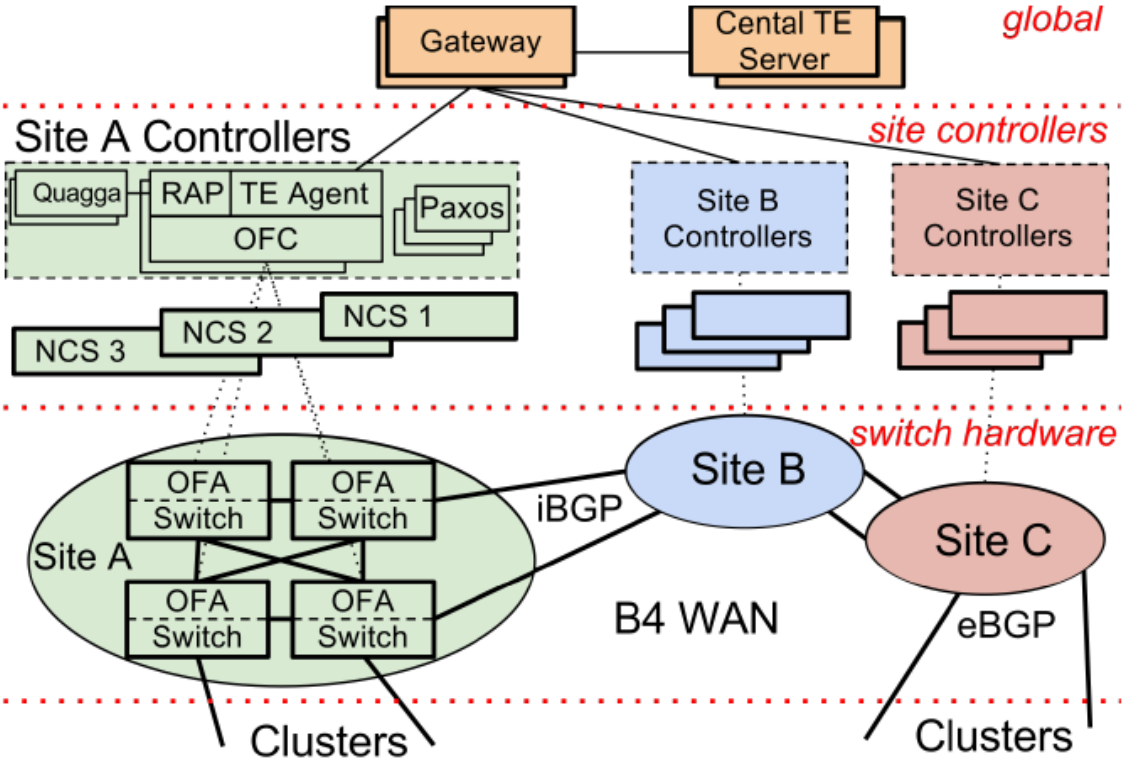


Figura 41 Google B4

La capa global consiste de aplicaciones de lógica centralizada, que permiten el control centralizado de la red a través de los NCA a nivel de los sitios. Cada clúster en la red es lógicamente autónomo, con un set de IP y routers BGP.

Si bien los equipos de enrutamiento de áreas grandes convencionales exigen grandes tablas de enrutamiento, grandes buffer y un hardware que lo soporte, se postuló que con un mejor manejo se podrán ajustar las transmisiones para evitar buffer tan grandes y evitar pérdidas de paquetes, además corren sobre un número limitado de data centers, por lo que las tablas de ruteo tampoco son tan grandes. También se vio que la mayoría de las fallas en los switch se originan en el software y no el hardware, por lo que al separar los mismos es más fácil controlar y corregir errores.

El controlador de la red es una versión modificada de Onix, con un agregado llamado NIB (Network Information Base) que contiene el estado actual de la red con respecto a topología, configuración y estado de los links.

Algunos de los mayores retos en esta red fue integrar el control basado en OpenFlow con el protocolo de ruteo existente. Para trabajar en las funciones de OpenFlow se decidió usar el software abierto Quagga para los BGP/ISIS del NCS (Network Control Servers). Se escribió una aplicación Proxy de ruteo (RoutingApplication Proxy, RAP) como una aplicación SDN, para permitir la conectividad entre Quagga y OpenFlow.

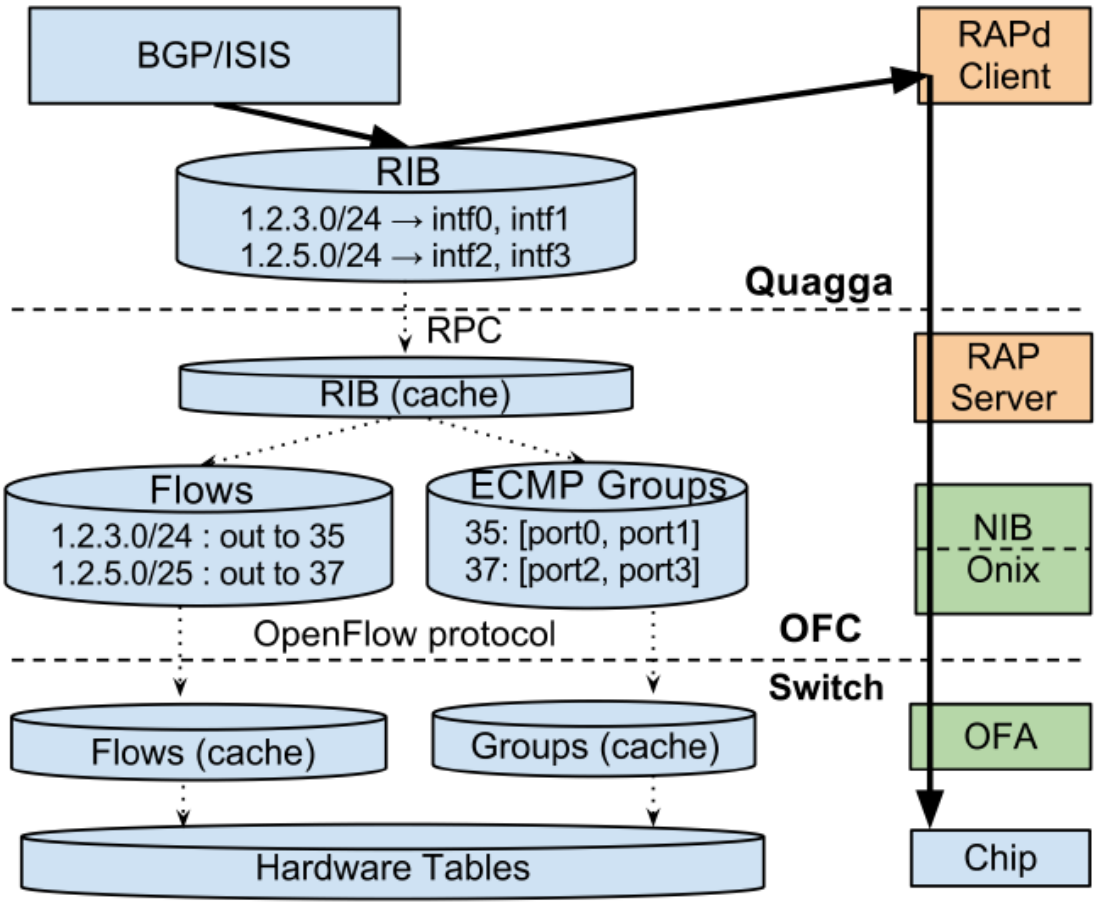


Figura 42 Google B4

En cuanto a ingeniería de tráfico, B4 apunto a poder compartir ancho de banda entre aplicaciones, estableciendo un mínimo y un máximo para cada una, tratando de no mejorar una en deterioro de las otras. Veamos un diagrama de flujo de su funcionamiento:

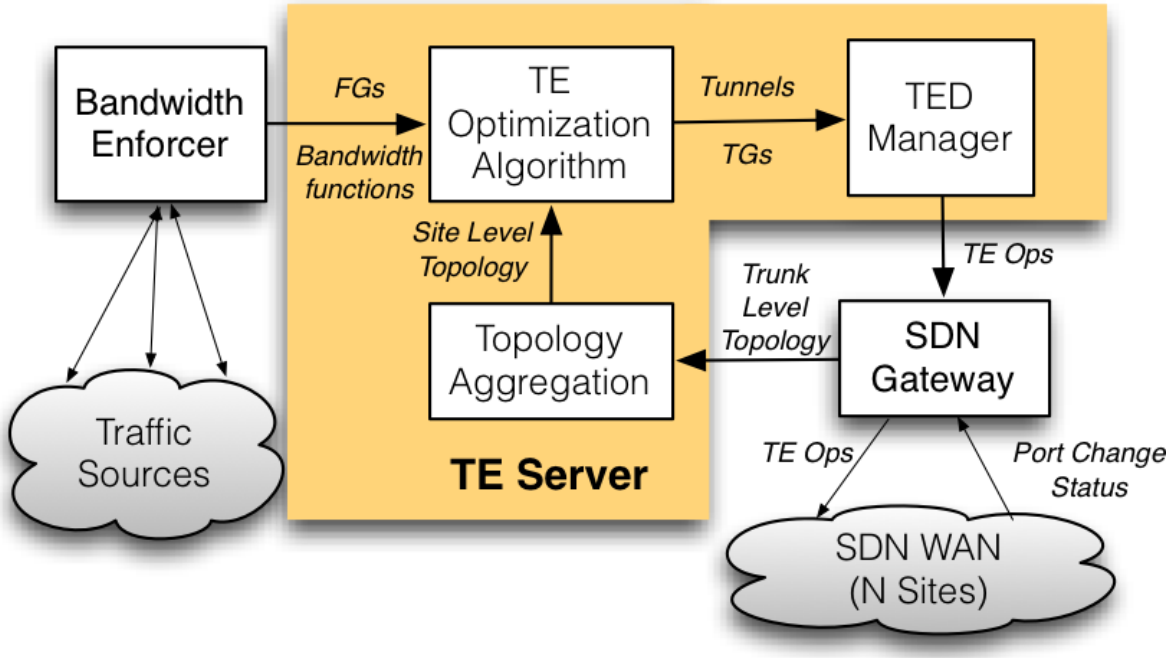


Figura 43 Google B4

El servidor TE (Trafficengineering) genera túneles y grupos de flujo para la Gateway SDN. Esta los redirige hacia el controlador que a su vez los instala en los switch. Para asociar un ancho de banda a cada aplicación se le asocia una función de ancho de banda, un tipo de contrato entre la aplicación y B4. Esta función indica a la aplicación el ancho de banda del que dispone basado en una prioridad relativa, lo que una llamaría darle lo justo y necesario. Este límite en ancho de banda lo impone el *BandWidthEnforcer*quien también calcula los anchos de banda en los límites de la red.

La red también trabaja con un algoritmo de optimización, el mismo consta de dos partes, la primera es la generación de grupos de túneles, encargada de ubicar los anchos de banda en grupos de flujos usando la función de ancho de banda para priorizar en los cuellos de botella y la segunda es la cuantificación de grupos de túneles encargado de traducir las grandes exigencias de tráfico de un grupo de túneles para que los equipos y la red puedan soportarlos.

Esta configuración de grupos de, túneles y grupos de flujo se traduce en estados de OpenFlow. Cada switch OF cumple tres roles, el primero es encapsular, iniciar los túneles y dividir el tráfico entre los mismos, el segundo es re direccionar paquetes basado en su cabecera y el tercero es des encapsular, cerrar los túneles y redirigir los paquetes usando reglas de enrutamiento regulares.

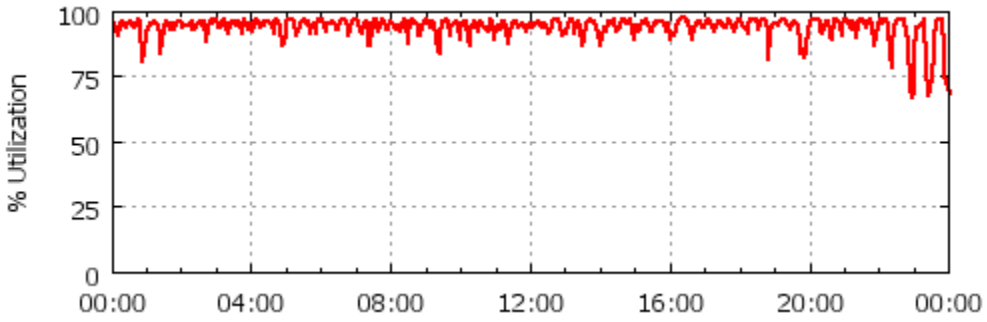
En el despliegue de esta red se colocaron 5 servidores TE geográficamente distribuidos, los TE secundarios permanecen en Standby y pueden asumir el control en menos de 10 segundos, mientras que el que está en control es estable y mantiene su estatus por al menos 11 días.

