Diseño de una Red Definida por Software empleando una solución basada en Software, para la Infraestructura de Cloud de la Facultad de Ingeniería en Ciencias Aplicadas (FICA)

Autores – Johanna Lisseth LEYTON PORTILLA, Ing. Carlos Alberto VÁSQUEZ AYALA

Facultad de Ingeniería en Ciencias Aplicadas, Universidad Técnica del Norte, Avenida 17 de Julio 5-21 y José María Córdova, Ibarra, Imbabura

jlleytonp@utn.edu.ec, cavasquez@utn.edu.ec

Resumen. *El presente proyecto tiene como finalidad presentar el diseño de una Red Definida por Software (SDN), como una solución a las limitaciones de las redes actuales y las tendencias de tráfico, realizando un estudio referente a la arquitectura de las redes SDN tomando en cuenta la separación de los planos de datos y de control, además de la investigación del protocolo Openflow en el cual están basadas dichas redes.*

En la infraestructura de las redes SDN se presenta un controlador, para lo cual se toma cuatro alternativas: NOX, POX, Beacon y Floodlight, con los cuales se realizarán pruebas para determinar el más óptimo y adecuado en la implementación de la red, otro elemento de la infraestructura es un switch virtual, la adpatación del mismo es por medio del software Open vSwitch, al mismo tiempo se definen las reglas de flujo.

En la implementación de la red se utilizara el software KVM y virtual manager, el cual permite crear máquinas virtuales, además de que toda la implementación se realizara sobre el sistema Operativo Ubuntu, finalmente se desarrolla una aplicación para el control de acceso al medio (NAC).

Palabras Claves

Redes Definida Por Software (SDN), OpenFlow, NOX, POX; Beacon, Floodlight, Open vSwitch, NAC.

Abstract. *This project present the design of a Network Software Defined (SDN) as a solution to the limitations of current networks and traffic trends, conducting a study concerning the architecture of SDN networks, taking into account the separation of the control and data planes, in addition to research OpenFlow protocol , the networks are based in that protocol.*

In the SDN network infrastructure a controller is presented, for that four alternatives is taken: NOX, POX, Beacon and Floodlight with which one tests will be performed to determine the most optimal and appropriate in the implementation of the network, another infrastructure element is a virtual switch, the adaption of it is through the Open vSwitch software while flow rules are defined.

In the implementation of the network the KVM software and virtual manage are usedr, which lets create virtual machines, in addition to all the implementation will take place on the Ubuntu operating system, finally an application for media access control is developed (NAC)

Keywords

Software Defined Networks (SDN), OpenFlow, NOX, POX; Beacon, Floodlight, Open vSwitch, NAC.

1. Introducción a SDN

Las empresas y usuarios de una red en la actualidad presentan muchos problemas en sus redes tradicionales, puesto que no cumplen con los requerimientos solicitados, por lo que las Redes Definidas por Software (SDN), se presentan como una solución estandarizada para cubrir las necesidad de las empresas y los usuarios de red, dando paso a la transformación de la arquitectura de las redes tradicionales.

Utilizar las redes tradicionales para satisfacer los requerimientos de telecomunicaciones, generados por los usuarios resulta improbable; las empresas a nivel mundial utilizan los departamentos de IT para aprovechar al máximo sus redes, como una solución pasajera, debido a que las redes existentes no están diseñadas para cubrir los

requerimientos actuales de los usuarios, compañías y proveedores. [1]

Con la arquitectura de red SDN se pueden crear redes que cubran los requerimientos de los usuarios y empresas en la actualidad, debido a que permita la separación del plano de control y el plano de datos, para que las personas encargadas de la administración de la red, lo realicen de forma centralizada. La manipulación del plano de control y del plano de datos se realiza por separado, para el plano de control se utiliza el protocolo OpenFlow, por medio de equipos controladores y el plano de datos es operado por los equipos de conexión.

2. Arquitectura de una SDN

Los dispositivos de conectividad en una red están integrados por dos componentes importantes: el plano de control y el plano de datos, el plano de datos es el encargado de la transmisión de los datos y el plano de control determina el camino circulación de la informacion para llegar a su destino.Una Red Definida por Software (SDN) aparece como una solución para las demandas y requerimientos tanto del plano de control como del plano de datos. [2]

En la figura 1 se representa la arquitectura de una Red Definida por Software.

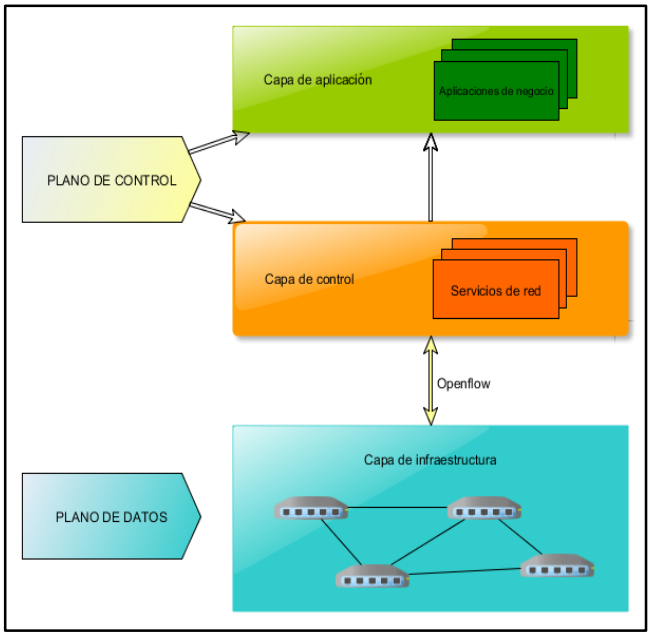


Figura 1. Arquitectura SDN
Fuente: SDxCentral. (2012). What's Software Defined Networking (SDN)? Definition. Obtenido de: https://goo.gl/i94XlG

Capa de infraestructura

Se encuentra conformada por los elementos de la red que se encargan de la conmutación y el enrutamiento de los paquetes de datos, ofreciendo acceso a través de una API, como es OpenFlow.

Capa de control

La función de control es centralizada, permite la utilización de las capacidades de la red, el controlador de una SDN tiene el control de un conjunto de recursos del plano de control, permitiendo el encaminamiento del tráfico de red, se encarga de la conmutación de datos y de elegir el camino óptimo para la transmisión, el controlador gestiona un rango de recursos del plano de datos, facilitando la configuración del mismo.

