IMPLEMENTACIÓN DE UN ALGORITMO QUE ASIGNA DINÁMICAMENTE RECURSOS PARA LA VIRTUALIZACIÓN DE REDES EN UNA RED DEFINIDA POR SOFTWARE.

DANIEL ARTURO OSORIO MONTOYA
JOSÉ JHOVANNY OSORIO MONTOYA
JHONNY ALEJANDRO ORREGO ACOSTA

UNIVERSIDAD DE CATÓLICA DE PERERIA
FACULTAD DE CIENCIAS BÁSICA E INGENIERÍA
INGENIERIA DE SISTEMAS Y TELECOMUNICACIONES
PEREIRA
2017

IMPLEMENTACIÓN DE UN ALGORITMO QUE ASIGNA DINÁMICAMENTE RECURSOS PARA LA VIRTUALIZACIÓN DE REDES EN UNA RED DEFINIDA POR SOFTWARE.

DANIEL ARTURO OSORIO MONTOYA
JOSÉ JHOVANNY OSORIO MONTOYA
JHONNY ALEJANDRO ORREGO ACOSTA

Trabajo de Grado presentado como opción para optar al título de:
Ingeniero de Sistemas y Telecomunicaciones

Asesor:
Ingeniero Nestor Alzate Mejía

UNIVERSIDAD DE CATÓLICA DE PEREIRA
FACULTAD DE CIENCIAS E INGENIERÍA
PROGRAMA SISTEMAS Y TELECOMUNICACIONES
PEREIRA 2017

PÁGINA DE ACEPTACIÓN

_____
_____
_____
_____
_____
_____
_____

_____
<NOMBRE COMPLETO>
JURADO

_____
<NOMBRE COMPLETO>
JURADO

_____
<NOMBRE COMPLETO>
JURADO

Pereira, 15 de junio de 2017

CONTENIDO

Pág.

LISTA DE IMÁGENES

Pág.

LISTA DE TABLAS

Pág.

RESUMEN

La actual arquitectura de red es uno de los principales obstáculos para los nuevos modelos que se están proponiendo en la prestación de servicios. Las redes definidas por software (SDN) se presentan como una alternativa de innovación. Mediante esta nueva arquitectura se pretende la separación del plano de control del plano de datos, tanto para los switches como para los routers y que toda la parte de control y enrutamiento se realice a través de software con la ayuda de herramientas como hipervisores y controladores, los cuales permiten la creación de porciones de red que funcionan de manera autónoma e independiente.
El incrustamiento de redes virtuales es uno de los principales retos y objetos de estudio en las redes definidas por software, este consiste en evitar que las nuevas peticiones de incrustamiento sean rechazadas, ya sea porque los recursos físicos de CPU y ancho de banda estén al límite o porque la red se encuentre fragmentada, es decir, una mala distribución de los espacios disponibles, lo que generaría una reducción en las ganancias percibidas por los prestadores y operadores del servicio.
Nuestra propuesta es la implementación de un algoritmo que asigna dinámicamente recursos, para la virtualización de redes en una red definida por software.
Las pruebas y evaluaciones del algoritmo se ejecutarán en conjunto con diversas herramientas como hipervisores y controladores en la plataforma GENI la cual es un banco de pruebas que permite la asignación de recursos dependiendo la necesidad del proyecto.

PALABRAS CLAVE: Sdn, algoritmo, hipervisores, Geni, vne.

ABSTRACT

The current network architecture is one of the main obstacles for the new models that are being proposed in the provision of services. The sdn software-defined networks are presented as an alternative of innovation. This new architecture aims at separating the control plane from the data plane for both switches and routers and that all the control and routing part is done through software with the help of tools such as hypervisors and controllers, which allow the creation of network portions that operate autonomously and independently.

The embedded virtual networks are one of the main challenges and case of study in software-defined networks, the goal it's to avoid that new requests for scaling are rejected, either because the physical resources CPU and bandwidth are at the limit or Because the network is fragmented, that is to say a poor distribution of the available spaces, which would generate losses in the gains perceived by the service providers and operators.

Our proposal is the implementation of an algorithm that dynamically allocates resources, for the virtualization of networks in a network defined by software.

The tests and evaluations of the algorithm will be executed in conjunction with diverse tools like hypervisors and controllers in the platform GENI which is a test bench that allows the allocation of resources depending on the necessity of the project.

Keywords: algorithm, hypervisors, Geni, vne.

INTRODUCCIÓN

Internet surge como un proyecto creado a principios de los años 70 por la agencia para proyectos avanzados del departamento de defensa norteamericano DARPA, su objetivo principal era el de conectar los equipos del gobierno de los Estados Unidos de Norte América.

Fue entonces cuando diferentes universidades y centros de investigación se unen y la red pasa a tener un carácter más investigativo y de desarrollo.

Durante de la década de los 90´s pasa de ser una red netamente académica, para abrirse paso al gran público, motivada por diversos factores [1].

- Aparición de computadores personales (pc), tanto para empresas como para hogares.
- Mejora de las redes de telecomunicaciones, dando como resultado mejor velocidad y calidad.
- Se establecen estándares en la red, la aparición de organizaciones como el Grupo de Trabajo de Ingeniería de Internet IETF.
- Nuevos servicios multimedia e interfaces intuitivas como la word wide web WWW.

Como consecuencia se pretenden implementar diversas características en la red, pero debido a grandes obstáculos como por ejemplo el lento proceso de estandarización, los investigadores son obligados a realizar pruebas en pequeños laboratorios para después trasladar estas nuevas ideas a la IETF para su implementación, si desde luego lograban ser financiadas.[2]

El periodo de maduración e implementación de un protocolo de red es generalmente lento, un ejemplo de ello es IPv6.

A comienzos del año 2000 las redes continuaban creciendo a un ritmo bastante acelerado, lo cual se comienza a traducir en una inminente saturación, debido a que el diseño original de la red no daba pie agrandes innovaciones, se hace indispensable entonces pensar en un despliegue de servicios más dinámico, flexible, programable y con la capacidad de atender nuevos servicios sin interrumpir los que se estuviesen ejecutando.

Las redes definidas por software (SDN), surgen como una posible solución al problema de osificación de la red, al proponer el desacoplamiento da la capa de datos de la capa de control, y que toda la parte de control se delegue a un solo elemento llamado controlador, encargado de decidir el reenvió de paquetes y de construir las tablas de enrutamiento. De esta manera se pretende unificar los diferentes elementos de la red y a través del protocolo Openflow lograr la

comunicación entre estos, permitiendo que la red se transforme en un sistema abierto [3].

OpenFlow

Para el año 2008 investigadores de la universidad de Stanford Nick McKeown et al. Idearon el protocolo OpenFlow, el cual es un estándar abierto que permite el uso de las capacidades de los dispositivos de la red que lo implementen, sin la necesidad de conocer el funcionamiento interno de estos. Esto permite establecer reglas de gestión de flujos sin utilizar las configuraciones propietarias de cada fabricante.

Este protocolo ha evolucionado de manera incremental desde sus principios en 2006 con la aparición de Ethane, a cargo de Martin Casado [4] hasta su última versión, la 1.5, lanzada en diciembre de 2014 a cargo de la ONF: Open Networking Foundation [5].

En la actualidad Openflow posee características bastante avanzadas, como lo son, el control de distintos espacios de red en diferentes tablas de flujos sincronizadas, ofrece también la posibilidad de filtrado detallado, por ejemplo, flags de TCP, uso de puertos de fibra óptica, entre otras.

La base fundamental de Openflow como ya se mencionaba anteriormente radica en la separación del plano de datos del plano de control. En los dispositivos actuales de red, estas dos funciones residen en cada uno de ellos, mientras que con el protocolo Openflow, las reglas de reenvió (plano de control), son realizadas por el controlador, logrando de esta manera centralizar la gestión de tráfico de la red.

Gracias a la arquitectura de las redes definidas por software SDN, es posible realizar la virtualización de redes, permitiendo a los proveedores de servicios, crear porciones de red virtuales sobre una red física, lo que se traduce en una mejor optimización de los recursos actuales y por ende un crecimiento en las ganancias [6].

Uno de los principales problemas para la virtualización de redes, se conoce como mapeo o incrustación de redes virtuales, este consiste en evitar que las nuevas peticiones de incrustamiento sean rechazadas, ya sea porque los recursos físicos CPU y ancho de banda estén al límite o porque la red se encuentre fragmentada, es decir una mala distribución de los espacios disponibles, lo que generaría perdidas en las ganancias percibidas por los prestadores del servicio. Por esta razón, los algoritmos desarrollados para dar solución a este problema, se conocen como algoritmos VNE [7],[8].