También se generaron errores y se midió ante tiempo de recuperación ante dichos eventos:

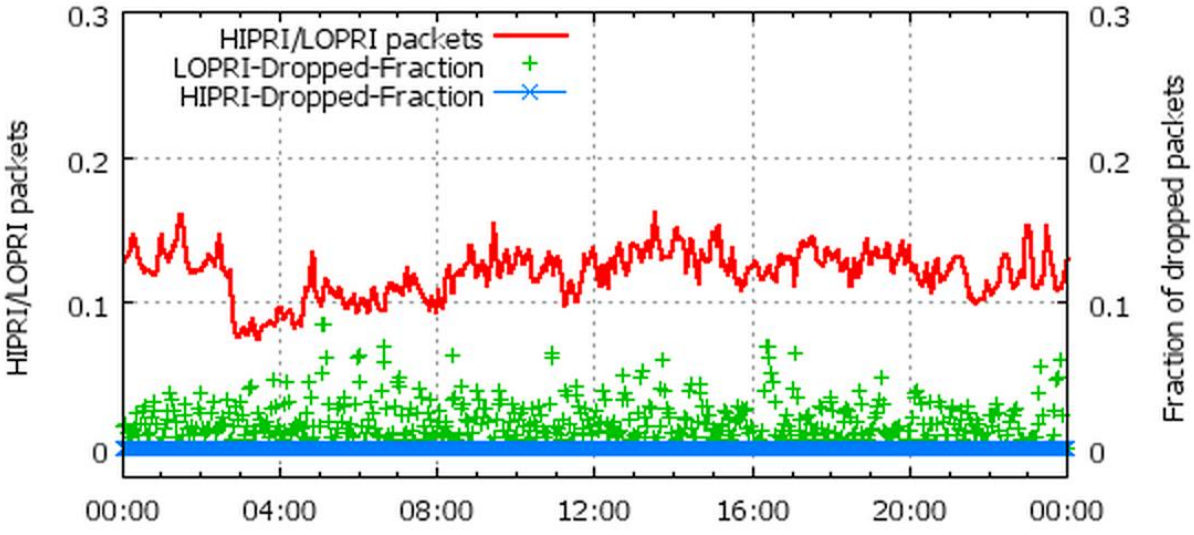
Failure Type	Packet Loss (ms)
Single link	4
Encap switch	10
Transit switch neighboring an encap switch	3300
OFC	0
TE Server	0
TE Disable/Enable	0

Tabla 41 Tiempode recuperación

Como vimos anteriormente este sistema lleva el uso de los enlaces a casi un 100%, esto, esto no es deseado en enlaces WAN, pero aquí se permite debido a la diferenciación entre clases de tráfico. Veamos un gráfico de la utilización de los enlaces entre dos sitios WAN:



Este sistema diferencia entre paquetes de alta prioridad y de baja, evitando la pérdida de los primeros:



Aquí podemos ver el tráfico de paquetes de alta prioridad (ROJO), los paquetes de baja prioridad (Verde) y la cantidad de paquetes de alta prioridad dropeados (Azul).

Una de las desventajas que se encontró fue al hacer un mantenimiento rutinario, en el que, por accidente, se le cambio a un switch el ID por uno duplicado, automáticamente el controlador al recibir notificaciones de

adyacencias de dos switch con el mismo ID comenzó a inundar la red de paquetes LSP (Link StatePacket), esto llevó a solapamiento de links y duplicidad de información. A su vez los switch duplicados al recibir LSP con la topología del otro switch responden al controlador con la propia, causando así mayor procesamiento. También se encontraron problemas en el volumen de paquetes entre OFC y OFA, llegando a colas de 400MB.

Ambas fallas mencionadas se intentaron solucionar reiniciando el OFC, pero un error del mismo evita la recuperación con alta carga, por lo que fue necesario desconectar él TE y vaciar el OFC para poder reiniciarlo.

La conclusión de google al cabo de 3 años de trabajo con este sistema fue que a pesar de las fallas por ser la primera implementación de este tipo B4 mueve más tráfico que la red clásica que funciona en paralelo, permitiendo un mejor uso de ancho de banda. De todas maneras B4 no soluciona todos los problemas, todavía existen cuellos de botella que requieren trabajo. También se destaca este trabajo por su implementación híbrida con redes comunes, viendo a futuro la introducción gradual de SDN en redes preexistente.

4.1.2 Google Andrómeda

Google acaba de aplicar la última tecnología en networking que maneja los servicios internos de su *Cloud PlatformUsers* de todo el mundo. Andrómeda ahora se encarga de 2 *EngineZones*: us-central1-b y europe-west1-a. Los clientes en estas zonas van a notar automáticamente una mejor performance. Luego se va a ir migrando completamente todas las zonas a Andrómeda. La siguiente imagen es de la presentación que hizo google en el Open Network Summit y hace referencia a la arquitectura de alto nivel de Andrómeda

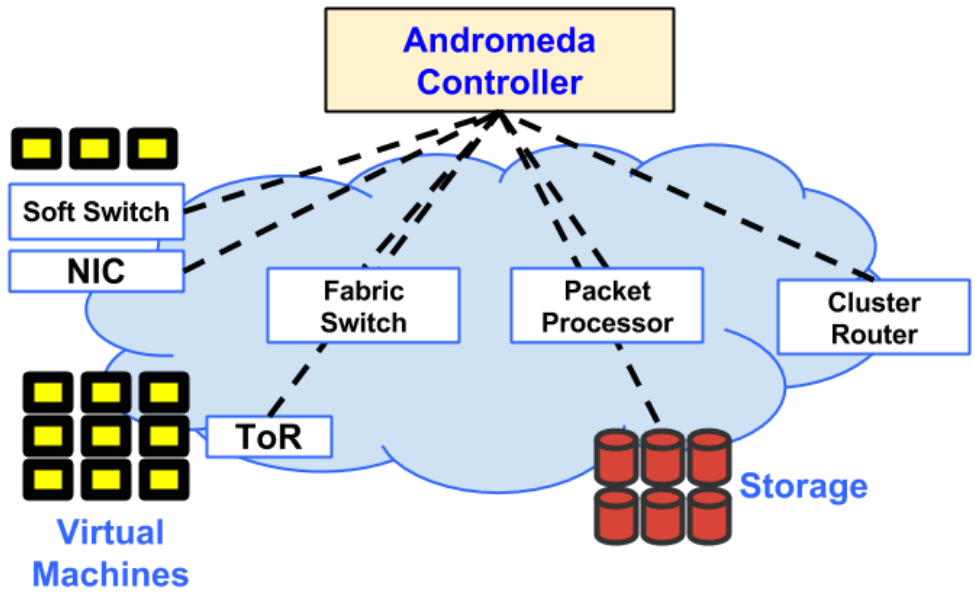


Figura 44 Google Andrómeda

El objetivo de Andrómeda es mostrar el rendimiento bruto de la red subyacente mientras se aplica la Network FunctionVirtualization(NVF). Demostrando el mismo procesamiento dentro de la red que permite escalar los servicios internos mientras se mantiene extensible y aislada del usuario final. **Esta funcionalidad incluye protección a denegación de servicios distribuida (DDoS)**, balance de tráfico transparente, **lista de control de acceso, y firewalls.**

Si bien Andrómeda no es un producto final de plataforma de red virtual, es la base para proveer servicios de redes virtuales con alta performance, disponibilidad, aislamiento y seguridad. Por ejemplo la plataforma virtual de **firewalls,** routing y reglas de reenvío gracias a la capa de aplicaciones interna de Andrómeda y su infraestructura

4.1.3 SDN en menor escala: Redes Hogareñas

Con la implementación de *BandwidthCaps* (Limitación de ancho de banda) por parte de algunos ISP de Estados Unidos y algunos lugares más del mundo surgió el problema del control de datos por el usuario final. Con el uso de los routers/switchs hogareños más comercializados, su gestión se vuelve confusa ya que brindan una interfaz gráfica poco amigable para el usuario común. Este problema se pudo mitigar con una aplicación llamada *uCap* que se puede instalar en routers Netgear N600, modelo WNDR3700v2, WNDR3800 y WDR3600 TP-Link N600 que corran con el firmware de projectBISmark. Esta aplicación les da a los usuarios la posibilidad de monitorear su uso de datos mediante una interfaz gráfica clara y simple.

Esta aplicación logra su propósito colectando y mandando información sobre el uso de datos de cada dispositivo conectado a la red a una base de datos. uCap usa el controlador OpenFlow Lithium (OPENDAYLIGHT) para permitirle a los usuarios configurar y manejar si se pasa de su cuota de uso, uCap lo bloquea.

Screenshots de uCap:

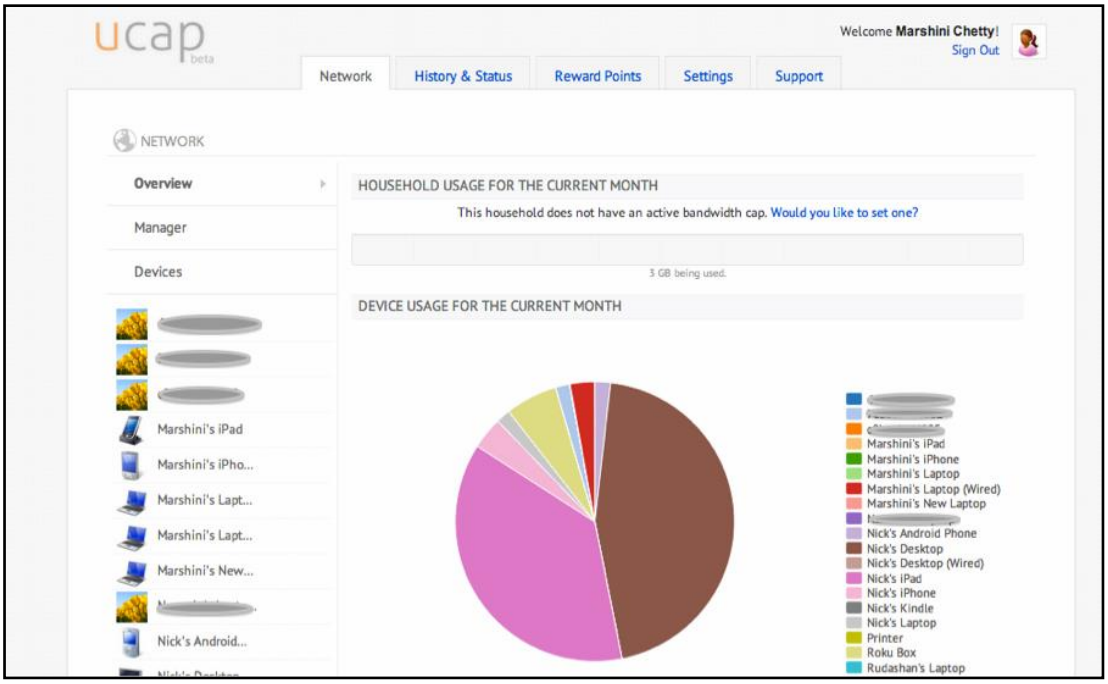


Imagen 42Ucap

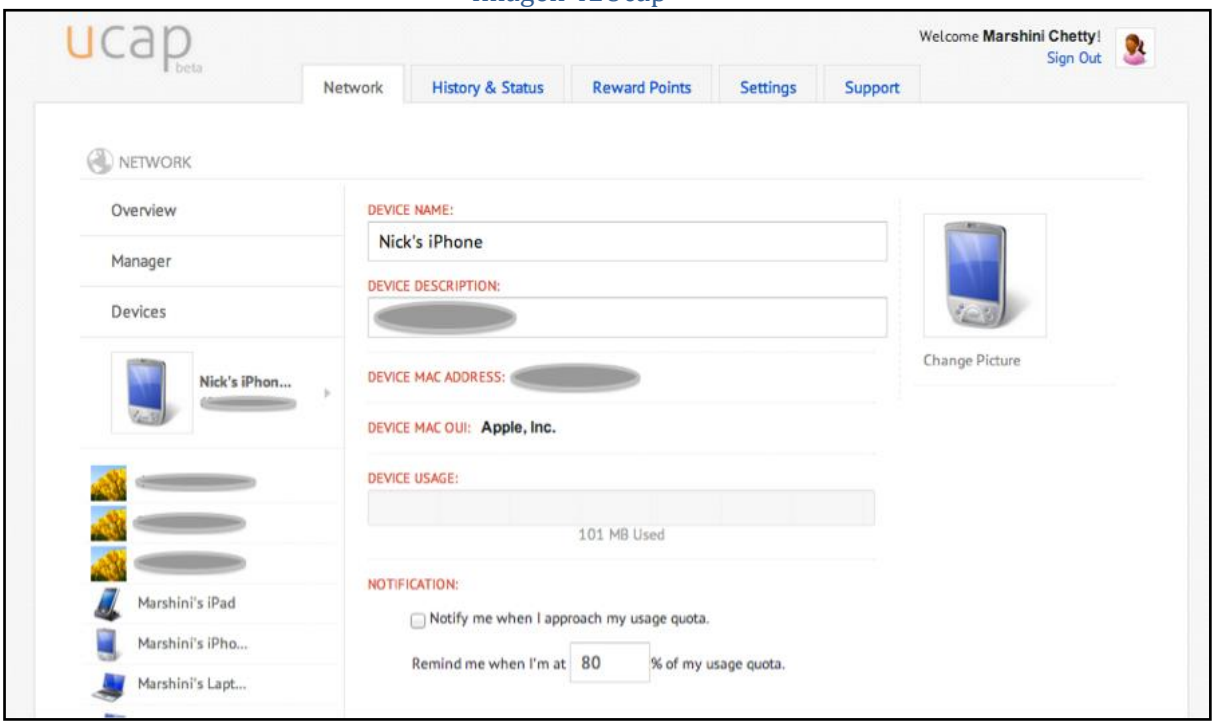


Imagen 43 Ucap

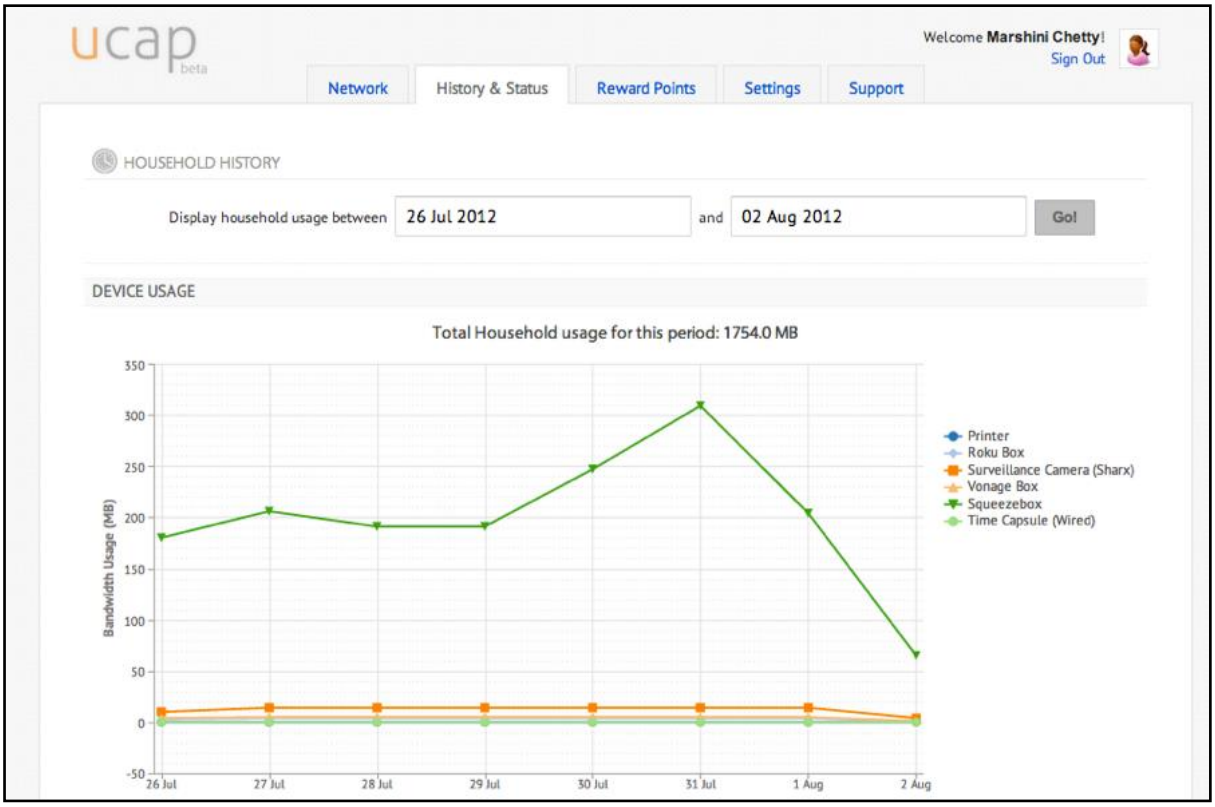


Imagen 44 uCap

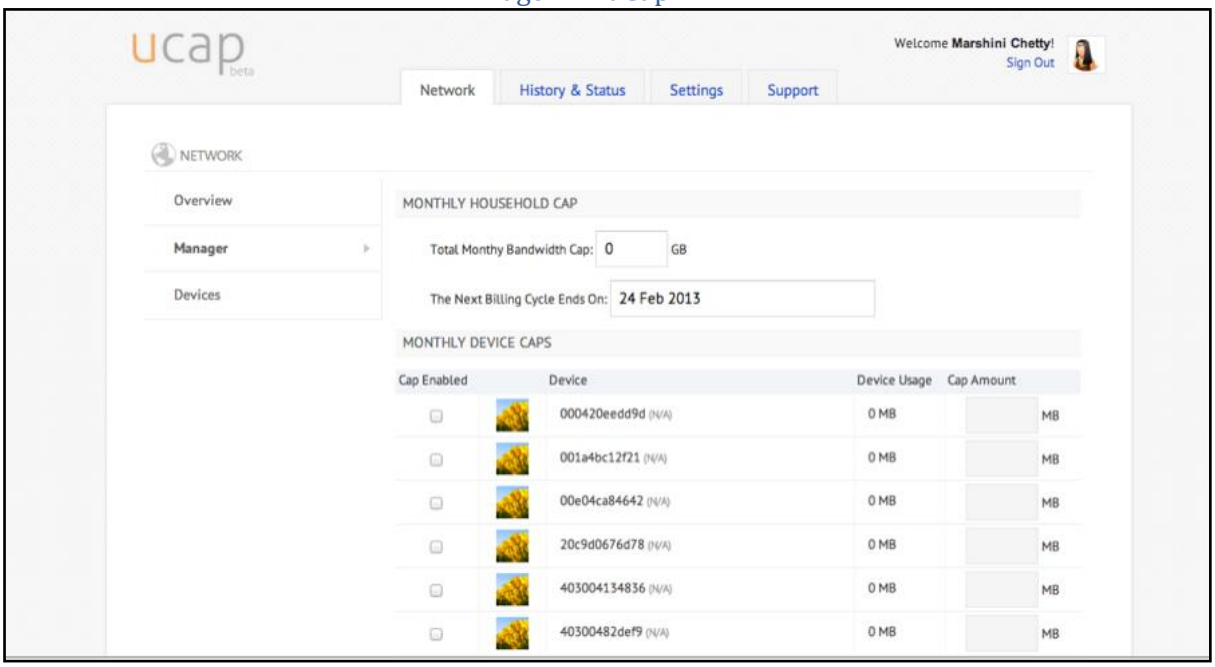


Imagen 45uCap

4.2 Vulnerabilidades de SDN Y OpenFlow

Una de las vulnerabilidades en seguridad de las redes definidas por software basadas en OpenFlow es que el canal de transmisión “del controlador al switch” puede ser encriptado o no. La encriptación utilizada se hace mediante el protocolo TLS el cual se explicará a continuación

4.2.1 Protocolo TLS (Transport Layer Security)

Es un protocolo mediante el cual se establece una conexión segura por medio de un canal cifrado entre el cliente y servidor. Así el intercambio de información se realiza en un entorno seguro y libre de ataques.

Normalmente el servidor es el único que es autenticado, garantizando así su identidad, pero el cliente se mantiene sin autenticar, ya que para la autenticación mutua se necesita una infraestructura de claves públicas (o PKI) para los clientes. Estos protocolos permiten prevenir escuchas (eavesdropping), evitar la falsificación de la identidad del remitente y mantener la integridad del mensaje en una aplicación cliente-servidor

4.2.1.1 Descripción del Protocolo

El protocolo SSL/TLS se basa en tres fases básicas:

- Negociación: Los dos extremos de la comunicación (cliente y servidor) negocian qué algoritmos criptográficos utilizarán para autenticar y cifrar la información. Actualmente existen diferentes opciones:

  Para criptografía de clave pública: RSA, Diffie-Hellman, DSA(Digital Signature Algorithm).

  Para cifrado simétrico: RC2, RC4, IDEA (International Data Encryption Algorithm), DES (Data Encryption Standard), Triple DES o AES (Advanced Encryption Standard).

  Con funciones hash: MD5 o de la familia SHA.
- Autenticación y Claves: Los extremos se autentican mediante certificados digitales e intercambian las claves para el cifrado, según la negociación.
- Transmisión Segura: los extremos pueden iniciar el tráfico de información cifrada y auténtica.

4.2.1.2 Objetivos del Protocolo

- Seguridad criptográfica: El protocolo se debe emplear para establecer una conexión segura entre dos partes.
- Interoperabilidad: Aplicaciones distintas deben poder intercambiar parámetros criptográficos sin necesidad de que ninguna de las dos conozca el código de la otra.

- Extensibilidad: El protocolo permite la incorporación de nuevos algoritmos criptográficos.
- Eficiencia: Los algoritmos criptográficos son costosos computacionalmente, por lo que el protocolo incluye un esquema de caché de sesiones para reducir el número de sesiones que deben inicializarse desde cero (usando criptografía de clave pública).

4.2.1.3 Funcionamiento del Protocolo

El protocolo está dividido en dos niveles:

- Protocolo de registro TLS (TLS Record Protocol).
- Protocolo de mutuo acuerdo TLS (TLS Handshake Protocol).

El de más bajo nivel es el Protocolo de Registro, que se implementa sobre un protocolo de transporte fiable como el TCP. El protocolo proporciona seguridad en la conexión con dos propiedades fundamentales:

- La conexión es privada. Para encriptar los datos se usan algoritmos de cifrado simétrico. Las claves se generan para cada conexión y se basan en un secreto negociado por otro protocolo (como el de mutuo acuerdo). El protocolo también se puede usar sin encriptación.
- La conexión es fiable. El transporte de mensajes incluye una verificación de integridad.

El Protocolo de mutuo acuerdo, proporciona seguridad en la conexión con tres propiedades básicas:

- La identidad del interlocutor puede ser autentificada usando criptografía de clave pública. Esta autenticación puede ser opcional, pero generalmente es necesaria al menos para uno de los interlocutores.
- La negociación de un secreto compartido es segura.
- La negociación es fiable, nadie puede modificar la negociación sin ser detectado por los interlocutores.

4.2.1.4 Aplicaciones

- El protocolo SSL/TLS tiene multitud de aplicaciones en uso actualmente. La mayoría de ellas son versiones seguras de programas que emplean protocolos que no lo son. Hay versiones seguras de servidores y clientes de protocolos como el http, nntp, ldap, imap, pop3, etc. El protocolo SSL/TLS se ejecuta en una capa entre los protocolos de aplicación como:
- HTTP sobre SSL/TLS es HTTPS, ofreciendo seguridad a páginas WWW para aplicaciones de comercio electrónico, utilizando

certificados de clave pública para verificar la identidad de los extremos. Visa, MasterCard, American Express y muchas de las principales instituciones financieras han aprobado SSL para el comercio sobre Internet.