Capa de aplicación

Consta de las aplicaciones solicitadas por los usuarios finales, por medio de las API de SDN las cuales permiten la utilización de servicios de comunicación, la capa de aplicación logra que las configuraciones simples y da paso a la gestión de nuevos servicios, formas de acceso y mejora de la red. [3]

OpenFlow

OpenFlow es una interfaz utilizada en las redes SDN, permite la comunicación entre los planos de datos y de control de los equipos que forman la red, ofrece la facilidad de acceder al plano de control y manipularlo de forma directa, ya sea en dispositivos físicos o virtuales.

3. Protocolo OpenFlow

OpenFlow es un protocolo reciente en lo que se refiere a comunicaciones, el cual permite determinar el correcto enrutamiento de los paquetes en una red, utilizando OpenFlow una red puede ser manipulada como un solo ente, centralizando las decisiones de movilidad de paquetes, en OpenFlow un controlador se encarga del determinar el encaminamiento de nivel alto. [4]

En una red SDN el protocolo OpenFlow debe estar operativo en los equipos que forman la red y en el controlador de la misma, para distinguir el tráfico de la red el protocolo utiliza flujos, que definen como el trafico viaja a través de los dispositivos de red tomando en cuenta parámetros de uso y aplicaciones, basándose en reglas predeterminadas en el controlador.

Un Switch OpenFlow es un dispositivo que envía paquetes de una red SDN el cual opera con la utilización de tablas de flujo y acciones de cada entrada del flujo, siendo posible dividir el tráfico de la red, el Switch OpenFlow permite la adopción de protocolos nuevos, modelos de seguridad y diseños de direccionamiento, el tráfico de la red no se ve perjudicado debido a que trata de forma aislada y se procesa de igual manera que en una red tradicional, se encuentra compuesto por 3 parámetros principales:

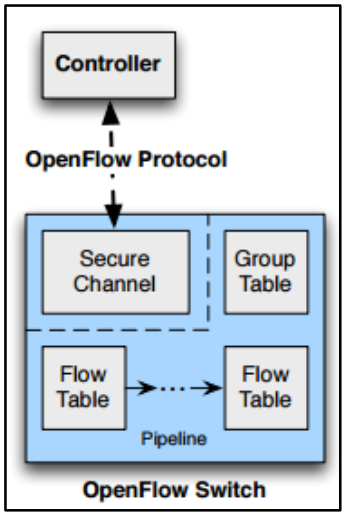


Figura 2. Componentes de switch OpenFlow
Fuente: OpenFlow. (2011). *OpenFlow Switch Specification*. Obtenido de: http://archive.openflow.org/documents/openflow-spec-v1.1.0.pdf

4. Controladores

4.1 NOX

En OpenFlow, NOX es el controlador original, fue desarrollado por Nicira, NOX es básicamente una plataforma que permite crear aplicaciones para el control de la red, la versión más actual de NOX ofrece un framework más liviano y de mayor rendimiento, su desarrollo se hace en el lenguaje C++ para Linux.

Funcionamiento

NOX funciona a base de tablas de flujo, el manejo de dichas tablas lo realiza por medio de una interfaz de programación en la cual se pueden configurar aplicaciones que proporcionen mayor control de acceso a la red ,dentro de los eventos de la misma, mediante la manipulación de los flujos de red se puede controlar las decisiones de reenvío y el tráfico de la red de una forma escalable; cuando se detecta una nueva entrada de flujo, dicho flujo es enviado al controlador y allí se lo transfiere a las aplicaciones adecuadas, además el correcto manejo de los flujos permite la modificación y análisis de paquetes y la obtener estadísticas. [5]

4.2 POX

NOX fue tomado como base para el desarrollo de POX, es un controlador de OpenFlow que se encuentra en constante desarrollo, creado para cubrir las necesidades de las SDN con el lenguaje Python para otros sistemas operativos.Las características principales de POX son:

- Su interfaz está desarrollada en Python.
- Se desarrolló para Windows, Mac OS y Windows.
- La interfaz gráfica y las herramientas para visualizar son parecidas a las de NOX
- Los elementos muestran la ruta de acceso, la topología, entre otras. [6]

4.3 Beacon

Beacon es un controlador muy eficaz y rápido, multiplataforma y modular, está desarrollado en el lenguaje Java, además provee paquetes con características avanzadas que permiten determinar el mejor camino y el más corto, la detección de la topología y una interfaz web, este controlador está basado en eventos y threads que son subprocesos planificados para un sistema. [7]

Beacon funciona por medio de la utilización de módulos llamados “budles”, los cuales poseen metadatos, clases de Java y otros archivos, los budles de Beacon funcionan en conjunto, las características de los budles son:

- Pueden importar y exportar paquetes Java.
- Ofrecen una interfaz web de gestión.
- Soportan OpenFlow 1.0,
- Compatibilidad con paquetes TCP/UDP e IP
- Operaciones basadas en eventos e hilos.
- Ofrecen módulos para la creación de aplicaciones.
- Se pueden modificar paquetes en la ejecución sin dañar otros paquetes.
- Manejan Proxies ARP y DHCP. [8]

4.4 Floodlight

Este controlador fue desarrollado en Java con una licencia Apache, puede resistir tanto switches virtuales como físicos, está compuesto por un módulo principal, el cual se encarga de identificar los paquetes y asignar eventos, también lo forman módulos secundarios registrados con el modulo principal, encargados de operar y anexar entradas a las tablas de flujos.

El funcionamiento de Floodlight se basa en módulos entre los cuales existen dos tipos:

Módulos de controlador

En los módulos del controlador se encuentran las características que se usan en las aplicaciones continuamente, como la revelación de topología y los eventos, el enlace con el controlador, una interfaz web de usuario, y los recursos compartidos de la red.

Módulos de aplicación

Estos módulos se encargan de introducir funciones determinadas, como las de un switch capa 2, firewall o enlaces bajados. [9]

4.5 Comparativa entre controladores

Cada uno de los controladores se integra de forma sencilla con el Open vSwitch, una vez que se aclaró la función de las interfaces. Cabe mencionar que el software Open vSwitch posee un gran soporte debido a que es muy utilizado en la infraestructura de SDN.