1. ÁREA PROBLEMÁTICA

Las redes tradicionales desde sus inicios presentan un esquema basado desde el hardware y hacia el hardware; por tal motivo la administración de los diferentes equipos de la red se debe realizar por separado y de manera individual.

Al ser un diseño con un plano de control distribuido, no provee la dinámica, escalabilidad y flexibilidad que las redes actuales requieren.

Los nuevos servicios de red y su implementación resultan ser bastante complejas y difíciles de administrar, ya que el hardware necesario para su funcionamiento cuenta con su propio protocolo, lo cual genera la necesidad por parte de los administradores de la red, de analizar y estudiar los diferentes protocolos con el fin de mantener en funcionamiento este esquema distribuido; en contraposición las SDN, proponen un esquema el cual permite la separación del plano de datos del plano de control, permitiendo la centralización bajo el protocolo Openflow, el cual hace posible una abstracción simple, estandarizada y fácil de administrar, que permite también la creación de nuevas redes virtuales inteligentes con arquitecturas flexibles, escalables, programables y que pueden soportar los múltiples servicios que en la actualidad se demandan, como por ejemplo el cloud computing y la virtualizacion de las funciones de red (VNF).

2. OBJETIVOS

2.1 OBJETIVO GENERAL

Implementar un algoritmo que asigne dinámicamente recursos para la virtualización de redes en una red definida por software.

2.2 OBJETIVOS ESPECÍFICOS

- Definir el hipervisor para la configuración del algoritmo.
- Definir el controlador para la configuración del sistema.
- Implementar una red definida por software en la plataforma Geni.
- Programar el hipervisor con el algoritmo de asignación dinámica de recursos.
- Realizar pruebas de la automatización del proceso de virtualización en la plataforma Geni.

3. JUSTIFICACIÓN

La investigación acerca de la implementación de un algoritmo que asigna dinámicamente recursos, para la virtualización de redes en una red definida por software, es muy importante, ya que pretende que tanto los operadores y los proveedores de redes puedan aprovechar al máximo la tecnología con la que cuentan en la actualidad, es decir, la parte de hardware; esto se lograría descentralizando la capa de control de la capa de datos, lo que permitiría crear porciones virtuales de la red física, lo que contribuiría a que estos puedan obtener más ganancias al incrementar la reutilización de sus servicios.

4. METODOLOGÍA

4.1 TIPO DE INVESTIGACION

El tipo de investigación es experimental ya que “el criterio en la naturaleza de los objetivos se propone controlar y manipular el objeto de estudio, sometiéndolo a pruebas.”[9]

4.2 DISEÑO DE LA INVESTIGACION

El desarrollo de la investigación está directamente ligado con la consecución de los objetivos específicos ya mencionados. De esta forma el desarrollo de la investigación se encuentra dividida en 5 fases con sus correspondientes actividades tal como se muestra en la tabla 1.

Tabla 1. Fases y actividades de la investigación.

FASES	**ACTIVIDADES**
1. Definir el hipervisor para la configuración del algoritmo.	• Búsqueda y selección de bibliografía, web, repositorios universitarios, IEEE. • Preselección de los hipervisores encontrados, basados en sus características y documentación obtenida de la búsqueda bibliográfica. • Instalación del hipervisor preseleccionado en la plataforma Geni.
2. Definir el controlador para la configuración del sistema.	• Búsqueda y selección bibliográfica en diferentes fuentes, web, repositorios universitarios, IEEE. • Preselección de los controladores encontrados, basados en sus características y documentación obtenida de la búsqueda bibliográfica. • Instalación de los controladores seleccionados en la plataforma Geni. • Pruebas con los controladores instalados para determinar cuál es el más conveniente para la implementación.
3. Implementar una red virtual en la plataforma Geni.	• Implementación de la incrustación de una red virtual de forma manual en la plataforma Geni.
4. Programar el	• Configuración del hipervisor con los diferentes

hipervisor con el algoritmo de asignación dinámica de recursos.	Scripts desarrollados que se usaran en la implementación del algoritmo en la plataforma Geni.
5. Realizar pruebas de la automatización del proceso de virtualización en la plataforma Geni.	• Pruebas de funcionamiento del script con los resultados entregados por el algoritmo, así como pruebas de rendimiento y efectividad.

5. DESARROLLO DEL PROYECTO

5.1 PROCEDIMIENTO

Este proyecto forma parte de un proyecto de maestría desarrollado por el ingeniero Néstor Álzate donde se definió previamente Geni como plataforma ya que es un banco de pruebas de tecnología integral, para promover la investigación en redes rápidas y el desarrollo de aplicaciones, esta proporcionan a los investigadores una plataforma para realizar experimentos de investigación a mayor escala [10]. A su vez también proporciona espacios para realizar experimentos computacionales aislados y soporta OpenFlow, el cual es un protocolo abierto para programar tablas de flujo en switches y enrutadores [3] sean físicos o virtuales, con una serie de mensajes que permiten el monitoreo del rendimiento de la red y su comportamiento. Geni posee funciones de Software Defined Networking (SDN) para realizar investigación exhaustiva de la red [11].
Dicha plataforma posee distintos tipos de racks de servidores de diversas marcas, y switches de red robustos a disposición de los investigadores. Geni es muy adecuada para la exploración de redes a escala, con lo que se promueven las innovaciones en la ciencia de las redes de telecomunicaciones, seguridad, servicios y aplicaciones.
GENI permite entre otros:

• Obtener recursos informáticos robustos de lugares alrededor de los Estados Unidos.
• Conectar recursos informáticos mediante redes de Capa 2 en topologías que mejor se adapte a los experimentos.

• Instalar el software personalizado o incluso sistemas operativos personalizados sobre los recursos informáticos.
• Controlar los switches de red y experimentar el manejo del flujo de tráfico de los mismos.

Para poder ingresar a Geni fue necesario solicitar acceso a la plataforma por medio de un tutor o supervisor de proyecto, dicho tutor es el responsable del buen uso de los recursos, el correcto manejo de los mismos y a su vez se encarga de invitar a los estudiantes o participantes del proyecto a la plataforma. Posterior a dicha solicitud y su correcta aprobación por parte del tutor se obtienen las credenciales de acceso. A la fecha de desarrollo de este proyecto en Colombia solo 5 personas poseían acceso a Geni incluidos los presentes. Para el manejo de la asignación de recursos y uso de la plataforma se tomó como referencia los tutoriales encontrados en el wiki de la página de la plataforma [12], al desarrollar los tutoriales se logró obtener la información de cómo tener control de los recursos proporcionados por Geni, los cuales físicamente se encuentran distribuidos en diversas universidades e instituciones a los largo de los Estados Unidos. Geni ofrece variadas opciones de hardware a nivel de switches virtuales, host virtuales y diferentes topologías de red, también permite seleccionar distintos sistemas operativos de acuerdo con la necesidad que se tenga, permitiendo tener recursos altamente flexibles y manejables. La manera de realizar la solicitud para la asignación de recursos se encuentra documentada en el anexo A.

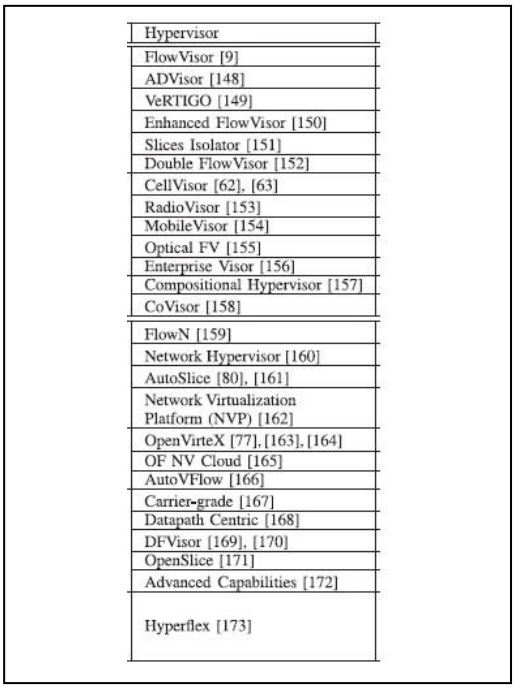
5.1.1 Fase 1. Definir el hipervisor para la configuración del algoritmo.