- SSH utiliza SSL/TLS por debajo.
- SMTP y NNTP pueden operar también de manera segura sobre SSL/TLS.
- POP3 y IMAP4 sobre SSL/TLS son POP3S e IMAPS.
- Existen múltiples productos clientes y servidores que pueden proporcionar SSL de forma nativa, pero también existen muchos que aún no lo permiten. una solución podría ser usar una aplicación SSL independiente como Stunnel para conseguir el cifrado, pero IETF recomendó en 1997 que los protocolos de aplicación ofrecieran una forma de actualizar a TLS a partir de una conexión sin cifrado (plaintext) en vez de usar un puerto diferente para cifrar las comunicaciones, evitando el uso de envolturas (wrappers) como Stunnel. SSL también puede ser usado para tunelar una red completa y crear una red privada virtual (VPN), como en el caso de OpenVPN.

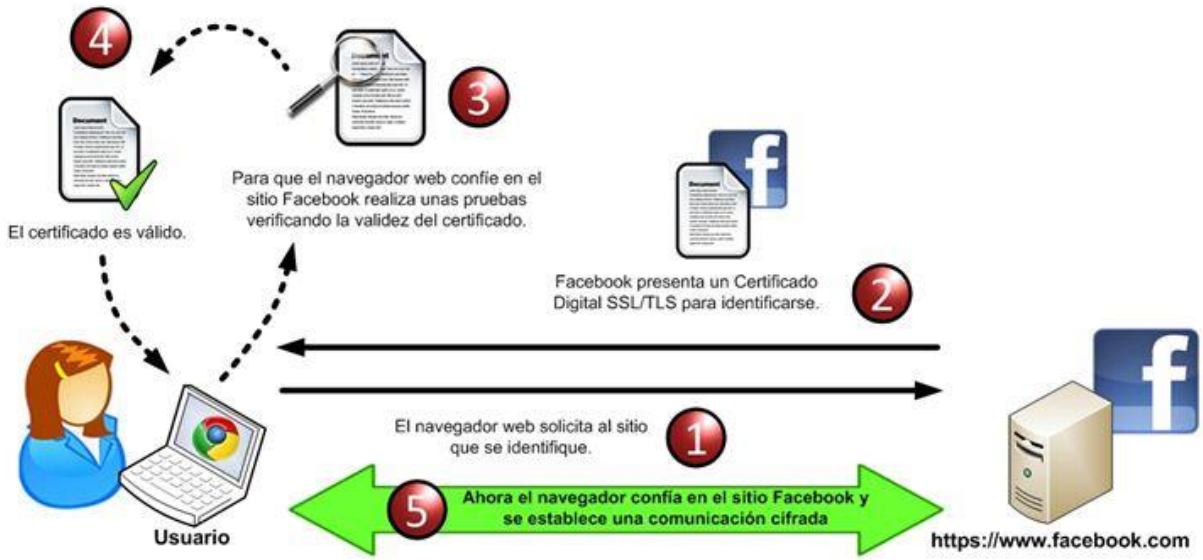


Figura 45 Encriptación TLS

4.2.2 Compatibilidad de Switches con TLS

Algunos switches que soportan TLS:

Brocade

Por defecto estos Switch trabajan con los puertos TCP 6633, 6653, o el que defina el usuario, sin encriptación y opcionalmente se puede encriptar los mismos puertos con TLS.

Cisco

Se puede configurar la seguridad TLS con el comando ***security TLS***

Extreme

Soporta encriptación EAP-TLS (Extensible Authentication Protocol - Transport Layer Security)

HP

Posee encriptación TLS en el canal con el controlador

Algunos switches que no soportan TLS:

Dell, Huawei, IBM, Juniper (Algunas de las últimas versiones de Junos OS si soportan TLS), Netgear, OVS.

Adicionalmente la versión de OpenFlow 1.0 solo soporta TLS, mientras que la versión 1.4 soporta TCP y TLS en la conexión Switch - Controlador. La conexión TCP se agregó para obtener mayor compatibilidad con equipos que no trabajen con TLS.

4.3 Vulnerabilidades de algunos controladores

Floodlight:
Entre sus vulnerabilidades conocidas están las siguientes:

- No tiene encriptación para las aplicaciones northbound http
- no requiere autenticación para estas mismas aplicaciones
- No soporta encriptación TLS

Opendaylight:
Entre sus vulnerabilidades conocidas están las siguientes:

- Soporta encriptación vía TLS pero no la requiere
- La encriptación de las aplicaciones northbound http vienen desactivadas por defecto
- Autenticación para aplicaciones northbound http es básica
- Password por defecto es débil
- Passwords más fuertes desactivados por defecto
- Existen exploits conocidos

POX:

- No soporta encriptación TLS

4.4 Mitigación de vulnerabilidades

4.4.1 VECTORES de ataques SDN

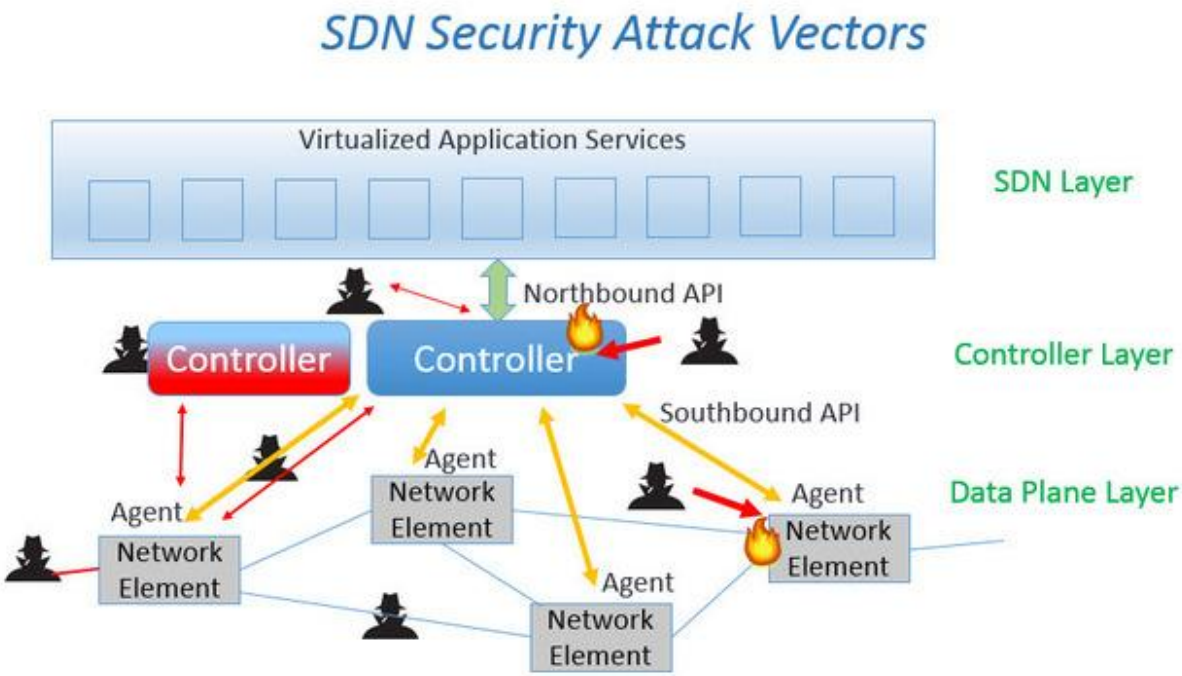


Figura46 Vectores de ataque a SDN

Vamos a separar el estudio de los vectores de ataque en 3 partes: El plano de datos, el plano de control y la capa de aplicaciones

4.4.1.1 Ataques a plano de datos

Los atacantes podrían apuntar a los elementos de la red desde dentro de la propia red. Un atacante podría teóricamente tener acceso físico o virtual no autorizado a la red o comprometer a un host que ya está conectado a la SDN y luego tratar de realizar ataques de desestabilizar los elementos de la red. Esto podría ser un tipo de denegación de servicio (DoS) o podría ser un tipo de ataque fuzzing para tratar de atacar a los elementos de la red.

Hay numerosas *Southbound APIs* y protocolos usados por el controlador para comunicarse con los elementos de red. Estas comunicaciones southbound pueden usar OpenFlow (OF), Open vSwitch Database Management Protocol (OVSDB), Path Computation Element Communication Protocol (PCEP), Interface to the Routing System (I2RS), BGP-LS, OpenStack Neutron, Open Management Infrastructure (OMI), Puppet, Chef, Diameter, Radius, NETCONF, Extensible Messaging and Presence Protocol (XMPP), Locator/ID Separation Protocol (LISP), Simple Network Management Protocol (SNMP), CLI, Embedded Event Manager (EEM), Cisco onePK, Application Centric

Infrastructure (ACI), Opflex, entre otros. Cada uno de estos protocolos tienen sus propios métodos de asegurar las comunicaciones de los elementos de red. Sin embargo, muchos de estos protocolos son muy nuevos y los implementadores pueden no configurarlos de la forma más segura.

Un atacante también podría aprovechar estos protocolos e intentar instanciar nuevos flujos en la tabla de flujos de los elementos de red. El atacante tendría que intentar introducir nuevos flujos para permitir un tipo específico de tráfico que no debería estar permitido en la red. Si un atacante podría crear un flujo que desvíe la dirección de tráfico que pasa a través de un firewall el atacante tendría una ventaja decisiva. Si el atacante puede dirigir el tráfico en dirección a ellos, pueden tratar de aprovechar esa capacidad para sniffear el tráfico y le realizará un ataque Man In The Middle (MITM).

A un atacante le gustaría espiar a escondidas para ver qué flujos están siendo usados y que tipo de tráfico es permitido en la red. El atacante podría intentar espiar la comunicación entre los elementos de red y los controladores. Esta información puede ser útil para un ataque de replay simplemente para los propósitos de reconocimiento.

Muchos sistemas SDN son desarrollados en data centers y estos usan un protocolos llamados Data Center Interconnect (DCI) como Network Virtualization using Generic Routing Encapsulation (NVGRE), Stateless Transport Tunneling (STT), Virtual Extensible LAN (VXLAN), Cisco Overlay Transport Virtualization (OTV), Layer 2 Multi-Path (L2MP), TRILL-based protocols (Cisco FabricPath, Juniper QFabric, Brocade VCS Fabric), Shortest Path Bridging (SPB), entre otros. Estos protocolos pueden carecer de autenticación y encriptación para asegurar el contenido de los datos. Estos nuevos protocolos podrían poseer vulnerabilidades debido a un aspecto del diseño del protocolo o de la forma en que el proveedor o cliente ha implementado el protocolo. Un atacante podría estar motivado a crear tráfico haciéndose pasar por otro elemento de la red de modo que modifique los links DCI o cree un ataque DoS a las conexiones DCI.

4.4.1.2 Ataques a la capa de control

Es obvio que el controlador de la red es un objetivo importante. Un atacante podría intentar apuntar al controlador para diferentes propósitos. El atacante podría instanciar nuevos flujos falsificando mensajes de las aplicaciones northbound o falsificando mensajes hacia los elementos southbound. Si un atacante puede falsificar flujos del controlador legítimos entonces él puede tener la habilidad de permitir tráfico a través de toda la SDN y posiblemente este pase las políticas de seguridad configuradas.

Un atacante puede intentar realizar una denegación de servicios del controlador o usar otro método que cause una falla del controlador. El hacker tendría la posibilidad de atacar de forma que haga que el controlador consuma muchos recursos y por lo tanto este responda muy lentamente comprometiendo la velocidad de la red.

Algunas veces los controladores SDN corren en algún tipo de GNU/Linux. Si el controlador corre en un sistema operativo de propósitos generales, las vulnerabilidades de ese equipo se convierten en vulnerabilidades del controlador. Algunas veces los controladores son implementados usando contraseñas por defecto y sin configuraciones de seguridad establecidas. Esto

se puede producir por el miedo del administrador de red a cambiar alguna configuración y que toda la red deje de funcionar cuando viene haciéndolo “bien por el momento”.

Por último, sería malo que un atacante cree su propio controlador y tomé la red como propia haciéndole creer a la red que él es el principal controlador. El atacante puede entonces crear entradas en las tablas de flujo y los administradores no tendrán visibilidad de esos flujos desde la perspectiva del controlador de producción. En este caso el atacante puede tener completo control de la red.