NOX es simple para instalar, debido a que es el primer controlador que se desarrolló para las redes SDN, no brinda todos los componentes requeridos para las aplicaciones, el soporte del mismo ya no se encuentra actualizado, además existen problemas de compatibilidad con las versiones actuales de Ubuntu.

POX esta creado a partir de NOX , por lo que su función es cumplir con los requerimientos de SDN, brindando ejemplos de aplicaciones y de su desarrollo, es necesario conocer el lenguaje de programación Python para su manipulación , el soporte del controlador esta poco desarrollado y la documentación del mismo es escasa.

Beacon posee muchos módulos que permiten el desarrollo de SDN, está escrito en el lenguaje de programación Java, para su instalación se necesitan muchas librerías y recursos para su funcionamiento, es necesario la utilización del software eclipse para administrarlo.

Floodlight está escrito en el lenguaje de programación Java, la instalación del mismo es muy sencilla, posee una interfaz web muy interactiva, permite obtener funcionalidades de ruteo y conmutación, brinda una serie de módulos, entre ellos uno esencial que es Firewall, el cual servirá para la realización de la aplicación para NAC, este controlador tiene una amplia documentación y el soporte del mismo es muy desarrollado.

5. Open vSwitch

Es un software de múltiples capas que trabaja con código abierto, permite ser utilizado como un switch virtual, el cual se encarga de reenviar tráfico entre máquinas virtuales que se encuentran en el mismo host o entre máquinas virtuales y las interfaces físicas de la red, Open vSwitch se encarga de exportarlas interfaces para manipular el estado del envío y la gestión de configuración de la red, es controlado por medio de la utilización de los protocolos OpenFlow y OVSDB. [10]

Open vSwitch es utilizado en entornos virtuales, permite el control y visibilidad para la capa de red virtual, realiza la distribución por medio de múltiples servidores físicos, es capaz de soportar varias tecnologías de virtualización como Xen/XenServer, KVM y Virtualbox, el código de Open vSwitch es portátil para funcionar en diversos sistemas operativos. [11]

6. Implementacion de la red

Para la realización del proyecto se plantea la utilización de Open vSwitch, que permite crear un puente con la interfaz física del equipo donde se instalará la red SDN, dicho puente permite la comunicación de la interfaz física con la interfaz virtual de cada una de las máquinas virtuales.

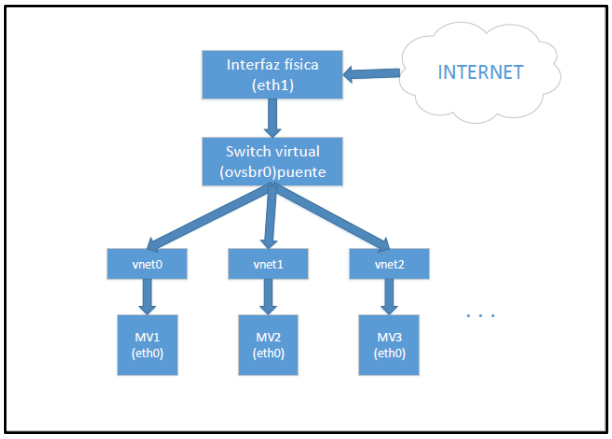


Figura 3. Diagrama para pruebas de Open vSwitch
Fuente: Paulo Colomes. (2015). *Introducción a Open vSwitch.* Obtenido de: http://www.redescisco.net/sitio/2015/11/23/introduccion-a-open-vswitch/

6.1 Configuración de la red

Para la realización de la red SDN, en el servidor físico se dispondrá del sistema operativo Ubuntu Server 14.043 LTS, en dicho servidor se instalara el Software Open vSwitch, el software de maquina virtuales KVM y el gestor Virt Manager, se crearan máquinas virtuales con cada uno de los controladores y dos clientes para la realización de pruebas para la elección del controlador más apto para la red, las máquinas virtuales cuentan con el sistema operativo Ubuntu 14.043 Desktop.

El gestor Virt Manager, configura por defecto una interfaz virbr0 que cumple de puente con la interfaz física del servidor, la cual de ver ser agregada al switch virtual, para que las máquinas virtuales puedan conectarse, por cada máquina virtual que se cree se crea de forma automática una interfaz virtual denominada vnetn donde “n” es el identificador de la máquina virtual.

Para el controlador de la red SDN se eligió Floodlight, debido a que después de realizar las configuraciones y pruebas con todos los controladores propuestos, se determinó que Floodlight es el más adecuado para la infraestructura de red

6.2 Reglas de tráfico

En la elección del software se seleccionó a Floodlight como controlador de la red, el cual tiene activado automáticamente en su configuración el módulo Forwarding, que se encarga del reenvío de datos entre dispositivos manejando flujos reactivos, de forma inmediata al encender el controlador, para la realización del proyecto se va a utilizar flujos estáticos, por lo que es necesario cambiar la configuración del módulo Forwarding.

6.3 Flujos ingresados

Los flujos estáticos permiten crear la tabla de flujo de forma manual, debido a que se insertan de acuerdo a las necesidades de los usuarios, los flujos son insertados por el controlador antes de que los paquetes de datos lleguen, una vez que los paquetes llegan al switch no son enviados al controlador para ser evaluados, debido a que coinciden con el flujo insertado.

Para la red SDN es necesario que se ingresen dos flujos primordiales, uno que permita el tráfico ARP y otro para la conexión de cada cliente a través del puerto, para el flujo de ARP el campo ether-type, específica el protocolo ARP en hexadecimal y en el campo action se realiza un flooding del flujo por los puertos.

Los flujos para la comunicación entre clientes, es necesario definir dos flujos uno en cada dirección, es decir que son bidireccionales, en ellos se incluye la dirección IP de cada uno de los host, por lo que se debe ingresar dos flujos para cada par de host de la red.

7. Desarrollo de aplicación NAC

La aplicación para NAC que se desarrollara para el presente proyecto está basada en el modelo cliente-servidor, por medio de la cual se controla los dispositivos que acceden a la red, utilizando reglas en las que si el dispositivo cuenta con seguridad se utiliza la acción *ALLOW* y si no posee ninguna seguridad se utiliza *DENY*.