- **Actividad 1. Búsqueda de bibliografía, web, repositorios universitarios, IEEE.**
  Se realiza una revisión sobre la información pertinente a hipervisores en los repositorios de la IEEE, la web entre otros de los cuales se localizan 15 artículos clave, con los que se logra un panorama más claro sobre la función del mismo, ya que los hipervisores son pieza fundamental en la concepción de las redes definidas por software virtuales (vSDNs). Estos son la capa intermedia entre los controladores SDN y las respectivas redes virtuales (VN).
  Con el fin de crear redes completamente desconectadas e independientes de la red física, los administradores utilizan un supervisor de máquina virtual conocido como hipervisor, mediante el cual es posible realizar un escaneo de los recursos físicos y determinar la manera más eficiente de crear las nuevas instancias virtuales o porciones de red.
  Con el uso de hipervisores y a través del protocolo Openflow los operadores de red, proveedores de servicios y empresas pueden compartir la infraestructura con la cual cuentan [7]. Para dicho elemento, la documentación sobre su configuración es muy general, para la sintaxis o

compatibilidad, no hay información concisa y completa. Lo cual hizo más compleja la selección del hipervisor más adecuado.

- **Actividad 2. Preselección de los hipervisores encontrados.**
  En los antecedentes bibliográficos hallados encontramos la existencia de múltiples hipervisores como se ve en la imagen 1. hallada en el estudio [13]. Diversos autores han desarrollado proyectos de investigación haciendo uso de Flowvisor como es el caso de [6],[14] y como se sugieren en [15], Flowvisor es el hipervisor más adecuado a nivel académico para realizar pruebas e investigación.

Imagen 1. Lista de hipervisores



Hypervisor
FlowVisor [9]
ADVisor [148]
VeRTIGO [149]
Enhanced FlowVisor [150]
Slices Isolator [151]
Double FlowVisor [152]
CellVisor [62], [63]
RadioVisor [153]
MobileVisor [154]
Optical FV [155]
Enterprise Visor [156]
Compositional Hypervisor [157]
CoVisor [158]
FlowN [159]
Network Hypervisor [160]
AutoSlice [80], [161]
Network Virtualization Platform (NVP) [162]
OpenVirteX [77], [163], [164]
OF NV Cloud [165]
AutoVFlow [166]
Carrier-grade [167]
Datapath Centric [168]
DFVisor [169], [170]
OpenSlice [171]
Advanced Capabilities [172]
Hyperflex [173]

Fuente: Survey on Network Virtualization Hypervisors for software defined networking [13].pág. 672

El objetivo principal de Flowvisor es ejecutar redes de producción y experimentales en el mismo hardware de red físico. Por lo tanto, se centra en los mecanismos para aislar la red experimental del tráfico de la red de producción.
Flowvisor trabaja de manera transparente, sin afectar las redes virtuales alojadas, proporcionando definiciones extensibles, modulares y flexibles de porciones de red [13]. Además de lo anterior cuenta con características como:

- Licencia de uso libre OpenSource.
- Programabilidad.
- Flexibilidad.
- Compatibilidad con la plataforma Geni.

Aunque por otro lado dicho hipervisor actualmente no tiene soporte, ni actualizaciones o mejoras en su código fuente, lo cual representa una limitante en compatibilidad con los nuevos desarrollos en lo que a SDN se refiere.

- **Actividad 3. Instalación del hipervisor seleccionado en la plataforma Geni.**

  Para realizar la instalación del hipervisor Flowvisor en la plataforma Geni se siguen los siguientes pasos:

  a) **Solicitud de recurso en la plataforma Geni.**
  Se solicita el recurso 1 large XEN VM with public IP (IG) el cual está disponible en la lista de hardware preconfigurado de Geni. Este consta de una máquina virtual con un sistema operativo Linux Ubuntu 14.04 preinstalado y una ip pública para el ejercicio de la experimentación. Dicho recurso se etiqueta o nombra como Hipervisor para su fácil identificación entre los diversos recursos que se solicitarán para el ejercicio experimental del proyecto.

  b) **Consulta de documentación.**
  Posteriormente a la asignación del recurso por parte de los racks de Geni, se realiza la consulta de diversas fuentes web con el objeto de tener una recopilación de información, referente al cómo realizar la instalación de Flowvisor en el recurso ya obtenido en la plataforma Geni, ya que como se ha mencionado previamente no hay documentación completa o manuales paso a paso de la instalación, por parte de una fuente confiable.

  c) **Instalación del software Flowvisor.**
  Para efectuar la instalación del software Flowvisor en el recurso asignado y nombrado previamente como Hipervisor en la plataforma Geni. Se llevan a cabo varias pruebas haciendo uso de la información ya recolectada con el objeto de identificar y aprender la forma más eficiente de realizar la instalación del software Flowvisor. De esta forma se genera un documento completo sobre el proceso funcional de dicho procedimiento el cual es el Anexo B de este proyecto.

- **Actividad 4. Pruebas con el hipervisor más conveniente para la implementación.**

  Para realizar las pruebas con el hipervisor es necesario tener claro los elementos vinculados en el proceso a nivel de hardware y software, de igual manera los recursos requeridos en la plataforma Geni estos son:

  - Recurso switch físico o virtual de red compatible con el protocolo OpenFlow.
  - Máquina virtual con software Controlador de red que tenga asignada una ip pública.
  - Máquina virtual con software Flowvisor con ip pública asignada.

  La documentación encontrada sugería ejercicios o ejemplos de forma separada elemento por elemento. Es decir, ejemplos sobre la implementación de dichos componentes en forma aislada y no de una manera completa, o donde se explicará cómo se unían todos los elementos o se configuraban para tener una conectividad entre los mismos, para realizar la virtualizacion de la red.
  Para realizar estas pruebas se siguen los pasos:

  i. **Solicitud de recursos en la plataforma Geni.**
     Se realiza la solicitud de dos recursos diferentes en la plataforma Geni, estos se encuentran en la lista de configuraciones pre-existentes de la misma y son:
     - Openflow OVS all XEN el cual es un recurso compuesto por un switch virtual Openvswitch, compatible con el protocolo Openflow, y a su vez este cuenta con tres nodos, los cuales son máquinas virtuales con un sistema operativo linux 14.04 preinstalado.
     - XEN VM POX Ctrl este recurso es el controlador y está compuesto por una máquina virtual con un sistema operativo Linux 14.04 con ip publica, a su vez cuenta con la preinstalación del software Pox el cual al ejecutarse cumple la función de controlador.
     - 1 large XEN VM with public IP (IG) este consta de una máquina virtual con un sistema operativo Linux Ubuntu 14.04 preinstalado y una ip pública para el ejercicio de la experimentación, en este equipo se realizará la instalación del software Flowvisor, el cual cumplirá la función de hipervisor.

ii. Configuración del software en los recursos solicitados.

Se configuran de manera independiente cada uno de los elementos ya mencionados, con su respectivo software como preparación para el desarrollo de pruebas. Obteniendo como resultado:

- 1 Switch virtual OpenvSwitch con Openflow activo, debidamente configurado y 3 host virtuales conectados y funcionales para las pruebas de transmisión.
- 1 Máquina virtual con ip pública y software Pox correctamente configurado para realizar la función de controlador de red.
- 1 Máquina virtual con ip pública y software Flowvisor correctamente instalado para realizar la función de hipervisor de red.

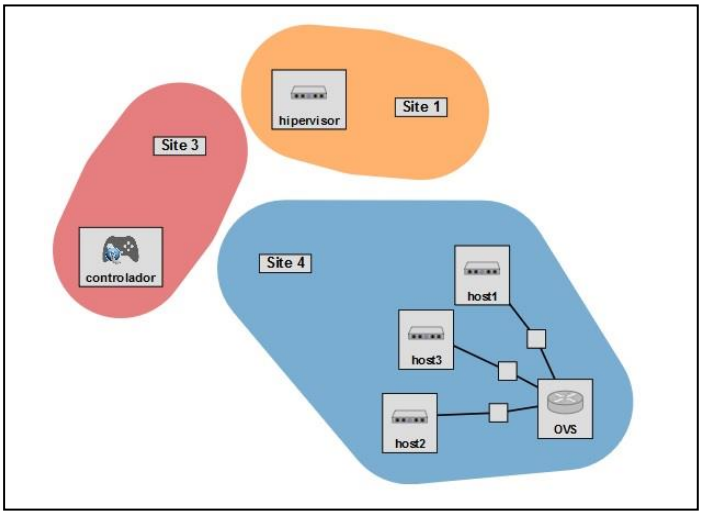
iii. Desarrollo de pruebas con el hipervisor.

El proceso de pruebas del hipervisor se divide en tres partes:

I. Planteamiento

Como prueba para verificar la funcionalidad del hipervisor Flowvisor se plantea el ejercicio de llevar a cabo la virtualizacion de una red, haciendo uso del hardware solicitado, asignado y configurado previamente en la plataforma Geni. Tal como se observa en la imagen 2.