4.4.1.3 Ataque a la capa de aplicaciones

Atacando la seguridad de los protocolos northboundtambiénsería como un vector de ataque. Hay muchas aplicaciones que son usadas por los controladores SDN. Northbound APIs pueden usar Python, Java, C, REST, XML, JSON entre otros lenguajes de programación. Si un atacante puede usar a su favor las vulnerabilidades de las northbound APIs, entonces él puede tener el control de toda la red a través del controlador. Si el controlador carece de seguridad para las northbound APIs, entonces el atacante tiene la posibilidad de crear sus propias políticas para ganar control de todo el entorno de la SDN.
Algunas veces, hay una contraseña por defecto que es usada por las REST API que se determina trivialmente. Si en un desarrollo de una SDN no se cambia la contraseña por defecto y el atacante puede crear paquetes a través de la interfaz de gestión del controlador, entonces él puede consultar la configuración de toda la SDN y colocar su propia configuración.

4.4.2 Segurizando una SDN

Con la introducción de las SDN, se necesita un nuevo método de asegurar el tráfico del plano de control. En las redes tradicionales la seguridad del plano de control venia de la mano de la seguridad de los protocolos de enrutamiento que incluye MD5 para EIGRP, IS-IS o OSPFv2, IPsec, AH en el caso de OSPFv3, o GTSM/ACLs/ contraseñas para Mp-BGP. Algunos implementadores ni siquiera siguen estas simples técnicas para redes tradicionales. Si se utiliza las SDN con el mismo concepto de falta de seguridad se expondrán las organizaciones que implementen esta nueva tecnología a ataques.

4.4.2.1 Segurizando el plano de datos

Los sistemas típicos de SDN usan a su favor procesadores X86 y usan TLS para la seguridad del plano de control. Estas sesiones HTTP de larga vida son susceptibles a un amplio rango de ataques que pueden poner en peligro la integridad del plano de datos. Las organizaciones deberían preferir usar TLS para autenticarse y encriptar el tráfico entre el controlador y los dispositivos de red. Usando TLS ayuda a autenticar los controladores y elementos de red evitando el espionaje y falsificación de comunicaciones desde el southbound.

Dependiendo de los protocolos southbound usados, hay muchas opciones para asegurar estas comunicaciones. Algunos protocolos pueden usar las sesiones TLS antes mencionadas. Otros protocolos pueden usar contraseñas secretas compartidas para evitar ataques replay. Protocolos como SNMPv3 ofrecen más seguridad que SNMPv2c y SSH es mucho mejor que Telnet. Otros protocolos propietarios del Southbound pueden tener sus propios métodos de autenticación de dispositivos y controladores y encriptar información entre ellos, frustrando así el espionaje y la suplantación de identidad del atacante.

Las organizaciones pueden creer que una red privada tiene cierta seguridad inherente. Como las organizaciones extienden su propia redes virtuales y servicios de nube de SDN a data centers remotos, verificar el camino físico puede no ser fácil. Prevenir el acceso no autorizado es más fácil cuando una organización controla el acceso físico, pero en una red virtualizada, el actual camino físico no se ve claramente. Es difícil asegurar lo que no se puede ver.

4.4.2.2 Segurizando el plano de control

El controlador es un punto de ataque importante y también debe ser asegurado. Asegurando la postura de seguridad de este y de los elementos de la red viene de la mano de asegurar el sistema operativo en que corre el controlador. Todas las mejores prácticas para segurizar servidores públicos de Linux son aplicables aquí. Igualmente las organizaciones querrán monitorear de cerca sus controladores en caso de que haya alguna actividad sospechosa.

Las organizaciones también deben prevenir acceso no autorizado al controlador de la red. Los sistemas SDN deberían permitir la configuración de la seguridad y autenticar el acceso como administrador al controlador. Tambiéndeberían ser requeridas políticas basadas en roles de acceso para los administradores del controlador (Role-Based Access Control (RBAC)). Autenticación y auditorías serán útiles para controlar cambios no autorizados por administradores o atacantes también.

Si hay un ataque de denegación de servicios al controlador, entonces es beneficioso tener una arquitectura de Alta Disponibilidad. SDNs que usan controladores redundantes pueden sufrir una pérdida de un controlador y continuar funcionando. Esto puede elevar la complejidad para el atacante para realizar el ataque de DoS a todos los controladores en el sistema. Además, ese ataque no sería particularmente sigiloso y echaría a perder los objetivos del atacante de permanecer sin ser detectado.

4.4.2.3 Segurizando la capa de aplicaciones

Otra medida de protección es usar una red Out Of Band (OOB) para el control de tráfico.

Es más fácil y menos costoso construir una red OOB en un datacenter que en una empresa WAN. Usando una red OOB para las comunicaciones northbound y southbound puede ayudar a asegurar los protocolos para la gestión del controlador

Usando TLS o SSH u otro método para asegurar las comunicaciones northbound y asegurando el manejo del controlador seria considerando una

mejor práctica. Las comunicaciones de las aplicaciones y servicios que solicitan servicios o datos desde el controlador se deben asegurar con métodos de autenticación y cifrado.

Prácticas de codificación segura para todas las aplicaciones northbound que solicitan recursos SDN debería ser una buena práctica. No sólo son prácticas beneficiosas la codificación segura para la seguridad de las aplicaciones web de Internet de cara al público, pero también son aplicables a las conexiones northbound SDN.

Hay algunos sistemas SDN que tienen la habilidad de validar flujos en dispositivos de red contra la política del controlador. Este tipo de control (similar a FlowChecker) de los flujos en los dispositivos de red en contra de la política de la política podría ayudar a identificar las discrepancias que son el resultado de un ataque.

4.4.3 Diferencias entre SDN y redes convencionales con respecto a la seguridad

Ventajas y desventajas de sdn con respecto a su seguridad

Como en todo, habrá tanto beneficios como problemas en la aplicación de las nuevas tecnologías. Vamos a repasar algunos de los beneficios de las redes definidas por software:

Beneficios:

- Al tener el beneficio de una red definida por software, los administradores pueden hacer modificaciones en su estructura en forma rápida con un entorno de alto nivel.
- Esta libertad se traduce en una mejor seguridad para la red porque al tener una vista de toda la red desde un punto centralizado se pueden hacer cambios con eficiencia.
- por ejemplo si hay un brote de malware dentro de la red, con las SDN y OpenFlow se puede limitar rápidamente ese brote desde un punto centralizado deteniendo el tráfico sin la necesidad de acceder a múltiples routers o switches.
- Teniendo la posibilidad de cambiar rápidamente configuraciones de la red, le da la posibilidad al administrador de mejorar el tráfico dándole forma al QoS de paquetes en una forma segura, esta habilidad existe hoy pero no se puede lograr a la misma velocidad que con las SDN

Preocupaciones con respecto a las SDN:

Con la implementación de nuevas tecnologías los aspectos de seguridad es fácil que se pasen por alto. La mayoría de las preocupaciones con respecto a la seguridad en las SDN son respecto a su controlador ya que se considera el cerebro de la red y permite que esta sea manejada en forma centralizada.

Este elemento debe ser robustizado y segurizado de la siguiente manera:

- Sabiendo y auditando quien tiene acceso y donde se ubica en la red. Es importante recordar que el acceso al mismo puede otorgarle al atacante el completo control entonces es vital que se le dé seguridad al mismo.
- Verificar la seguridad entre el controlador y los nodos finales (routers y switches) en especial que se estén comunicando con SSL/TLS para prevenir accesos malintencionados. Como con todo lo demás si la seguridad no se implementa en el principio, debe implementarse más tarde y es más difícil y costoso de hacer. Hay que asegurarse que la seguridad entre el controlador y el nodo esté configurada correctamente.
- Verificar si los controladores tienen alta disponibilidad. Crear una alta disponibilidad de los mismos es importante porque si se pierde, la capacidad de gestionar de la red también se pierde y también todos los beneficios de las SDN y OpenFlow.
- Verificar que todo lo que se haga en el sistema quede registrado. Desde que los administradores tienen el control se debe registrar todos los cambios hechos.
- Cuando se implemente SDN verificar que los IPS, IDS Firewalls y alguna otra tecnología de filtrado que pueda bloquear tenga su actualización al día. Seguir eventos desde el SIEM (Security Información Event Manager) como fallas en acceso y cambios de políticas ayudaría a mejorar la seguridad de todo el sistema
- Verificar si el IPS no está identificando el tráfico del controlador como malicioso. Configurar las reglas de filtrado así el controlador se puede comunicar cuando lo necesite.

En conclusión, SDN es una tecnología emergente que puede permitir seguridad dando al administrador una visión completa de la red de la empresa. Este sistema es el cerebro de la red, y sin una configuración de seguridad hecha correctamente la red se vuelve vulnerable a ataques maliciosos o a cambios accidentales, lo cual ambos pueden tirar abajo por completo la red.

Capítulo 5

Implementación

5.1 Ejemplo de implementación

En este capítulo explicaremos los recursos disponibles y metodología utilizada para realizar diferentes experimentos de prueba en este proyecto.

5.2 Materiales Utilizados

Utilizaremos dos Notebooks, ambas corriendo Ubuntu 14.04 y un switch **HP2920-24G (J9726A).** El software utilizado será el analizador de vulnerabilidades OpenVas y el controlador openflow POX.

5.3 Experimentación

5.3.1 Por qué usamos pox

Una de las principales razones por la que elegimos POX es debido a su lenguaje de programación el cual es Python, ya que este no es difícil de interpretar y hace su curva de aprendizaje simple permitiendo leer y escribir código fácilmente

Otra ventaja de este controlador es que posee una comunidad amplia que lo usa por lo cual ya está muy probado y es más fácil de encontrar respuestas a inconvenientes de configuración

5.3.2 Instalación de pox

Abrimos una consola en Ubuntu y luego escribimos las siguientes líneas

```
sudo apt-get install git
git clone http://github.com/noxrepo/pox
cd pox
./pox.py openflow.of_01 --address=&lt;NUESTRAIP&gt; --port=6633
```

5.3.3 Sistema operativo

Dentro del abanico de sistemas operativos de código abierto disponibles hoy en día, la elección para la notebook donde instalamos el controlador fue Ubuntu 14.04.

Haciendo referencia a que la mayoría de los desarrollos e investigaciones en aplicaciones y en el mismo protocolo OpenFlow, se realizan sobre este sistema lo que evita dificultades a la hora de solucionar problemas de compatibilidad con otros sistemas.

5.3.4 Escaneador de vulnerabilidades

OpenVAS (Open VulnerabilityAssessmentSystem), inicialmente denominado GNessUs, es una suite de software, que ofrece un marco de trabajo para integrar servicios y herramientas especializadas en el escaneo y gestión de vulnerabilidades de seguridad de sistemas informáticos.

El escáner se corrió con una “Virtual Appliance” corriendo en una máquina virtual soportada por el software Virtualbox

5.3.5 Hardware

En esta parte del proyecto emplearemos el siguiente hardware

Características de la Notebook del controlador:

CPU: Intel core i5 2450 M 2.50 GHz
Memoria RAM: 8GB
Disco Rigido: 1T
Sistema Operativo: Ubuntu 14.04 de 64 bits

Características de la Notebook corriendo openvas:

CPU: Intel I7 4702M
Memoria RAM: 8GB
Disco Rigido: 1TB
Sistema Operativo: Ubuntu 14.04 de 64 bits

CaracteristicasSwitch HP 2920-24G (J9726A):

Diferenciador

- SwitchPoE+ de 24 Puertos L3 10/100/1000 con 2 ranuras para módulos opcionales de 10 G, 1 ranura para el módulo apilamiento opcional y una fuente de alimentación modular

Puertos

- (20) puertos RJ-45 10/100/1000 con detección automática

- (4) puertos RJ-45 10/100/1000 de doble función
- (2) ranuras para módulos

Memoria y procesador

- Tri Core ARM1176 a 625 MHz
- 512 MB de SDRAM
- Tamaño de búfer para paquetes: 11,25 MB (salida dinámica de 6,75 MB + entrada de 4,5 MB)
- 1 GB flash MB

Latencia

- Latencia de 100 Mb: < 9 µs
- Latencia de 1000 Mb: < 3,3 µs
- Latencia de 10 Gbps: < 3,3 µs

Velocidad

- hasta 95.2 Mpps

Capacidad de Switching

- 128 Gbps

Capacidad de apilado

- Virtual
- 4 conmutadores

Funciones de gestión

- IMC - Intelligent Management Center
- Interfaz de línea de comandos
- Navegador de Web
- Menú Configuración
- Administración fuera de banda (RJ-45 Ethernet)
- Administrador de SNMP
- Telnet
- RMON1
- FTP
- En línea y fuera de banda
- Administración fuera de banda (RS-232c serie o micro usb)

5.4 Proceso de implementación

En primer instancia se configuró apropiadamente el switch HP2920-24G J9726A para que funcione con el protocolo openflow cuyo archivo de configuración se adjunta en anexo.