La aplicación *Cliente* se encarga de comunicar un dispositivo que accede a la red con el servidor, dicha comunicación la realiza por medio de un puerto TCP, la aplicación permite definir si un equipo tiene establecido algún mecanismo de seguridad, en la aplicación desarrollada para el proyecto se detectará un archivo que permita verificar las políticas de seguridad.

La aplicación *ServidorNAC* está a la espera delas peticiones realizadas por el *Cliente*, cuando se realiza la conexión, dicho servidor envía las reglas al controlador, para permitir o denegar el acceso a un dispositivo.

8. Topología

Para el desarrollo del proyecto en la parte de hardware se cuenta con un equipo servidor, la infraestructura sobre la cual se desarrollara la aplicación es la que se muestra en la Figura 4.

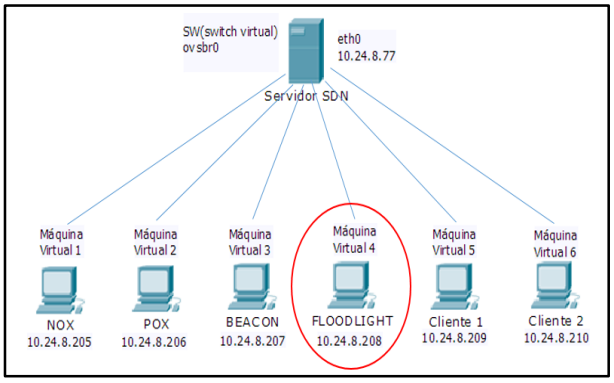


Figura 4. Infraestructura de red

Fuente: Criterio de diseño

Para la parte del software se utiliza Eclipse como plataforma y Avior para el servidor, tomando en cuenta los requisitos de la aplicación NAC.

9. Conclusiones

En este proyecto se realizó la implementación de una red SDN, para la cual se utilizó una topología constituida por switch virtual, el controlador y los clientes, por medio del manejo de flujos estáticos la persona encarga de la administración de red puede controlar el tráfico de datos, tomando en cuenta las necesidades de la red, además se desarrolló la aplicación NAC, con la cual se gestiona el acceso a la red por medio de reglas de tráfico, principalmente ARP y TCP.

En el presente proyecto se creó una SDN, en la que se implementó una solución por software, que ofrece una virtualización económica, demostrando que este tipo de implementación es una alternativa para espacios en los que no se puede realizar una SDN basada en hardware.

Para la implementación del controlador se tomó en cuenta 4 posibles opciones: NOX, POX, Beacon y Floodlight , en la elección de uno de ellos para la red SDN se verificó que no existe información necesaria de algunos de los mismos, por lo que la elección del más idóneo no resulta fácil, sin embrago la realización de pruebas con cada controlador permitió que se establezcan ciertos criterios, por ejemplo POX es un controlador muy acertado para redes SDN, debido a que posee muchos módulos, pero maneja un lenguaje poco conocido, Python el cual resulto en un inconveniente, generando que se explore otros controladores.

NOX es el primero que se desarrolló para la creación de SDN, este se tomó como base para el desarrollo de POX, por lo que se ha dejado de lado a NOX, estancando su avance, debido a esta limitación se dice que no es el más idóneo para el proyecto, además de que presenta problemas de compatibilidad con la versión de sistema operativo Ubuntu 14.04

Floodlight al igual que Beacon están desarrollados en Java, con la diferencia de que en Floodlight el número de paquetes es menor que en Beacon, debido a que este último necesita de más librerías y paquetes para su trabajo ,exigiendo a los programadores mayor conocimiento en el lenguaje Java, resulta más sencillo el manejo de Floodlight .

StaticFlowEntryPusher y Firewall son módulos de Floodlight, los cuales se emplearon para el desarrollo de la aplicación NAC y para la culminación de la red SDN, llegándose a establecer que Floodlight posee muchos módulos con los que se pueden realizar diversas pruebas, para que el switch se comporte como uno de capa2, capa 3 o incluso hub, es necesario modificar una línea de código lo que implican un manejo sencillo de los módulos.

Floodlight se eligió como el controlador idóneo para la red SDN debido a que está escrito en el lenguaje Java y posee una API la cual maneja JSON/REST, por medio de misma es posible utilizar diversas acciones que permitan determinar una política de seguridad por medio de la inserción de reglas de flujo.

En las redes tradicionales el acceso de un equipo a la red se lo realiza por medio de un administrador, en le proyecto se implementó una aplicación NAC para que cumpla con la función del administrador y determine si un equipo se conecta o no a la red, permitiendo que la red tenga sus propias características.

El desarrollo de la aplicación NAC en la red es uno de los objetivos principales del proyecto, ya que permite controlar el acceso y dar opciones de seguridad a la misma, se demuestra que es posible crear y establecer diversos tipos de aplicaciones para mejorar el servicio, ingeniería de tráfico y movilidad, y de esta forma cubrir los necesidades y requisitos de los usuarios.

Agradecimientos

Extiendo un reconocimiento merecido de gratitud a las autoridades de la Facultad de Ingeniería en Ciencias Aplicadas de la Universidad Técnica del Norte por haber facilitado el servidor en el que se implemento este proyecto de investigación, asimismo el agradecimiento sincero al Ing. MAYA EDGAR, responsable general del Proyecto Cloud-UTN, por el continuo apoyo y respaldo brindando al trabajo desempeñado; y de manera especial al Ing. VÁSQUEZ CARLOS, supervisor de este trabajo en específico, por su dedicación, paciencia y experiencia compartida en cada una de las etapas de desarrollo y revisión del mismo.

Referencias Bibliográficas

[1] Velez, J. (2012, Junio 1). *julyvezwordpress.* Retrieved from https://julyvelez.wordpress.com/2012/06/01/ventajas-y-desventajas-de-la-tecnologia-en-la-sociedad/

[2] MorilloFuentala, D. G. (2014, Mayo). Implementacion de un prototipode una red definida por software (SDN), empleando una solucion basada ensoftware. Quito, Pichincha, Ecuador.

[3] Millan, R. (2015). *ramonmillan.* Retrieved from http://www.ramonmillan.com/tutoriales/sdnredesinteligentes.php#referencias

[4] *opennetworking*. (2015). Retrieved from https://www.opennetworking.org/sdn-resources/onf-specifications/OpenFlow

[5] *noxrepo.org*. (2014). Retrieved from http://www.noxrepo.org/nox/about-nox/

[6] *openflow.stanford*. (2015, Marzo 5). Retrieved from https://openflow.stanford.edu/display/ONL/POX+Wiki#POXWiki-ComponentsinPOX

[7] MorilloFuentala, D. G. (2014, Mayo). Implementacion de un prototipode una red definida por software (SDN), empleando una solucion basada ensoftware. Quito, Pichincha, Ecuador.