Imagen 2. Recursos plataforma Geni pruebas hipervisor



Fuente:https://portal.geni.net/secure/slice-add-resources

.

II. Desarrollo

Para iniciar el desarrollo del ejercicio se realiza la configuración individual de los diversos recursos asignados por Geni, dichos elementos se configuran de acuerdo a la documentación hallada en la web y los repositorios de la plataforma.

Desafortunadamente tras la configuración no se logró obtener conectividad entre los recursos asignados y previamente configurados.

El hipervisor, el switch y el controlador no se lograban conectar entre sí, por lo tanto, no se podía realizar la verificación del funcionamiento del hipervisor, y mucho menos evaluar su funcionalidad para el objeto del proyecto.

Como se mencionó al inicio de esta actividad no se cuenta con un manual paso a paso, con la configuración para la integración de los recursos ya asignados en la plataforma, por dicha falta de documentación es necesario un periodo de experimentación en Geni, e investigación detallada de los manuales de funcionamiento de cada uno de los recursos ya asignados y configurados en la plataforma. Y de esta manera encontrar la solución al problema de conectividad.

III. Investigación

Como resultado de la investigación y experimentación en el desarrollo de las pruebas, el inconveniente de conectividad fue plenamente identificado.

El problema radicaba en las diversas versiones que posee el protocolo Openflow en los recursos previamente configurados y asignados por Geni [3], el protocolo Openflow es el encargado de la intercomunicación entre los dispositivos de la red virtual como el hipervisor, controlador y los switches. Actualmente el protocolo posee diferentes versiones que van desde la 1.0 a la 1.3 y cada dispositivo trae por defecto configurado una versión diferente del protocolo.

Dicha diferencia entre las versiones fue lo cual ocasiono la falla a la hora de realizar la comunicación entre los dispositivos. Para llevar a efecto la correcta integración de los elementos se debió definir y configurar una sola versión entre los dispositivos. Ya que como se mencionó anteriormente sobre el hipervisor Flowvisor este no posee actualizaciones o mejoras y funciona sobre la versión 1.0 del protocolo Openflow, esta falla en la conectividad entre el hipervisor, el controlador y los switches no se halló documentada ni mencionada en la información consultada, y encontrar dicha solución solo fue posible por medio de la

experimentación y evaluación exhaustiva de los manuales de configuración de los elementos en forma aislada.

IV. Ejecución y resultado
Al ejecutar la solución hallada se logró tener conectividad entre el hipervisor, el switch y el controlador. Se pudo terminar la fase de pruebas del hipervisor, y a su vez se determinó que Flowvisor es el hipervisor más adecuado y compatible con la plataforma Geni para el desarrollo de este proyecto. De igual manera se genera el documento anexo D en cual se registra la configuración adecuada para tener la conectividad entre el hipervisor, el switch y el controlador respectivamente.

5.1.2 Fase 2. Definir el controlador para la configuración del sistema.

- **Actividad 1. Búsqueda de bibliografía, web, repositorios universitarios, IEEE.**

Se realiza una revisión sobre la información pertinente al controlador de red en los repositorios de la IEEE y Science Direct encontrando alrededor de 12 artículos con los cuales se aclaran las funciones del mismo, su papel protagónico en la virtualizacion y programación de redes virtuales. Ya que en las redes definidas por software el controlador es un software encargado de administrar el grupo de swiches que manejan el tráfico de la red y a su vez permite tener un mayor control del mismo como se estudió en [16]. Este software ofrece a los proveedores el control de la red virtual de los switches. De dicho componente se encuentra documentación clara y general sobre sus configuraciones, sintaxis y compatibilidad, pero en la implementación con el hipervisor y los switch no se encuentra información clara como ya se mencionó.

- **Actividad 2. Preselección de los controladores encontrados.**

En los antecedentes bibliográficos hallados se encontraron diversos controladores clasificados por su tipo de licencia, lenguaje de programación, y versión que posee de Openflow como se observa en la imagen 3.

Para la selección del controlador más adecuado se tuvo en cuenta características tales como:

- Interfaz grafica
  El controlador debe tener un ambiente grafico para lograr una visualización en pantalla, y de esta forma realizar un seguimiento o

exposición de las topologías de red de que se están evaluando de una manera más intuitiva, y que a su vez sea más cómodo a la hora de mostrar los resultados de la investigación.

Imagen 3. Listado de controladores.

CONTROLLERS CLASSIFICATION

Name	Architecture	Northbound API	Consistency	Faults	License	Prog. language	Version
Beacon [186]	centralized multi-threaded	ad-hoc API	no	no	GPLv2	Java	v1.0
DISCO [185]	distributed	REST	—	yes	—	Java	v1.1
Fleet [199]	distributed	ad-hoc	no	no	—	—	v1.0
Floodlight [189]	centralized multi-threaded	RESTful API	no	no	Apache	Java	v1.1
HP VAN SDN [184]	distributed	RESTful API	weak	yes	—	Java	v1.0
HyperFlow [195]	distributed	—	weak	yes	—	C++	v1.0
Kandoo [228]	hierarchically distributed	—	no	no	—	C, C++, Python	v1.0
Onix [7]	distributed	NVP NBAPI	weak, strong	yes	commercial	Python, C	v1.0
Maestro [188]	centralized multi-threaded	ad-hoc API	no	no	LGPLv2.1	Java	v1.0
Meridian [192]	centralized multi-threaded	extensible API layer	no	no	—	Java	v1.0
MobileFlow [222]	—	SDMN API	—	—	—	—	v1.2
MuL [229]	centralized multi-threaded	multi-level interface	no	no	GPLv2	C	v1.0
NOX [26]	centralized	ad-hoc API	no	no	GPLv3	C++	v1.0
NOX-MT [187]	centralized multi-threaded	ad-hoc API	no	no	GPLv3	C++	v1.0
NVP Controller [112]	distributed	—	—	—	commercial	—	—
OpenContrail [183]	—	REST API	no	no	Apache 2.0	Python, C++, Java	v1.0
OpenDaylight [13]	distributed	REST, RESTCONF	weak	no	EPL v1.0	Java	v1.{0,3}
ONOS [117]	distributed	RESTful API	weak, strong	yes	—	Java	v1.0
PANE [197]	distributed	PANE API	yes	—	—	—	—
POX [230]	centralized	ad-hoc API	no	no	GPLv3	Python	v1.0
ProgrammableFlow [231]	centralized	—	—	—	—	C	v1.3
Rosemary [194]	centralized	ad-hoc	—	—	—	—	v1.0
Ryu NOS [191]	centralized multi-threaded	ad-hoc API	no	no	Apache 2.0	Python	v1.{0,2,3}
SMaRtLight [198]	distributed	RESTful API	no	no	Apache	Java	v1.0
SNAC [232]	centralized	ad-hoc API	no	no	GPL	C++	v1.0
Trema [190]	centralized multi-threaded	ad-hoc API	no	no	GPLv2	C, Ruby	v1.0
Unified Controller [171]	—	REST API	—	—	commercial	—	v1.0
yanc [196]	distributed	file system	—	—	—	—	—

Fuente: A Comprehensive Survey[17] pág. 19.

- Compatibilidad con el protocolo Openflow versión 1.0
  Es un requisito importante que el controlador posea compatibilidad con la versión 1.0 del protocolo Openflow, ya que el hipervisor seleccionado cuenta con el software Flowvisor el cual solo es compatible con la versión ya mencionada, y de lo contrario se tendrían problemas de conectividad entre los diversos dispositivos, como ya sucedió anteriormente en la sección 5.1.1.4.c donde se evidencio dicho problema por las diversas versiones del protocolo Openflow.

- Documentación

Es necesario que el controlador cuente con amplia documentación de su sintaxis, forma de instalación, ejemplos o ejercicio de su uso y soporte. Para de esta manera contrarrestar las dificultades que puedan surgir en el transcurso del desarrollo del proyecto.

- Licencia de uso libre
  El software controlador debe contar con esta característica ya que por costos del proyecto la adquisición de una licencia tipo comercial no fue contemplada, y se requiere poder tener el control de cambios al código fuente si así lo requiere el caso.

- Programabilidad
  El controlador de red es un elemento que debe permitir cualquier tipo de modificación o creación de scripts, ajustados a la necesidad puntual de la red, hablado en términos de flujos de información. Por esta razón se requiere que posea la característica de ser programable.

- Compatibilidad con la plataforma de pruebas
  La plataforma seleccionada para el desarrollo de este proyecto es Geni, y por lo tanto es necesario que el software controlador seleccionado sea compatible con dicha plataforma, o en otras palabras sea instalable, configurable y funcional en los recursos que proporciona Geni.