Luego se procedió a conectar la siguiente topología de red.

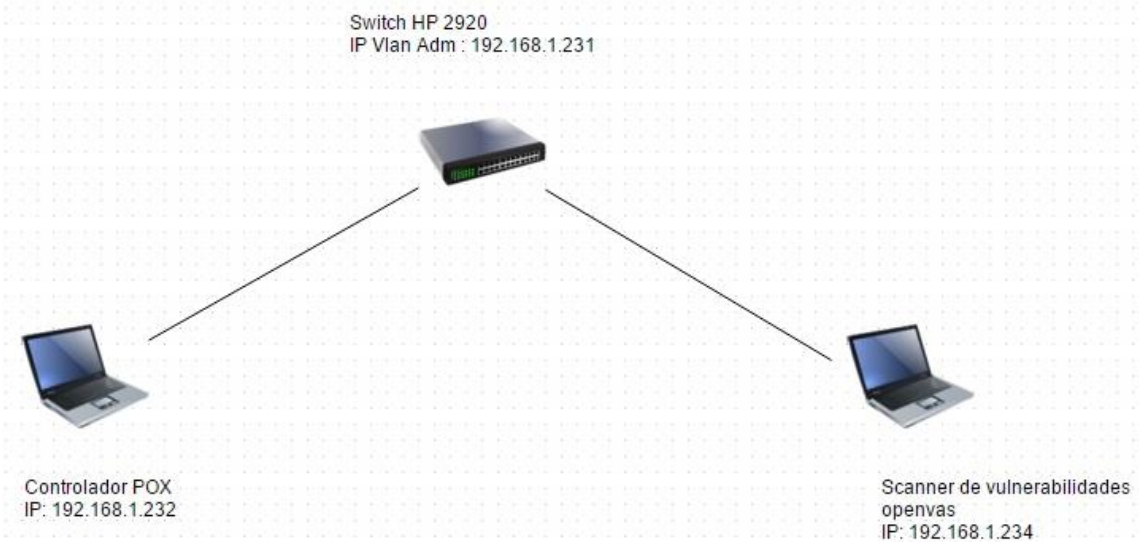


Figura 54 Topología de la implementación

Las pruebas fueron corridas en el modo “Full and Fast” del openvas, escaneando los puertos por defecto más algunos utilizados por openflow los cuales se pueden ver en el informe.

5.4.1 Escaneo al controlador

Mientras se corría esta prueba hacia el controlador en la IP 192.168.1.232 se podía apreciar que había una actividad contra él.

Screenshots del controlador mientras era escaneado:

```
sdnsecurity@sdnsecurity-laptop: ~/pox
sdnsecurity@sdnsecurity-laptop:~/pox$ ./pox.py
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:core:POX 0.2.0 (carp) is up.
INFO:openflow.of_01:[64-51-06-c5-90-c0|4096 1] connected
INFO:openflow.of_01:[None 2] disconnected
INFO:openflow.of_01:[None 2] closed
WARNING:openflow.of_01:Bad OpenFlow version (0x16) on connection [None 3]
INFO:openflow.of_01:[None 3] closed
WARNING:openflow.of_01:Bad OpenFlow version (0x47) on connection [None 4]
INFO:openflow.of_01:[None 4] closed
INFO:openflow.of_01:[None 5] closed
INFO:openflow.of_01:[None 6] closed
WARNING:openflow.of_01:Bad OpenFlow version (0x12) on connection [None 7]
INFO:openflow.of_01:[None 7] closed
WARNING:openflow.of_01:Bad OpenFlow version (0x5a) on connection [None 8]
INFO:openflow.of_01:[None 8] closed
WARNING:openflow.of_01:Bad OpenFlow version (0x16) on connection [None 10]
INFO:openflow.of_01:[None 10] closed
INFO:openflow.of_01:[None 9] closed
WARNING:openflow.of_01:Bad OpenFlow version (0x47) on connection [None 11]
INFO:openflow.of_01:[None 11] closed
WARNING:openflow.of_01:Bad OpenFlow version (0x4f) on connection [None 12]
INFO:openflow.of_01:[None 12] closed
WARNING:openflow.of_01:Bad OpenFlow version (0x4f) on connection [None 13]
INFO:openflow.of_01:[None 13] closed
ERROR:openflow.of_01:Exception reading connection [None 14]
Traceback (most recent call last):
  File "/home/sdnsecurity/pox/pox/openflow/of_01.py", line 912, in run
    if con.read() is False:
  File "/home/sdnsecurity/pox/pox/openflow/of_01.py", line 751, in read
    new_offset,msg = unpackers[ofp_type](self.buf, offset)
  File "/home/sdnsecurity/pox/pox/openflow/libopenflow_01.py", line 197, in unpack_new
    assert (r-offset) == length, o
AssertionError: ofp_hello
  header:
    version: 128
    type:    0 (OFPT_HELLO)
    length:  8
    xid:     1929256211
```