[8] Erickson, D. (2013). *openflow.stanford.* Retrieved from https://openflow.stanford.edu/display/Beacon/Benchmarking

[9] *projectfloodlight.* (2014). Retrieved from http://www.projectfloodlight.org/

[10] *git.openvswitch.* (2012). Retrieved from http://git.openvswitch.org/cgi-bin/gitweb.cgi?p=openvswitch;a=blob_plain;f=FAQ;hb=HEAD

[11] *github.* (2014, Diciembre 2). Retrieved from https://github.com/openvswitch/ovs/blob/master/README.md

Sobre los Autores...

Johanna L. LEYTON P. Nació en Tulcán, Provincia del Carchi el 18 de septirmbre de 1991. Realizó sus estudios primarios en la Escuela "María Angélica Idrobo". Los estudios secundarios los realizó en la Unidad Eductiva "Sagrado Corazòn de Jesús Hmns. Bethlemitas", donde finalizó en el año 2009, obteniendo el título de Bachiller en Ciencias Especialización Físico Matemático. Actualmente, está realizando su proceso de titulación en Ingeniería en Electrónica y Redes de Comunicación, Universidad Técnica del Norte – Ecuador.

Carlos A. VÁSQUEZ A. Nació en Quito - Ecuador el 19 de Septiembre de 1981. Ingeniero en Electrónica y Telecomunicaciones, Escuela Politécnica Nacional en 2008. Actualmente es docente de la Carrera de Ingeniería en Electrónica y Redes de Comunicación en la Universidad Técnica del Norte, Ibarra-Ecuador, y es egresado de la Maestría en Redes de Comunicación, Pontificia Universidad Católica del Ecuador, Quito- Ecuador.

Design of a Clear-cut Network by Software employing a solution based in Software, for the Infrastructure of Cloud of the Faculty of Engineering andn Sciences Applied (FICA)

Authoris – Johanna Lisseth LEYTON PORTILLA, Ing. Carlos Alberto VÁSQUEZ AYALA

Faculty of Engineering in Sciences Applied, north Technical University, Avenue 17 of Julio 5-21 and José María Córdova, Ibarra, Imbabura

jlleytonp@utn.edu.ec, cavasquez@utn.edu.ec

Summary. *The present project has like purpose present the design of a Clear-cut Network by Softwarand (SDN), like a solution to the limitations of the current networks and the tendencias of traffic, making a study concerning the architecture of the networks SDN taking in account the separation of the planes of data and of control, in addition to the investigation of the protocol Openflow in which they are bastogive said networks.*

In the infrastructure of the networks SDN presents a controller, for which takes four alternatives: NOX, POX, Beacon and Floodlight, with which will make n proofs to determine the most optimum and adapted in the implementation of the network, another element of the infrastructure is a switch virtual, the adpatación of the same is by means of the software Open vSwitch, at the same time define the rules of flow.

In the implementation of the network used the software KVM and virtual manager, which allows to create virtual machines, in addition to that all the implementation made on the operating system Ubuntu, finally develops an application pairto the control of access to the half (NAC).

Key words

Clear-cut networks By Software (SDN), OpenFlow, NOX, POX; Beacon, Floodlight, Open vSwitch, NAC.

Resumen. *El presente proyecto tiene como finalidad presentar el diseño de una Red Definida por Software (SDN), como una solución a las limitaciones de las redes actuales y las tendencias de tráfico, realizando un estudio referente a la arquitectura de las redes SDN tomando en cuenta la separación de los planos de datos y de control, además de la investigación del protocolo Openflow en el cual están basadas dichas redes.*

En la infraestructura de las redes SDN se presenta un controlador, para lo cual se toma cuatro alternativas: NOX, POX, Beacon y Floodlight, con los cuales se realizarán pruebas para determinar el más óptimo y adecuado en la implementación de la red, otro elemento de la infraestructura es un switch virtual, la adpatación del mismo es por medio del software Open vSwitch, al mismo tiempo se definen las reglas de flujo.

En la implementación de la red se utilizara el software KVM y virtual manager, el cual permite crear máquinas virtuales, además de que toda la implementación se realizara sobre el sistema Operativo Ubuntu, finalmente se desarrolla una aplicación para el control de acceso al medio (NAC).

Palabras claves

Software Defined Networks (SDN), OpenFlow, NOX, POX; Beacon, Floodlight, Open vSwitch, NAC.

10. Introduction to SDN

The companies and users of a network in the actuality present a lot of problems in his traditional networks, since they do not fulfil with the requests requested, by what the Clear-cut Networks by Software (SDN), present like a solution standardised to cover the need of the companies and the users of network, giving step to the transformation of the architecture of the traditional networks.

Use the traditional networks to satisfy the requests of telecommunications, generated by the users results unlikely; the companies to world-wide level use the departments of IT to take advantage of to the maximum his networks, like a solution passenger, due to the fact that the existent networks are not designed to cover the current requests of the users, companies and providers. [1]

With the architecture of network SDN can create networks that cover the requests of the users and companies in the actuality, due to the fact that it allow the separation of the plane of control and the plane of data, so that the people commissioned of the administration of the network, make it of form centralised. The manipulation of the plane of control and of the plane of data makes separately, for the plane of control uses the protocol OpenFlow, by means of teams controllers and the plane of data is operated by the teams of connection.

11. Architecture of a SDN

The devices of connectivity in a network are integrated by two important components: the plane of control and the plane of data, the plane of data is the attendant of the transmission of the data and the plane of control determines the way circulation of the informacion to arrive to his destination.A Clear-cut Network by Software (SDN) appears like a solution for the demands and requests so much of the plane of control as of the plane of data. [2]

In the figure 1 represents the architecture of a Clear-cut Network by Software.