Teniendo en cuenta los criterios de selección mencionados y la documentación encontrada como [18], se preseleccionan 2 controladores para ser instalados y sometidos a pruebas de conectividad y uso con el hipervisor y el switch en la plataforma Geni. Los preseleccionados fueron:

- Opendayligth
  Software controlador programado en java, poseedor de una interfaz gráfica muy robusta, además de actualizaciones permanentes en su código fuente lo cual lo hace muy estable y documentado para su uso en pruebas. Además, cuenta con una licencia de código abierto.

- Floodlight
  Software controlador para redes definidas por software, programado en java, poseedor de interfaz gráfica, además soporte en su código fuente, amplia documentación en pruebas, estabilidad y licencia de uso libre.

Ambos controladores con características muy similares y como se destaca en [18] Floodlight es de interés ya que es uno de los controladores de código abierto, tiene una amplia documentación, y se desarrolla activamente . Además, Floodlight admite swiches virtuales, lo que facilita el desarrollo y la prueba de módulos tantas veces como sea necesario en un entorno virtual. Ya que está escrito en Java, es modular, soporta múltiples subprocesos, gestión de memoria, y se ejecuta en varias plataformas.
En el caso de Opendaylight dicho controlador cuenta con características similares a Floodlight pero carece de compatibilidad para trabajar con Openflow versión 1.0 generando conflicto ya que el hipervisor seleccionado Flowvisor solo se comunica por medio del versión de Openflow ya mencionada, esto se comprobó a través de la instalación del mismo y las pruebas realizadas entre el controlador y el hipervisor en la plataforma Geni y por dicha razón fue descartado como opción para este proyecto.

- **Actividad 3. Instalación del controlador seleccionado en la plataforma Geni.**

Para realizar la instalación del controlador en la plataforma Geni se completaron los siguientes pasos:

Para realizar la instalación del hipervisor Flowvisor en la plataforma Geni se debió seguir una serie de pasos, estos son:

a) **Solicitud de recurso en la plataforma Geni.**
Se solicita el recurso XEN OpenFlow Controller el cual está disponible en la lista de hardware pre configurado de Geni. Este consta de una máquina virtual con un sistema operativo Linux Ubuntu 14.04 preinstalado y una ip pública para el ejercicio de la experimentación.

b) Consulta de documentación.
Posteriormente a la asignación del recurso por parte de los racks de Geni, se realiza la consulta de diversas fuentes web con el objeto de tener una recopilación de información, referente al cómo realizar la instalación de Floodlight en el recurso ya obtenido en la plataforma Geni. Dicho software cuenta con diversas fuentes bibliográficas sobre el proceso de instalación y puesta en marcha [19], [20] .

c) Instalación del software controlador Floodlight.
Para efectuar la instalación del software Floodlight en el recurso asignado previamente en la plataforma Geni. Se llevan a cabo varias pruebas haciendo uso de la información ya recolectada con el objeto de identificar y aprender la forma más eficiente de realizar la instalación del software Floodlight. De esta forma se genera un documento completo

sobre el proceso funcional de dicho procedimiento el cual es el Anexo D de este proyecto.
Cabe aclarar que dicho controlador viene pre configurado para trabajar con el protocolo Openflow en la versión 1.3 por defecto, pero por medio de la modificación de su código fuente, dicho controlador puede ser modificado para operar en la versión 1.0 del protocolo y tener una completa conectividad con el hipervisor Flowvisor.

5.1.3 Fase 3. Implementar una red virtual en la plataforma Geni.

Para realizar la implementación de la red virtual son necesarios diversos recursos, para estos casos se solicita a la plataforma los siguientes elementos:

- Grupo de 3 Switches físicos o virtuales de red compatible con el protocolo OpenFlow interconectados entre sí.
- Máquina virtual con software Controlador de red que tenga asignada una ip pública.
- Máquina virtual con software Flowvisor con ip pública asignada.

Como se mencionó anteriormente la documentación encontrada sugería ejercicios o ejemplos de forma separada elemento por elemento, y no de una manera completa, o donde se explicará cómo se unían todos los elementos o se configuraban para tener una conectividad entre los mismos. Para realizar esta implementación se siguen los siguientes pasos:

i. Solicitud de recursos en la plataforma Geni.
Se realiza la solicitud de los recursos a la plataforma Geni como se ha expuesto previamente en el anexo A de este documento. Algunos de los elementos requeridos se encuentran en la lista de configuraciones pre-existentes de la misma y son:

- Openflow OVS el cual es un recurso compuesto por un switch virtual Openvswitch compatible con el protocolo Openflow. De este elemento se requiere solicitar 3 equipos e interconectarlos por medio de enlaces. La interconexión y selección de los elementos se deben realizar de manera manual, ya que esta topología de red no se encuentra entre las configuraciones preseleccionadas de Geni.
- XEN VM POX Ctrl este recurso es el controlador y está compuesto por una máquina virtual con un sistema operativo Linux 14.04 con ip pública, en este recurso se debe realizar la instalación del software controlador Floodlight, como se expuso en el anexo C de este documento, y dicho equipo cumplirá la función de controlador red.

- 1 large XEN VM with public IP (IG) este consta de una máquina virtual con un sistema operativo Linux Ubuntu 14.04 preinstalado y una ip pública para el ejercicio de la experimentación en este equipo se realizará la instalación del software Flowvisor como se explicó en el anexo B, y este cumplirá la función de hipervisor.

ii. Configuración del software en los recursos solicitados.

Se configura de manera independiente cada uno de los elementos ya mencionados con su respectivo software. Obteniendo como resultado:

- 3 Switch virtual OpenvSwitch con Openflow activo, debidamente configurados e interconectados.
- 1 máquina virtual con ip pública y software Floodlight correctamente configurado para realizar la función de controlador de red.
- 1 máquina virtual con ip pública y software Flowvisor correctamente instalado para realizar la función de hipervisor de red.

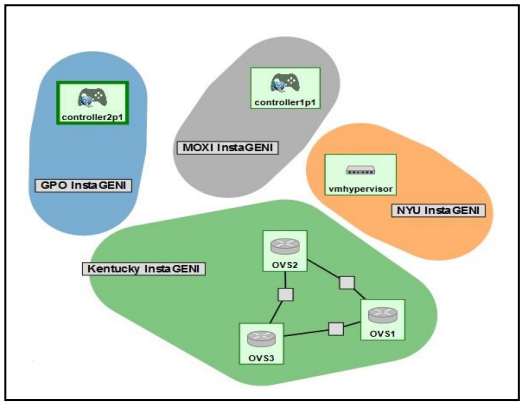
iii. Desarrollo de la implementación.

El proceso de implementación se divide en cuatro partes:

a. Planteamiento

Se plante una topología que consta de 3 switches, 1 controlador y 1 hipervisor. Los 3 switches funcionaran como la red sustrato sobre la cual se realizará la incrustación de la red virtual, el controlador de red que se encargara de controlar los flujos de información de la red virtual y el hipervisor cumplirá con la función de incrustación sobre la red sustrato. Tal como se observa en la imagen 4.

Imagen 4. Topología de red para incrustación de red virtual.



Fuente: https://portal.geni.net/secure/slice-add-resources

b. Desarrollo

Para iniciar el desarrollo de la implementación se realiza la configuración individual de los diversos recursos asignados por Geni, dichos elementos se configuran como se realizó y registro previamente en los anexos B, C, D.

Después de la configuración se verifica la conectividad entre los recursos asignados y ya configurados. Posteriormente se procede a realizar las respectivas configuraciones en el hipervisor para la correcta implementación de la incrustación de la red virtual en la red sustrato.

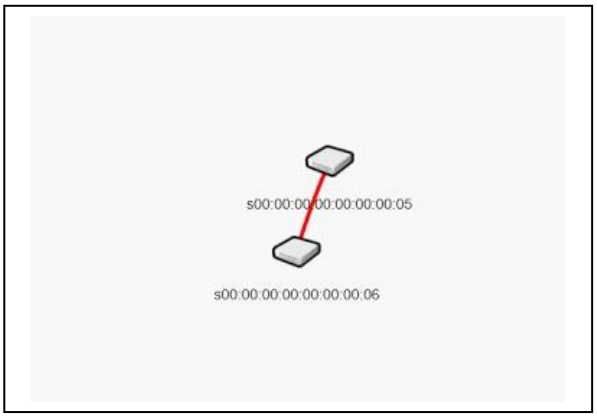
c. Implementación

Para realizar la implementación se deben crear los respectivos slices y flowspaces por parte del hipervisor. Para hacer esto es necesario que todos los elementos vinculados funcionen y se comuniquen de forma correcta. Ya que sin esta convergencia no sería posible llevar acabo la incrustación de la red virtual. Y como se mencionó anteriormente, la dificultad de comunicación encontrada en el desarrollo de este proyecto entre los equipos de la red, dependía de las diversas versiones del protocolo Openflow utilizado por los swiches, el controlador y el hipervisor. Teniendo plenamente identificado y corregido el problema de comunicación, la implementación continúa con la ejecución de los comandos del hipervisor Flowvisor necesarios para realizar la incrustación de la red virtual.

d. Ejecución y resultado

Como resultado de la adecuada ejecución por parte del hipervisor, se obtiene una incrustación exitosa de una red virtual, en la red sustrato previamente configurada. Esto se comprueba por medio de la interfaz gráfica proporcionada por el controlador Floodlight, ya que se puede observar la topología de red incrustada como se aprecia en la imagen 5.