Imagen 54Screenshot controlador en escaneo

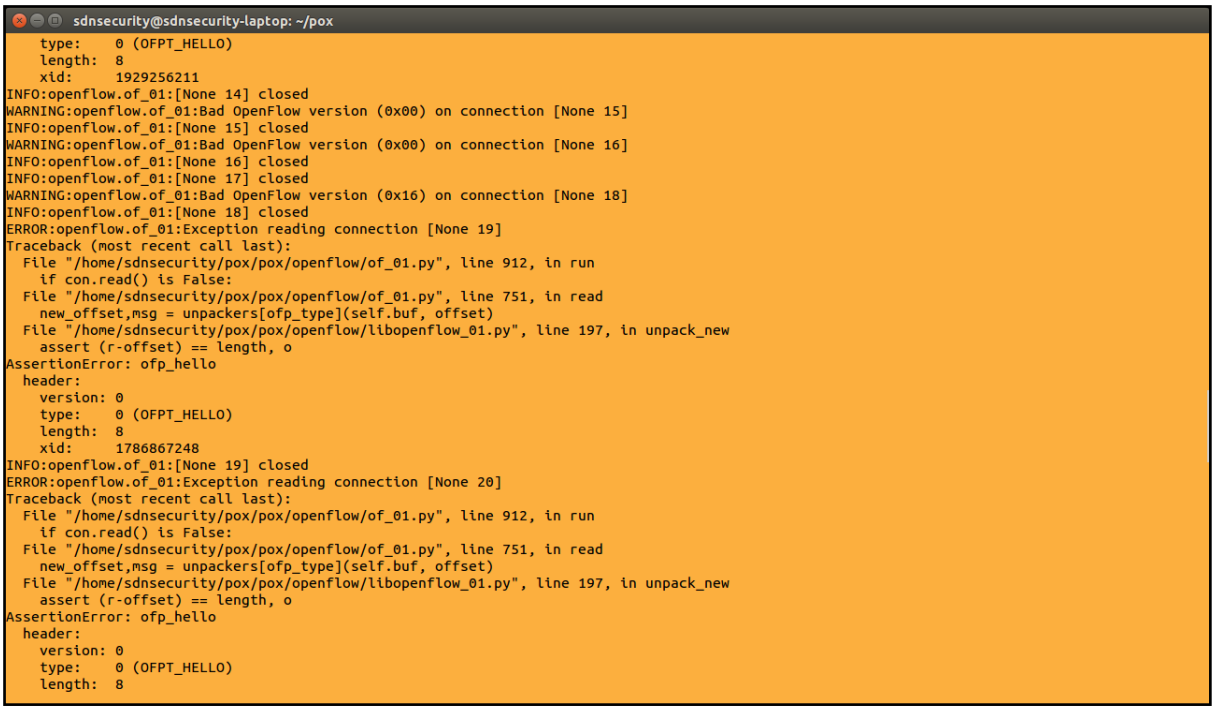


Imagen 55Screenshot controlador en escaneo

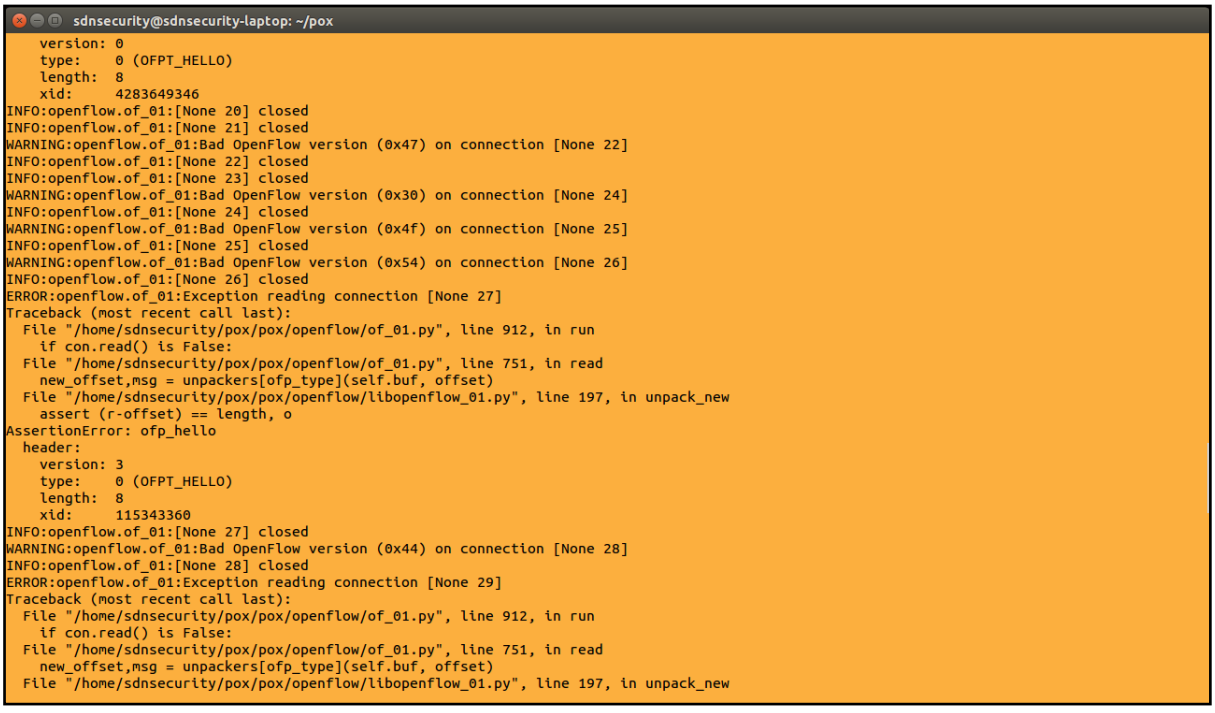


Imagen 56Screenshot controlador en escaneo

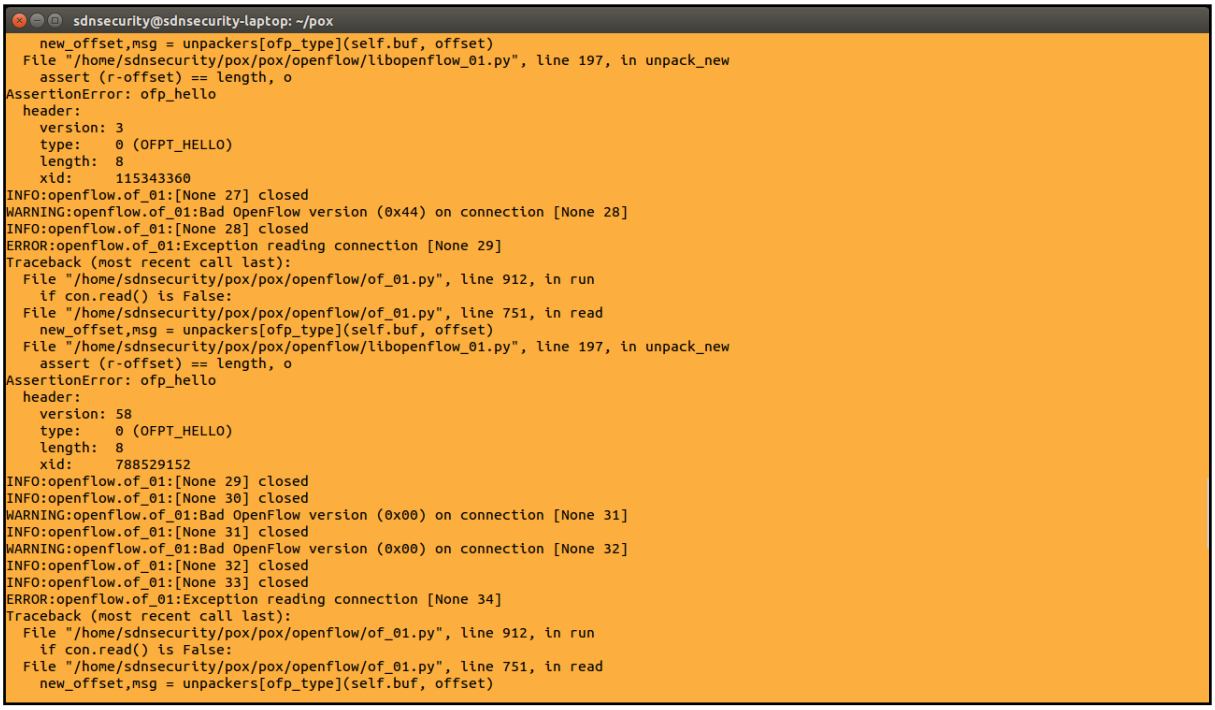


Imagen 57Screenshot controlador en escaneo

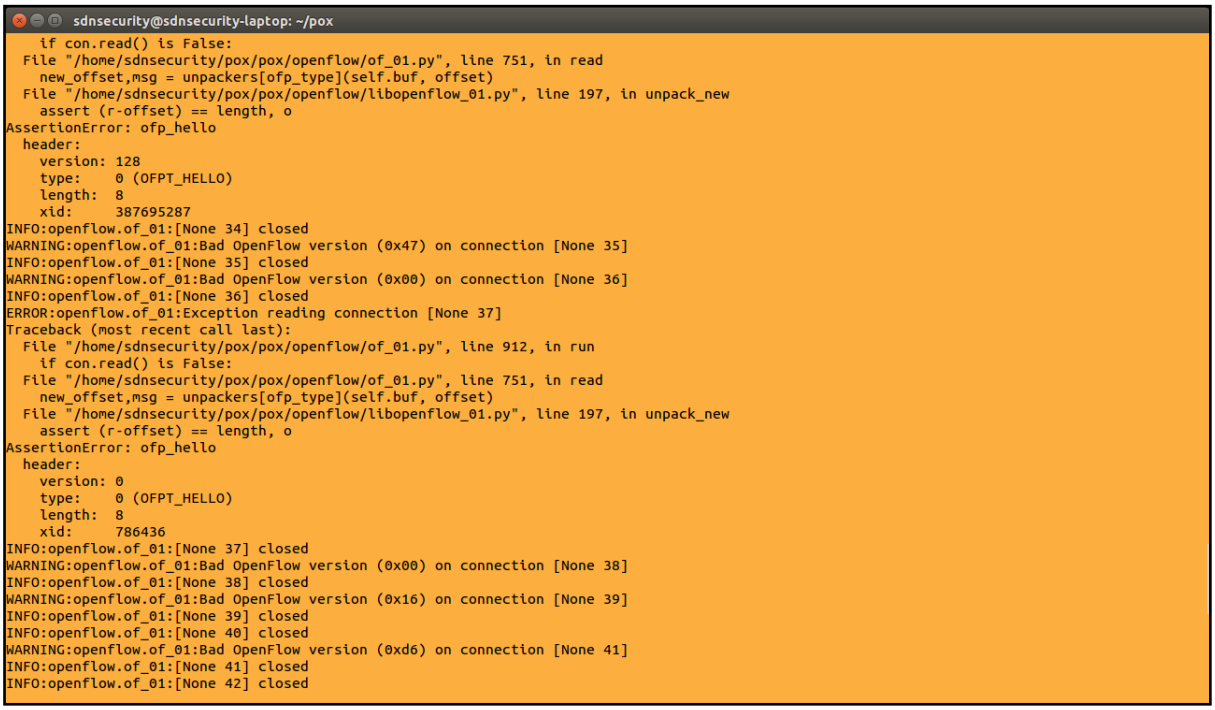


Imagen 58Screenshot controlador en escaneo

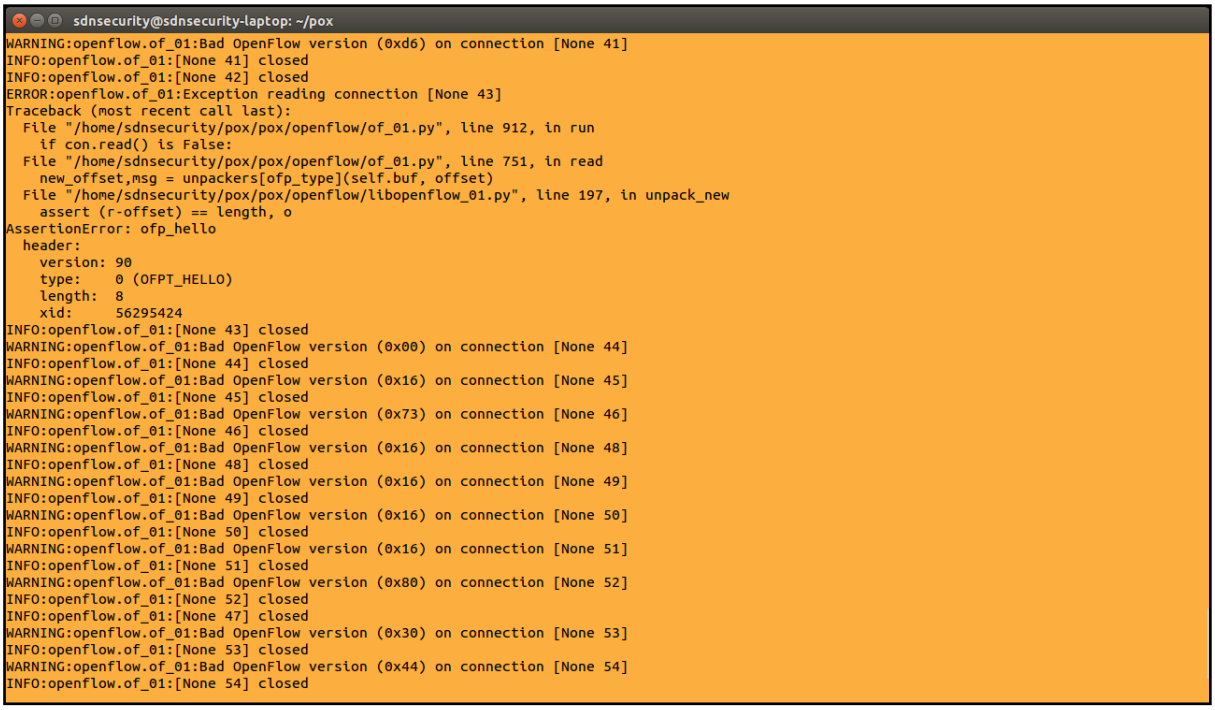


Imagen 59Screenshot controlador en escaneo

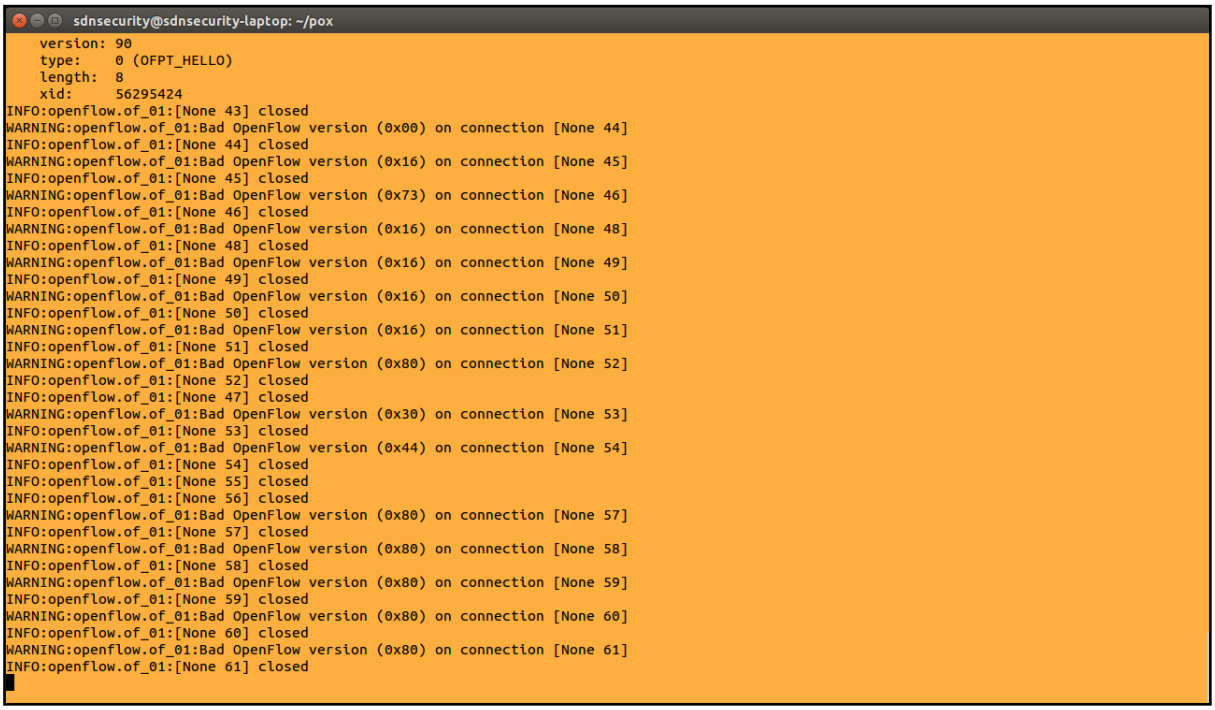


Imagen 60Screenshot controlador en escaneo

El Informe completo del escaneo se adjunta como pdf en el anexo. El mismo nos destaca las siguientes vulnerabilidades:

Service (Port)	Threat Level
9390/tcp	Medium
general/tcp	Low
9390/tcp	Log
general/tcp	Log
general/icmp	Log
general/CPE-T	Log
80/tcp	Log
6633/tcp	Log
443/tcp	Log

Tabla 54 Vulnerabilidades del controlador

La vulnerabilidad del puerto 9390 enumeradas en el informe son varias, algunas de las cuales son de facilsolucion, cambiando protocolos por versiones más nuevas, por ejemplo SSLv3 por TLSv1+ como sugiere el informe. También debemos tomar en cuenta que el controlador para esta prueba está corriendo sobre una computadora personal, no un equipo dedicado a este fin, por lo tanto se tiene muchas vulnerabilidades por puertos abiertos y protocolos en uso que pòsiblemente en un equipo dedicado al controlador no existirían.

También se ven vulnerabilidades sobre TCP, las mismas según sugiere el informe serían corregibles con un cambio en las configuraciones del protocolo.

Hay también fallas de seguridad consideradas menores, por ejemplo respuestas a protocolo ICMP, Traceroute o Valores de TIME STAMP que se pueden obtener del equipo, estos son corregibles simplemente configurando que no se conteste a esas consultas o protocolos en el caso ICMP.

Como se aclaró antes este informe es sobre una computadora personal, por lo tanto también observamos vulnerabilidades sobre servicios web, los mismos en un equipo dedicado al controlador no serían necesarios.