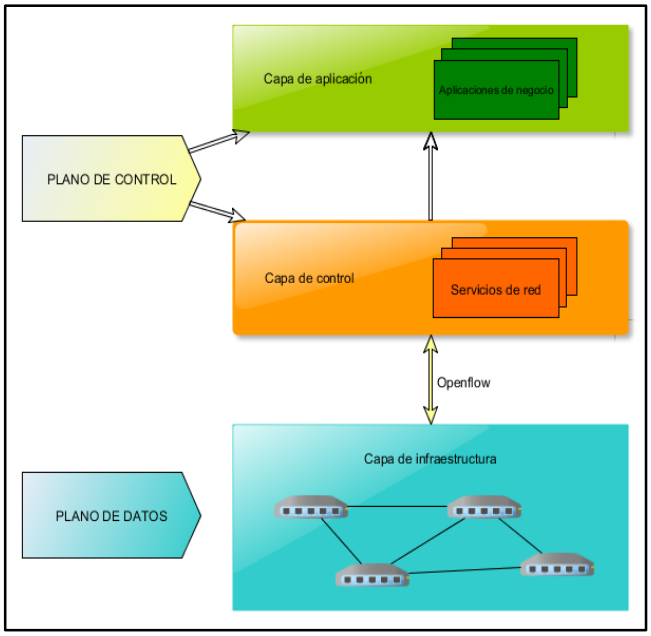


Figure 3. Architecture SDN
Source: SDxCentral. (2012). What's Software Defined Networking (SDN)? Definition. Obtained of: https://goo.gl/i94xlg

Layer of infrastructure

It finds conformed by the elements of the network that commission of the conmutación and the enrutamiento of the packages of data, offering access through an API, as it is OpenFlow.

Layer of control

The function ofcont role is centralised, allows the utilisation of the capacities of the network, the controller of a SDN has the control of a group of resources of the plane of control, allowing the routing of the traffic of network, commissions of the conmutación of data and to choose the optimum way pploughs the transmission, thec ontrolador manages a rank of resources of the plane of data, facilitating the configuration of the same.

Layer of application

It consists of the applications requested by the final users, by means of the API of SDN which allow the utilisation of services of communication, the layer of application attains that the simple configurations and gives step to the management of new services, forms of access and improvement of the network. [3]

OpenFlow

OpenFlow Is an interface used in the networks SDN, allows the communication between the planes of data and of control of the teams that form the network, offers the ease to access to the plane of control and manipulate it of direct form, already was in physical or virtual devices.

12. Protocol OpenFlow

OpenFlow Is a recent protocol regarding communications, which allows to determine the correct enrutamiento of the packages in a network, using OpenFlow a network can be manipulated like an alone body, centralising the decisions of mobility of packages, in OpenFlow a controller commissions of the determine the routing of high level. [4]

In a network SDN the protocol OpenFlow has to be operative in the teams that form the network and in the controller of the same, to distinguish the traffic of the network the protocol uses flows, that define like the trafico travels through the devices of network taking in account parameters of use and applications, basing in rules predetermined in the controller.

A Switch OpenFlow is a device that sends packages of a network SDN which operates with the utilisation of tables of flow and actions of each entrance of the flow, being possible to divide the traffic of the network, the Switch OpenFlow allows the adoption of new protocols, models of security and designs of direccionamiento, the traffic of the network does not see prejudiced due to the fact that it treats of isolated form and processes of equal way that in a traditional network, finds composed by 3 main parameters:

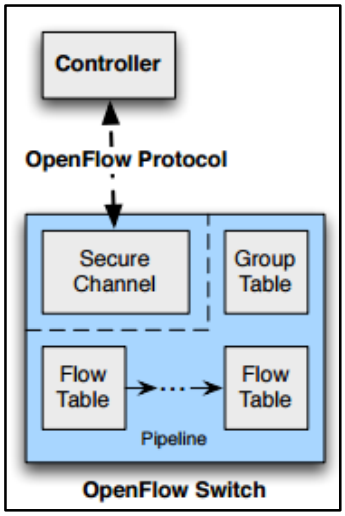


Figure 4. Components of switch OpenFlow
Source: OpenFlow. (2011). *OpenFlow Switch Specification*. Obtained of: http://archive.openflow.org/documents/openflow-spec-v1.1.0.pdf

13. Controllers

13.1 NOX

In OpenFlow, NOX is the original controller, was developed by Nicira, NOX is basically a platform that allows to create applications for the control of the network, the most current version of NOX offers a framework lighter and of greater performance, his development does in the language C++ for Linux.

Operation

NOX Works to base of tables of flow, the handle of said tables makes it by means of an interface of programming in which they can configure applications that provide greater control of access to the network ,inside the events of the same, by means of the manipulation of the flows of network can control the decisions of reenvío and the traffic of the network of a scalable form; when it detects a new entrance of flow, said flow is envoy to the controller and there transfers it to him to the suitable applications, besides the correct handle of the flows allows the modification and analysis of packages and the obtain statistics. [5]

13.2 POX

NOX Was taken like base for the development of POX, is a controller of OpenFlow that finds in constant development, created to cover the needs of the SDN with the language Python for other operating systems.The main characteristics of POX are:

- His interface is developed in Python.
- It developed for Windows, Mac YOU and Windows.
- The graphic interface and the tools to visualise are resembled the ones of NOX
- The elements show the route of access, the topology, between others. [6]

13.3 Beacon

Beacon Is a very effective and fast controller, multiplataforma and modulate, is developed in the language Java, besides caters packages with characteristics advanced that allow to determine the best way and the shortest, the detection of the topology and an interface web, this controller is based in events and threads that are subprocesos scheduled for a system. [7]

Beacon Works by means of the utilisation of modules called “budles”, which possess metadatos, classes of Java and other archives, the budles of Beacon work in group, the characteristics of the budles are:

- They can matter and export packages Java.
- They offer an interface web of management.
- They bear OpenFlow 1.0,
- Compatibility with packages TCP/UDP and IP
- Operations based in events and threads.
- They offer modules for the creation of applications.
- Can modify packages in the execution without damaging other packages.
- They handle Proxies ARP and DHCP. [8]

13.4 Floodlight

This controller was developed in Java with a licence Apache, can resist so much switches virtual like physicists, is composed by a main module, which commissions to identify the packages and assign events, also form it secondary modules registered with the modulate main, commissioned to operate and attach entrances to the tables of flows.

The operation of Floodlight bases in modules between which exist two types:

Modules of controller

In the modules of the controller find the characteristics that use in the applications continuously, like the disclosure of topology and the events, the link with the controller, an interface web of user, and the resources shared of the network.