Imagen 5. Red virtual incrustada.



Fuente: https://128.104.159.128:8080

La ejecución de esta incrustación fue realizada de forma manual y la misma se encuentra debidamente documentada paso a paso, como el anexo E de este proyecto.

5.1.4 Fase 4. Programar el hipervisor con el algoritmo de asignación dinámica de recursos.

El proceso de desarrollo de los scripts de automatización en el hipervisor Flowvisor consto de varias etapas:

a. Planteamiento.
Se establecieron como funciones principales y alcances del script los siguientes:

- Lectura de archivo con mapeo de nodos y enlaces.
- Creación de slices de red.
- Creación de flowspaces.
- Borrado de slices
- Borrado de flowspaces.
- Medición de tiempos en ejecución de la creación de los flowspaces.
- Medición de tiempos en ejecución de la creación de los y slices.

b. Desarrollo
Se realiza la evaluación de diversos lenguajes de programación como Python, java, php y bash. Se decide realizar el desarrollo de los scripts

en lenguaje bash por su alta compatibilidad con Linux, bajo consumo de recursos y la razón más importante lo compatible que este es con el software Flowvisor. Ya que para este desarrollo de programación es importante tener en cuenta que el script se ejecutara desde el mismo equipo que cumple la función de hipervisor.

c. Investigación

Se realiza una revisión bibliográfica de la sintaxis, ejemplos y manejo del lenguaje bash, ya que este posee una sintaxis propia, y compleja. Adicionalmente este no posee un ide de programación que permita identificar errores a nivel sintáctico. Dicha limitante hizo más complejo el desarrollo de los scripts por las constantes revisiones y pruebas para verificar la correcta codificación y ejecución del programa. De igual manera se realizó una revisión más profunda de la documentación de los comandos principales del software Flowvisor para la creación de slices y flowspaces. Y de esta forma tener una mayor claridad de como incluirlos y usarlos adecuadamente en el desarrollo de esta implementación automatizada.

d. Implementación

Se realiza la programación de los diversos scripts en lenguaje bash, que consta de diversos sub-scripts con funciones como creación, borrado, lectura de archivos entre otras. Estos se desarrollan usando como base principal los resultados entregados por el algoritmo BNP-VNE (Bactrack New Path Algebra-virtual network embedding). Dichos resultados proporciona la ruta de mapeo de los nodos y enlaces de las redes virtuales que se deben incrustar en la red sustrato, de acuerdo a diversas métricas propias del algoritmo. Los scripts desarrollados poseen una documentación clara y concisa. Toda esta información se encuentra como el anexo F de este proyecto.

5.1.5 Fase 5. Realizar pruebas de la automatización del proceso de virtualización en la plataforma Geni.

Para la realización de pruebas con el script de automatización es necesario:

1) **Solicitud de recursos en la plataforma Geni.**

Se realiza la solicitud de los recursos a la plataforma Geni como se ha expuesto previamente en el anexo A de este documento. Algunos de los elementos requeridos se encuentran en la lista de configuraciones pre-existentes:

- Openflow OVS el cual es un recurso compuesto por un switch virtual Openvswitch compatible con el protocolo Openflow. De este elemento se requiere solicitar 3 equipos e interconectarlos

por medio de enlaces. La interconexión y selección de los elementos se deben realizar de manera manual, ya que esta topología de red no se encuentra entre las configuraciones preseleccionadas de Geni.

- XEN VM POX Ctrl este recurso es el controlador y está compuesto por una máquina virtual con un sistema operativo Linux 14.04 con ip pública, en este recurso se debe realizar la instalación del software controlador Floodlight, como se expuso en el anexo C de este documento, y dicho equipo cumplirá la función de controlador red. De este elemento es necesario solicitar 2 equipos, ya que se realizaran la incrustación de dos redes virtuales, y cabe aclarar que cada red virtual estrictamente requiere de un controlador independiente y propio.

- 1 large XEN VM with public IP (IG) este consta de una máquina virtual con un sistema operativo Linux Ubuntu 14.04 preinstalado y una ip pública en este equipo se realizará la instalación del software Flowvisor como se explicó en el anexo B, este cumplirá la función de hipervisor y tendrá los scripts encargados de realizar la incrustación de las dos redes virtuales.

2) Configuración del software en los recursos solicitados.

Se configura de manera independiente cada uno de los elementos ya mencionados con su respectivo software. Obteniendo como resultado:

- 3 Switch virtual OpenvSwitch con Openflow activo, debidamente configurados e interconectados.
- 3 máquina virtual con ip pública y software Floodlight correctamente configurado para realizar la función de controlador de red.
- 1 máquina virtual con ip pública y software Flowvisor correctamente instalado para realizar la función de hipervisor de red.

3) Desarrollo de la implementación.

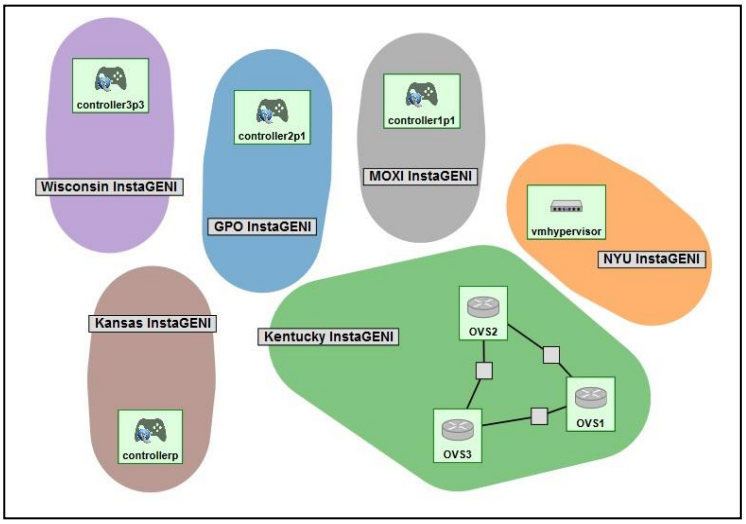
El proceso de implementación se divide en cuatro partes:

I. Planteamiento

Se plante una topología que consta de 3 switches, 3 controladores y 1 hipervisor. Los 3 switches funcionaran como

la red sustrato sobre la cual se realizará la incrustación de las redes virtual, los controladores de red que se encargara de controlar los flujos de información de la red virtual y el hipervisor cumplirá con la función de incrustación sobre la red sustrato por medio del uso del script de automatización de incrustación de redes virtuales. Tal como se observa en la imagen 6.

Imagen 6. Topología de red para prueba del script de automatización de incrustación de redes virtuales.



Fuente: https://portal.geni.net/secure/slice

II. Desarrollo

Para iniciar el desarrollo de la implementación se realiza la configuración individual de los diversos recursos asignados por Geni, dichos elementos se configuran como se realizó y registro previamente en los anexos B, C, D.

Después de la configuración se verifica la conectividad entre los recursos asignados y ya configurados. Posteriormente se procede a realizar las respectivas configuraciones en el hipervisor para la correcta implementación de la incrustación de la red virtual en la red sustrato.