5.4.2 Escaneo al Switch

Se escaneo la IP 192.168.1.231 que es la dirección de la VLAN de administración. En esta prueba en el controlador no se ve tráfico. El resultado se adjunta como archivo pdf en el anexo. El mismo nos muestra solo amenazas de bajo nivel:

Service (Port)	Threat Level
general/tcp	Low
general/tcp	Log
general/CPE-T	Log
80/tcp	Log
23/tcp	Log
22/tcp	Log

Tabla 55 Vulnerabilidades del controlador

En este informe vemos por ejemplo los Timestamps que se envían en el protocolo TCP, que también pueden ser deshabilitados. También observamos que ICMP está activo, a través de este se podría obtener por ejemplo la versión del sistema, por lo que sería conveniente deshabilitar. También al igual que en el caso del controlador evitar el Trace Route.

En ambos casos vemos también que ante la solicitud de archivos no existentes en lugar de enviar “404 NotFound” puede responder enviando información disponible en el equipo.

El reporte también nos informa de una conexión telnet , la misma se debe a la computadora conectada al switch usada para configurar el mismo.

Capítulo 6

Conclusiones

Este proyecto fue interesante ya que pudimos investigar una tecnología en desarrollo y realizar práctica sobre los instrumentos utilizados hoy en día como son los switchs HP2920.

Nos demandó mucho tiempo la recolección de información sobre los distintos aspectos de las SDN utilizadas hoy en dia y sus enfoques de seguridad pero fue muy interesante aprender sobre ellos.

6.1 Sobre la experimentación

Fue inicialmente un aspecto complicado por el hecho de que utilizamos un sistema operativo al cual no estábamos acostumbrados a usar (Ubuntu 14.04) y se nos complicó con la configuración de las instalación y correcta configuración del software utilizado. Luego cuando pudimos configurar todo correctamente las pruebas pudieron realizarse sin mayores inconvenientes.

6.1.1 Sobre POX

Este no es el controlador más seguro ya que no soporta la encriptación TLS, sin embargo en él se pueden instalar las diferentes aplicaciones como firewalls, IDS, IPS que pueden aportar seguridad a la red.

6.1.2 Sobre openvas

Fue de lo más complicado de la implementación el lograr una configuración inicial correcta de esta herramienta, por lo cual optamos por el uso de la virtual appliance corriendo en una máquina virtual la cual también requiere de su configuración del adaptador de red en modo adaptador puente y configuración ip fija.

Luego de tener todo listo, la configuración del escaneo resultó simple e intuitiva. Se pudo elegir fácilmente qué puertos escanear y fueron agregados los más utilizados por el protocolo openflow

6.1.3 Sobre la adaptabilidad de la tecnología a diferentes entorno de redes

Otra de las grandes ventajas de SDN es que el fabricante solo provee el Hardware, dejando al administrador la libertad de elegir el software más conveniente y la configuración del mismo. Por lo tanto no es necesario la elección de todos los dispositivos del mismo vendedor. Todo esto es gracias a que el protocolo Openflow utilizado se encuentra regulado por la ONF, una organización que regula los estándares del este protocolo.

6.1.4 Sobre la seguridad de esta nueva tecnología

Esta nueva tecnología proporciona opciones de seguridad que las redes convencionales no. El hecho de poder aislar nodos de forma instantánea agiliza de gran manera el trabajo del administrador de red. Asimismo ofrece la opción de automatizar la mitigación de anomalías con el uso de scripts programados a la necesidad del administrador, esto implica que este tenga conocimientos sobre los lenguajes de programación utilizados en las aplicaciones.

Estas aplicaciones se implementan con solo correrlas en el controlador, depende de cada controlador su forma de ejecución. En pox por ejemplo solo con escribir la ruta de donde se encuentra la aplicación, ésta se ejecuta.

6.1.5 Sobre las características de seguridad de su arquitectura

Sobre las vulnerabilidades de la arquitectura de las SDN podemos destacar las siguientes características más relevantes.

Plano de datos:

Uno de los problemas del plano de control es que se utilizan protocolos muy nuevos y por eso mismo pueden no ser configurados correctamente al momento de su implementación.

Un ámbito de uso de las SDN son los data center los cuales tienen protocolos específicos como DCI, NVGRE, STT entre otros. Estos pueden carecer de encriptación los cuales los vuelven vulnerables.

Capa de control:

Este punto es el máscrítico de todo el sistema ya que si un atacante lo llegara a comprometer, este tomaría el control de toda la red, esta es la desventaja más fuerte con respecto a las SDN. A Veces los controladores corren en algún tipo de GNU/Linux como en el caso de nuestra implementación, por lo tanto las vulnerabilidades de este SO se vuelven vulnerabilidades del controlador y por lo tanto de la red misma.

6.1.6 Sobre la seguridad de su gestión

Con la implementación de las SDN no habría necesidad de recurrir a una herramienta de gestión de seguridad para todos los planos ya que las diferentes aplicaciones corren directamente sobre el controlador, a lo sumo se podría utilizar un controlador diferente para instalar aplicaciones programadas en otro lenguaje distinto al controlador utilizado anteriormente.

6.1.7 Aspectos económicos

Desde un punto de vista económico, las SDN brindan un ahorro importante ya que algunos elementos de hardware como firewalls pueden ser emulados con aplicaciones instaladas en el controlador.

También ahorra tiempo de trabajo al administrador de la red permitiéndole realizar una gestión másrápida y eficiente.

Cabe destacar otro aspecto importante como la implementación por etapas ya que al poder utilizar los dispositivos con tráfico OpenFlow y tráfico convencional permitiéndonos hacer un cambio gradual de equipos y habilitando los servicios SDN a medida que sea necesario y posible.

6.2 Observaciones finales

La seguridad en las SDN es un aspecto poco investigado hoy en día, por lo cual no hay mucha información sobre casos prácticos, son más que nada teóricos. El aspecto más importante y práctico encontrado fue sobre la encriptación del canal de administración entre el controlador y el switch y las configuraciones contraseña de fábrica que no todos los administradores de red modifican. Esto nos muestra la importancia de la elección del controlador, ya que el POX utilizado en esta implementación no soporta encriptación TLS como sí hacen otros, esto permitiría que usuarios no deseados capturen tráfico (sniffing).

Existen desarrollos en el área de seguridad de las SDN. Podemos mencionar por ejemplo la aplicación de control DefenseFlow de la compañía Radaware que es una herramienta muy completa e intuitiva para detectar y proteger de ataques de denegación de servicios.

Desde nuestro punto de vista las SDN usan soluciones de redes convencionales para mitigar vulnerabilidades. De todas formas creemos conveniente la implementación de esta tecnología ya que tienen ventajas sobre las redes convencionales y además pueden convivir con ellas, ya que un switch que soporte el protocolo openflow puede trabajar con el tráfico utilizado hoy en día.

Referencias y bibliografías

Nadeau, Thomas D. & Gray, Ken. (2013). SDN: Software Defined Networks. EstadosUnidos: O'Reilly Media

Hommes, Stefan. (2014). Fault Detection and Network Security In Software Defined Networks With Openflow. (PhD Thesis). Université du Luxembourg, Luxemburgo..

Klöti, Rowan. OpenFlow: A Security Analysis. (Master Thesis).Zurich, Switzerland.

Herrando, Adolfo N. y Kim, Insong. Openflow: Redes definidas por software. (Tesis de grado). Instituto Universitario Aeronáutico, Córdoba, Argentina

Braga, Rodrigo. Mota, Edjard. Passito, Alexandre. (Octubre, 2010). Lightweight DDoS Flooding Attack Detection Using NOX/OpenFlow. 35th Annual IEEE Conference on Local Computer Networks.

(2015). OpenDaylight Project. Recuperado de https://en.wikipedia.org/wiki/OpenDaylight_Project

Linux Foundation. OpenDaylight. Recuperado de www.opendaylight.org

NOX. Recuperado de http://www.noxrepo.org/

Linux Foundation. Open Network Summit. Recuperado de http://opennetsummit.org/

UCap. Recuperado de http://ucap.projectbismark.net/promo/

Google Cloud Platform Blog. Enter the Andromeda zone - Google Cloud Platform's latest networking stack. Recuperado de http://googlecloudplatform.blogspot.com.ar/2014/04/enter-andromeda-zone-google-cloud-platforms-latest-networking-stack.html

Network computing. Inside Google's Software-Defined Network: Google B4. Recuperado de http://www.networkcomputing.com/networking/inside-googles-software-defined-network/a/d-id/1234201?

B4: Experience with a Globally-Deployed Software Defined WAN. Recuperado de http://cseweb.ucsd.edu/~vahdat/papers/b4-sigcomm13.pdf

EcuRed. Protocolo TLS. Recuperado de http://www.ecured.cu/index.php/TLS

Brocade. TLS support. Recuperado de https://www.brocade.com/content/html/en/user-guide/SDN-Controller-2.1.0-User-Guide/GUID-BBA83BAB-79CC-4732-A44D-E2711BBB2D8D.htmlSwitch

CISCO. Cisco Plug-in for OpenFlow Commands. Recuperdo de http://www.cisco.com/c/en/us/td/docs/switches/datacenter/sdn/command/openflow-cr-bookmap/cr-openflow.html

Extreme Networks. ExtremeXOS OpenFlow User Guide. Recuperado de http://documentation.extremenetworks.com/openflow/_Common/Glossary/E.shtml

Dell. Dell OpenFlow Deployment and User Guide. Recuperado de http://topics-cdn.dell.com/pdf/force10-sw-defined-ntw_Deployment%20Guide3_en-us.pdf

HP. HP OpenFlow Protocol Overview. Recuperado de http://h17007.www1.hp.com/docs/networking/solutions/sdn/devcenter/03_-_HP_OpenFlow_Technical_Overview_TSG_v1_2013-10-01.pdf

Juniper. OpenFlow v1.0 Compliance Matrix for EX4550 Switches. Recuperado de http://www.juniper.net/documentation/en_US/junos15.1/topics/reference/general/junos-sdn-openflow-v1.0-compliance-matrix-ex4550.html

Hogg, Scott. Network World. (2014). SDN Security Attack Vectors and SDN Hardening. Recuperado de http://www.networkworld.com/article/2840273/sdn/sdn-security-attack-vectors-and-sdn-hardening.html

Pascucci, Matthew. Search Security. (2012). Software-defined networking: Exploring SDN security pros and cons. Recuperado de http://searchsecurity.techtarget.com/tip/Software-defined-networking-Exploring-SDN-security-pros-and-cons

Benton, K. Camp, L. J. Y Small, C. OpenFlow Vulnerability Assessment. Recuperado de http://homes.soic.indiana.edu/ktbenton/research/openflow_vulnerability_assessment.pdf

Radaware. DefenseFlowNetFlow and SDN based DDoS Attack Defense. Recuperado de http://www.radware.com/Products/DefenseFlow/

Feamster, Nick. Software Defined Networking. Recuperado de https://www.coursera.org/course/sdn1

Anexos

Configuración del switch:

```
J9726A Configuration Editor;
Created on release #WB.15.16.0005
;
Ver #06:0c.fc.f3.ff.35.0d:c2


hostname "HP-2920-24G"
module 1 type j9726a
timesyncsntp
sntp broadcast
timetimezone -3

ip access-list extended "admin"
  exit
ip authorized-managers 192.168.1.231 255.255.255.0
access manager
interface 1
  ip access-group "admin" in
  exit
snmp-server community "public" unrestricted
snmp-server contact "iua"
openflow

controller-id 1 ip 192.168.88.230 controller-interface vlan 2

controller-id 2 ip 192.168.88.236 controller-interface vlan 2

controller-id 3 ip 192.168.1.232 controller-interface vlan 1

instance aggregate
     controller-id 3
     enable
     exit
  enable
  exit
oobm

ip address dhcp-bootp
  exit

vlan 1
  name "DEFAULT VLAN"
  no untagged 13-16,21-24
  untagged 1-12,A1-A2,B1-B2
  tagged 17-20
  ip address 192.168.1.231 255.255.255.0
```

```
dhcp-server
  exit

vlan 2
  name "VLAN2"
  untagged 13-24
  ip address 192.168.2.230 255.255.255.0

dhcp-server
  exit

noautorun

nodhcpconfig-file-update

nodhcp image-file-update

dhcp-server pool "rango1"
  authoritative
  network 192.168.1.0 255.255.255.0
  range 192.168.1.232 192.168.1.234
  exit

dhcp-server pool "rango2"
  authoritative
  network 192.168.2.0 255.255.255.0
  range 192.168.2.230 192.168.2.232
  exit

dhcp-server enable

password manager

passwordoperator
```