Modules of application

These modules commission to enter determinate functions, as the ones of a switch layer 2, firewall or links gone down. [9]

13.5 Comparative between controllers

Each one of the controllers integrates of simple form with the Open vSwitch, once that it cleared the function of the interfaces. It fits to mention that the software Open vSwitch possesses a big support due to the fact that it is very used in the infrastructure of SDN.

NOX Is simple to install, due to the fact that it is the first controller that developed for the networks SDN, does not offer all the components required for the applications, the support of the same no longer finds up to date, besides exist problems of compatibility with the current versions of Ubuntu.

POX This created from NOX , by what his function is to fulfil with the requests of SDN, offering examples of applications and of his development, is necessary to know the programming language Python for his manipulation , the support of the controller this little developed and the documentation of the same is scarce.

Beacon Possesses a lot of modules that allow the development of SDN, is written in the programming language Java, for his installation need a lot of bookshops and resources for his operation, is necessary the utilisation of the software eclipse to administer it.

Floodlight Is written in the programming language Java, the installation of the same is very simple, possesses an interface web very interactive, allows to obtain functionalities of ruteo and conmutación, offers a series of modules, between them one essential that is Firewall, which will serve for the realisation of the application for NAC, this controller has a wide documentation and the support of the same is very developed.

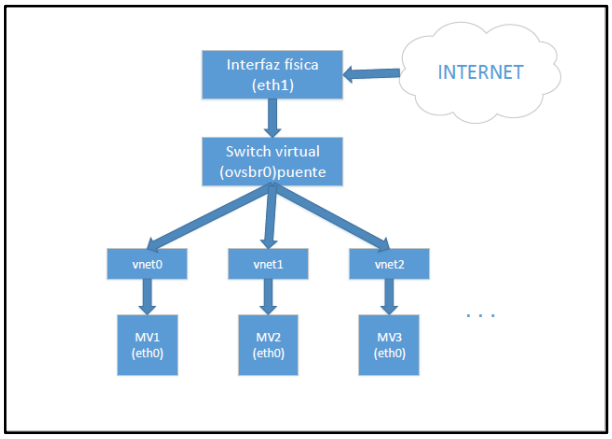
14. Open vSwitch

It is a software of multiple layers that works with open source, allow to be used like a switch virtual, which commissions of reenviar traffic between virtual machines that find in the same host or between virtual machines and the physical interfaces of the network, Open vSwitch commissions to export them interfaces to manipulate the state of the sending and the management of configuration of the network, is controlled by means of the utilisation of the protocols OpenFlow and OVSDB. [10]

Open vSwitch Is used in virtual surroundings, allows the control and visibility for the layer of virtual network, makes the distribution by means of multiple physical servers, is able to bear several technologies of virtualización like Xen/XenServer, KVM and Virtualbox, the code of Open vSwitch is portable to work in diverse operating systems. [11]

15. Implementacion Of the network

For the realisation of the project poses the utilisation of Open vSwitch, that allows to create a bridge with the physical interface of the team where will install the network SDN, said bridge allows the communication of the physical interface with the virtual interface of each one of the virtual machines.



It appears 3. Diagram for proofs of Open vSwitch
Source: Paulo Colomes. (2015). *Introduction to Open vSwitch.* Obtained of: http://www.redescisco.net/sitio/2015/11/23/introduccion-a-open-vswitch/

15.1 Configuration of the network

For the realisation of the network SDN, in the physical server will have of the operating system Ubuntu Server 14.043 LTS, in said server installed the Software Open vSwitch, the software of scheme virtual KVM and the managing Virt Manager, created virtual machines with each one of the controllers and two customers for the realisation of proofs for the election of the aptest controller for the network, the virtual machines have the operating system Ubuntu 14.043 Desktop.

The managing Virt Manager, configures by defect an interface virbr0 that fulfils of bridge with the physical interface of the server, which to see be added to the switch virtual, so that the virtual machines can connect, by each virtual machine that believes creates of automatic form a virtual interface designated vnetn where “n” is the identifier of the virtual machine.

For the controller of the network SDN chose Floodlight, due to the fact that after making the configurations and proofs with all the controllers proposed, determined that Floodlight is the most adapted for the infrastructure of network

15.2 Rules of traffic

In the election of the software selected to Floodlight like controller of the network, which has activated automatically in his configuration the module Forwarding, that commissions of the reenvío of data between devices

handling reactive flows, of immediate form when lighting the controller, for the realisation of the project goes to use static flows, by what is necessary to change the configuration of the module Forwarding.

15.3 Flows ingresados

The static flows allow to create the table of flow of manual form, due to the fact that they insert of agreement to the needs of the users, the flows are inserted by the controller before the packages of data arrive, once that the packages arrive to the switch are not envoys to the controller to be evaluated, due to the fact that they coincide with the flow inserted.

For the network SDN is necessary that ingresen two paramount flows, one that allow the traffic ARP and another for the connection of each customer through the port, for the flow of ARP the field ether-type, specific the protocol ARP in hexadecimal and in the field action makes a flooding of the flow by the ports.

The flows for the communication between customers, is necessary to define two flows one in each direction, that is to say that they are bidirectional, in them includes the direction IP of each one of the host, by what has to ingresar two flows for each pair of host of the network.

16. Development of application NAC

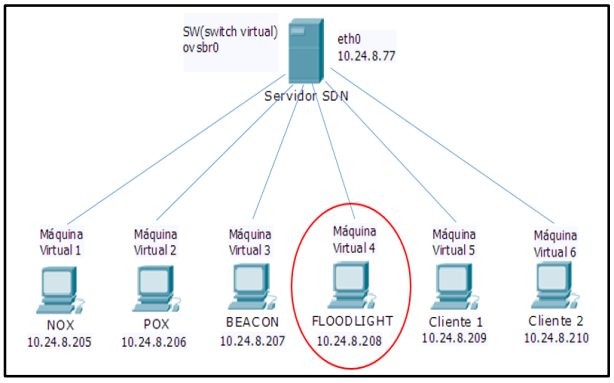
The application for NAC that developed for the present project is based in the model customer-server, by means of which controls the devices that access to the network, using rules in which if the device has security uses the action *ALLOW* and if it does not possess any security uses *DENY*.