III. Implementación

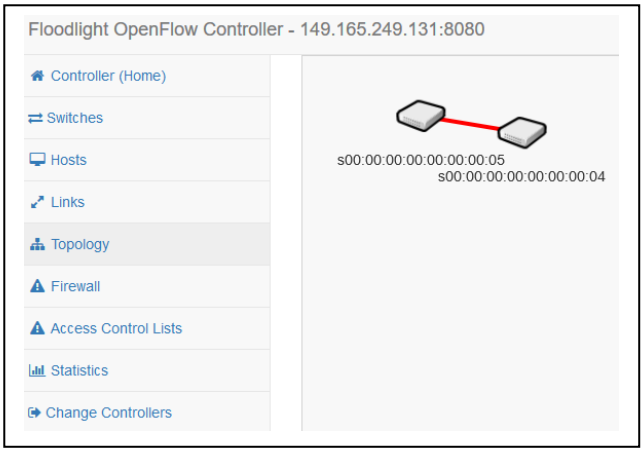
Para realizar la implementación se deben ejecutar el script programado para crear los respectivos slices y flowspaces por parte del hipervisor. Dicho script se encuentra debidamente documentado como el anexo F de este proyecto. Dicho script ejecuta los comandos del hipervisor Flowvisor necesarios para

realizar la incrustación de las redes virtuales de acuerdo a los resultados proporcionados por archivo el mapeo de nodos y enlaces.

IV. Ejecución y resultados

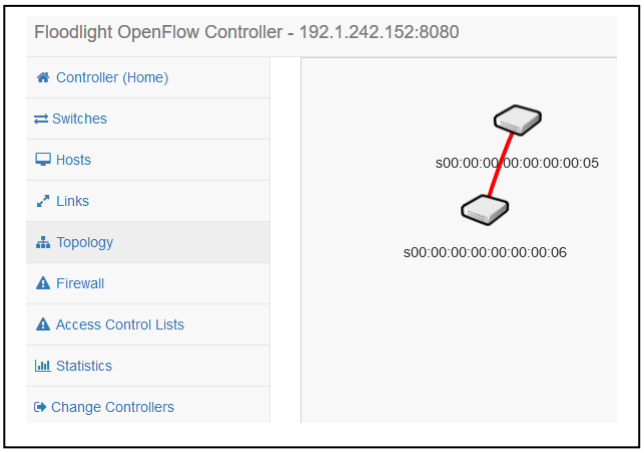
Como resultado de la adecuada ejecución por parte script de automatización de incrustación de redes virtuales del hipervisor Flowvisor, se obtiene el mapeo exitoso de dos redes virtuales, sobre la red sustrato solicitada a Geni previamente. Esto se comprueba por medio de la interfaz gráfica proporcionada por el controlador Floodlight, ya que se pueden observar las topologías de red pertenecientes a las dos redes incrustadas como se aprecia en la imagen 7 y 8.

Imagen 7. Red virtual 1 incrustada.



Fuente: https://149.165.249.131:8080

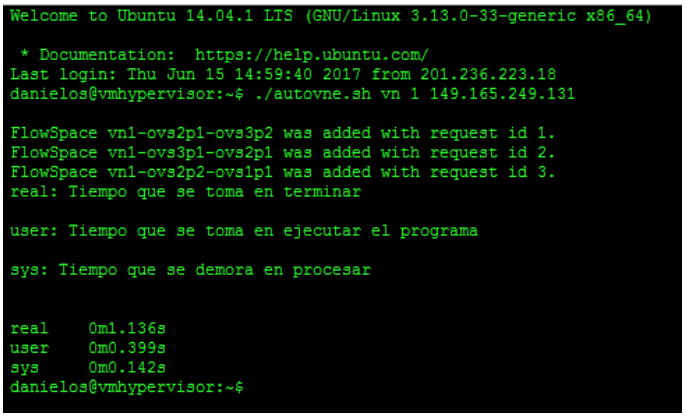
Imagen 8. Red virtual 2 incrustada.



Fuente: https://192.1.242.152:8080

En los scripts desarrollados se implementó una función para poder obtener los tiempos de incrustación de cada red virtual y de igual manera los tiempos de borrado de las mismas.
Los tiempos obtenidos en la prueba de incrustación de dos redes virtuales se encuentran a continuación:

Imagen 9. Tiempos de incrustación



Fuente: Fuente propia captura de pantalla plataforma Geni.

Como se observa el tiempo de creación de la nueva red virtual es de 1.1 segundos.

Imagen 10. Tiempos de eliminación

Fuente: Fuente propia captura de pantalla plataforma Geni.

En esta imagen observamos que el tiempo tomado por el script para eliminar las redes virtuales creadas es de 1.9 segundos.

6. RESULTADOS

6.1 DESCRIPCIÓN DE RESULTADOS

•En las primeras fases de investigación y ejecución, se realiza la incrustación de una red virtual de forma manual.
•Se documentan los manuales de instalación de los elementos preseleccionados para el proyecto como son el controlador, hipervisor y switch donde se explica de manera detallada las distintas configuraciones que permiten la comunicación entre estos, y así generar un escenario de pruebas.
•Se desarrolla un script en Bash, lenguaje Shell de Unix, que permite la programación del hipervisor Flowvisor, para automatizar el proceso de creación de redes virtuales, con los resultados entregados por el algoritmo de asignación dinámica de recursos.
El script de creación de redes virtuales posee las siguientes limitaciones:

- Incrustar redes virtuales con máximo dos nodos virtuales.
- Máximo dos saltos en la ruta mapeo de nodos.
- La ruta de mapeo de los enlaces, tanto de ida como para el regreso deben ser iguales.

•Se desarrolla un script el cual permite realizar el borrado completo de las redes virtuales creadas.
•Se agrega la funcionalidad para medir los tiempos de la incrustación y borrado de las redes virtuales por parte del hipervisor Flowvisor.

7. CONCLUSIONES

- La automatización completa de todo el procedimiento necesario para la incrustación de redes, puede ser planteado como un buen desafío para trabajos futuros, ya que elementos como el switch, el controlador y el hipervisor deben ser programados y asignados de forma independiente. El proceso se encuentra segmentado y en este trabajo solo se automatizo una parte del proceso (matlab, hipervisor), aun falta dar una solución completa y total del proceso ya mencionado.

- De acuerdo con los tiempos obtenidos en las diferentes incrustaciones de redes virtuales realizadas, se concluye que son óptimos y que proporcionan las condiciones favorables para su implementación a mayor escala.

- Debido a la naturaleza virtual del ambiente en el cual se realizan las diferentes pruebas de este proyecto, no proporcionan una confiabilidad total en los resultados, ya que en la actualidad esta plataforma se encuentra realizando una migración, lo que genera inestabilidad en los recursos sobre los que se desarrolla el trabajo.

- De acuerdo con la investigación realizada y las diferentes búsquedas bibliográficas, no se encuentran registros documentados acerca de la implementación e integración de los tres componentes necesarios para la ejecución de este proyecto
  .
- Actualmente podemos encontrar alternativas robustas en el tema de virtualizacion de redes, tales como VMware NSX, HP VAN, IBM SDN VE entre otros, los cuales proveen soluciones a nivel industrial.

- A pesar de que los componentes utilizados para este proyecto (Controlador, Hipervisor, Switches) se encuentran diseñados en diferentes lenguajes de programación, el protocolo Openflow se encarga de unificar y permitir la comunicación entre estos, siendo este protocolo la pieza fundamental para la ejecución y convergencia de dichos componentes.

- La plataforma Geni cuenta con una infraestructura distribuida en los Estados Unidos que proporciona las herramientas necesarias y los recursos adecuados para realización de diversas investigaciones.

8. RECOMENDACIONES

- Realizar una integración del algoritmo bnpa-vne con el hipervisor Flowvisor.
- Realizar un hibrido entre plataforma de pruebas Geni y elementos físicos de red, que permitan la obtención de resultados más confiables y acertados.
- Se recomienda la elaboración de un script de creación de redes virtuales mejorado que permita la programación del hipervisor Flowvisor, esto con el fin de poder:
    - Virtualizar redes con n nodos y n saltos
    - Que implemente la solución entregada por el algoritmo de incrustación de forma directa.

9. BIBLIOGRAFÍA

[1] D. R. L. García, *INTERNET, LA RED CON MAYUSCULAS* 1997.

[2] N. Feamster, J. Rexford, and E. Zegura, "An intellectual history of programmable networks," *ACM Queue,* vol. 11, pp. 1-21, 2013.

[3] T. A. Nick McKeown, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, Jonathan Turner. (2008, OpenFlow: Enabling Innovation in Campus Networks. [white paper]. 6.

[4] M. Chowdhury, M. R. Rahman, and R. Boutaba, "ViNEYard: Virtual Network Embedding Algorithms With Coordinated Node and Link Mapping," *Networking, IEEE/ACM Transactions on,* vol. 20, pp. 206-219, 2012.

[5] O. N. Foundation. (2014, OpenFlow Switch Specification. 277. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.0.noipr.pdf

[6] C. Huang, J. Zhu, M. Luo, and W. Chou, "A new mechanism for SDN network virtualization service," in *Smart Communications in Network Technologies (SaCoNeT), 2014 International Conference on*, 2014, pp. 1-6.