The application *Customer* commissions to communicate a device that accesses to the network with the server, said communication makes it by means of a port TCP, the application allows to define if a team has established some mechanism of security, in the application developed for the project will detect an archive that allow to verify the politics of security.

The application *ServidorNAC* is expecting delas requests made by *the Customer*, when it makes the connection, said server sends the rules to the controller, to allow or denegar the access to a device.

17. Topology

For the development of the project in the part of hardware has a teamrvidor, lto infrastructure on which developed the application is the one who shows in the Figure 4.



It appears 4. Infrastructure of network

Source: Criterion of design

For the part of the software uses Eclipse like platform and Avior for the server, taking in account the requirements of the application NAC.

18. Conclusions

In this project made the implementation of a network SDN, for which used a topology constituted by switch virtual, the controller and the customers, by means of the handle of static flows the person commissions of the administration of network can control the traffic of data, taking in account the needs of the network, besides developed the application NAC, with which manages the access to the network by means of rules of traffic, mainly ARP and TCP.

In the present project created a SDN, in which it implemented a solution by software, that offers a virtualización economic, showing that this type of implementation is an alternative for spaces in which it can not make a SDN based in hardware.

For the implementation of the controller took in account 4 possible options: NOX, POX, Beacon and Floodlight , in the election of one of them for the network SDN verified that it does not exist necessary information of some of the same, by what the election of the most ideal does not result easy, without embrago the realisation of proofs with each controller allowed that they establish some criteria, for example POX is a very hit controller for networks SDN, due to the fact that it possesses a lot of modules, but handles a little known language, Python which result in a problem, generating that it explore other controllers.

NOX Is the first that developed for the creation of SDN, this took like base for the development of POX, by what has shelved to NOX, stagnating his advance, because of this limitation says that it is not the most ideal for the project, in addition to that presents problems of compatibility with the version of operating system Ubuntu 14.04

Floodlight To the equal that Beacon are developed in Java, with the difference that in Floodlight the number of packages is lower that in Beacon, due to the fact that this last needs of more bookshops and packages for his work ,demanding to the greater programmers knowledge in the language Java, results simpler the handle of Floodlight .

StaticFlowEntryPusher And Firewall are modules of Floodlight, which employed for the development of the application NAC and for the culmination of the network SDN, arriving to establish that Floodlight possesses a lot of modules with which can make diverse proofs, so that the switch comport like one of layer2, layer 3 or even hub, is necessary to modify a line of code what involve a handle simple of the modules.

Floodlight Chose like the ideal controller for the network SDN due to the fact that it is written in the language Java and possesses an API which handles JSON/REST, by means of same is possible to use diverse actions that allow to determine a politics of security by means of the insertion of rules of flow.

In the traditional networks the access of a team to the network makes it to him by means of an administrator, in project him implemented an application NAC so that it fulfil with the function of the administrator and determine if a team connects or no to the network, allowing that the network have his own characteristic.

The development of the application NAC in the network is one of the main aims of the project, since it allows to control the access and give options of security to the same, shows that it is possible to create and establish diverse types of applications to improve the service, engineering of traffic and mobility, and of this form cover the needs and requirements of the users.

Gratitudes

I extend a recognition deserved of gratitude to the authorities of the Faculty of Engineering in Sciences Applied of the north Technical University for having facilitated the server in which I implement this project of investigation, tosimismo the sincere gratitude to the Ing. MAYA EDGAR, general manager of the Project Cloud-UTN, by the continuous support and backrest offering to the work exerted; and of special way to the Ing. VÁSQUEZ CARLOS, supervisor of this work in specific, by his dedication, patience and experience shared in each one of the stages of development and review of the same.

Bibliographic references

[1] Velez, J. (2012, June 1). *julyvezwordpress.* Retrieved from https://julyvelez.wordpress.com/2012/06/01/ventajas-y-desventajas-de-la-tecnologia-en-la-sociedad/

[2] MorilloFuentala, D. G. (2014, May). Implementacion Of a prototipode a clear-cut network by software (SDN), employing a solucion based ensoftware. I remove, Pichincha, Ecuador.

[3] Millan, R. (2015). *ramonmillan.* Retrieved from http://www.ramonmillan.com/tutoriales/sdnredesinteligentes.php# References

[4] *opennetworking.* (2015). Retrieved from https://www.opennetworking.org/sdn-resources/onf-specifications/openflow

[5] *noxrepo.org.* (2014). Retrieved from http://www.noxrepo.org/nox/about-nox/

[6] *openflow.stanford.* (2015, March 5). Retrieved from https://openflow.stanford.edu/display/onl/pox+Wiki#POXWiki-ComponentsinPOX

[7] MorilloFuentala, D. G. (2014, May). Implementacion Of a prototipode a clear-cut network by software (SDN), employing a solucion based ensoftware. I remove, Pichincha, Ecuador.

[8] Erickson, D. (2013). *openflow.stanford.* Retrieved from https://openflow.stanford.edu/display/beacon/benchmarking

[9] *projectfloodlight.* (2014). Retrieved from http://www.projectfloodlight.org/

[10] *git.openvswitch.* (2012). Retrieved from http://git.openvswitch.org/cgi-bin/gitweb.cgi?p=openvswitch;To=blob_plain;f=FAQ;hb=HEAD

[11] *github.* (2014, December 2). Retrieved from https://github.com/openvswitch/ovs/blob/master/readme.md

SAct the Authoris...

Johanna L. LEYTON P. N Tució in Tulcán,P rovincia of the Carchi the 18 of septirmbre of 1991. It made his primary studies in the School “María Angélica Idrobo”. The secondary studies made them in l to Unit Eductiva “Holy Corazòn of Jesús Hmns. Bethlemitas”, where finalised in the year 2009 , obteniendo the title of Bachiller in Sciences Physical Specialisation Mathematician. At present, it is making his process of degree in Ingeniería in Electronics and Networks ofC omunicación, north Technical University – Ecuador.

Carlos To. VÁSQUEZ To. It was born in I Remove - Ecuador on 19 September 1981. Engineer in Electronics and Telecommunications, National Polytechnical School in 2008. At present it is educational of the Career of Engineering in Electronics and Networks of Communication in the north Technical University, Ibarra-Ecuador, and is graduate of the Mastery in Networks of Communication, Pontificia Catholic University of the Ecuador, Remove- Ecuador.