[7] A. Blenk, A. Basta, J. Zerwas, and W. Kellerer, "Pairing SDN with network virtualization: The network hypervisor placement problem," in *Network Function Virtualization and Software Defined Network (NFV-SDN), 2015 IEEE Conference on*, 2015, pp. 198-204.

[8] X. H.-S. Santi Villanueva, José Roberto Amazonas, "Contribution to the Virtual Network Embedding problem using a new Paths Algebra strategy," Universidad Politécnica de Catalunya, Catalunya, 2014.

[9] G. O. Arias, "Como escribir la investigación académica," in *Como escribir la investigación académica*, E. d. l. U, Ed., ed Bogotá, 2014, pp. 51-52.

[10] D. A. Drutskoy, "Software-Defined Network Virtualization with FlowN," In Partial Fullfillment of the Requirements for the Master of Science in Engineering Master of Science in Engineering
Department of Computer Science, Princeton University, 2012.

[11] P. D. a. V. R. D. Vinod K. Mishra, Ph.D., "GENI Deployment and Research at US Army Research Laboratory," presented at the IEEE Military Communications Conference, 2014.

[12] N. S. Foundation. (2016). *Wiki Geni Experimenter*. Available: http://groups.geni.net/geni/wiki/GENIExperimenter

[13] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, "Survey on Network Virtualization Hypervisors for Software Defined Networking," *Communications Surveys & Tutorials, IEEE,* vol. 18, pp. 655-685, 2016.

[14] A. Musa, S. Hwang, F. Muthohar, G. Bang, D. Choi, K. Kim*, et al.*, "Aggregation management design for user-defined network

infrastructure," in *The 16th Asia-Pacific Network Operations and Management Symposium*, 2014, pp. 1-4.

[15] M. Demirci and M. Ammar, "Design and analysis of techniques for mapping virtual networks to software-defined network substrates," *Computer Communications,* vol. 45, pp. 1-10, 6/1/ 2014.

[16] P. Heleno Isolani, J. Araujo Wickboldt, C. B. Both, J. Rochol, and L. Zambenedetti Granville, "Interactive monitoring, visualization, and configuration of OpenFlow-based SDN," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, 2015, pp. 207-215.

[17] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE,* vol. 103, pp. 14-76, 2015.

[18] L. V. Morales, A. F. Murillo, and S. J. Rueda, "Extending the Floodlight Controller," in *Network Computing and Applications (NCA), 2015 IEEE 14th International Symposium on*, 2015, pp. 126-133.

[19] Geni. *Intro to OpenFlow Tutorial with FloodLight Controller*. Available: http://groups.geni.net/geni/wiki/GENIExperimenter/Tutorials/OpenFlowOVS-Floodlight

[20] R. I. kc.wang@bigswitch.com. (2016). *Installation Guide*. Available: https://floodlight.atlassian.net/wiki/display/floodlightcontroller/Installation+Guide

ANEXOS

ANEXO A. SOLICITUD Y ASIGNACION DE RECURSOS PLATAFORMA GENI

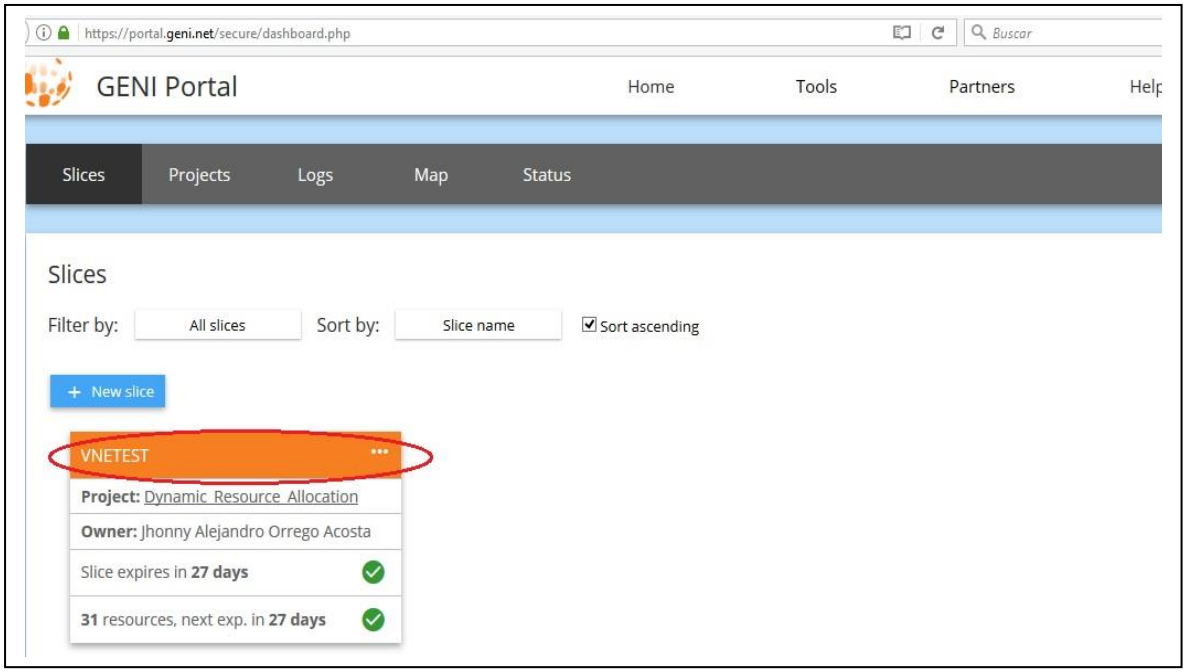
1. Requisitos para la solicitud y asignación de recursos en la plataforma Geni

Para realizar la solicitud de recursos en la plataforma se debe poseer credenciales de acceso. Dichas claves se deben hacer por medio de un tutor o investigador ya vinculado a la plataforma. Posterior al trámite de asignación de usuario y contraseña se debe pertenecer a un proyecto de investigación creado por el tutor o investigador. Y ya siendo parte de un proyecto de investigación se pueden solicitar los recursos que se necesiten.

2. Solicitud de recursos

Para realizar la solicitud de recursos en la plataforma se debe ingresar al proyecto creado con anticipación para este ejemplo VNETEST como se ve en la imagen 1.

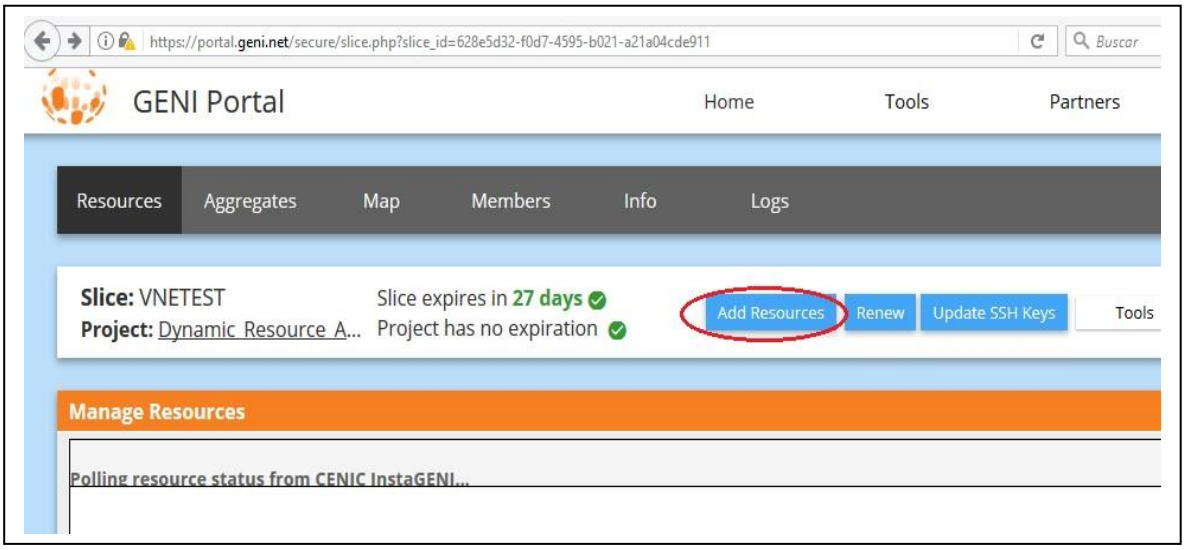
Imagen 1. Proyecto de investigación.



Fuente: Fuente propia captura de pantalla plataforma Geni.

Posteriormente se ingresa a la sección Add Resources como se ilustra en la imagen 2. Para acceder a la lista de recursos pre-configurados por Geni.

Imagen 2. Add Resources.

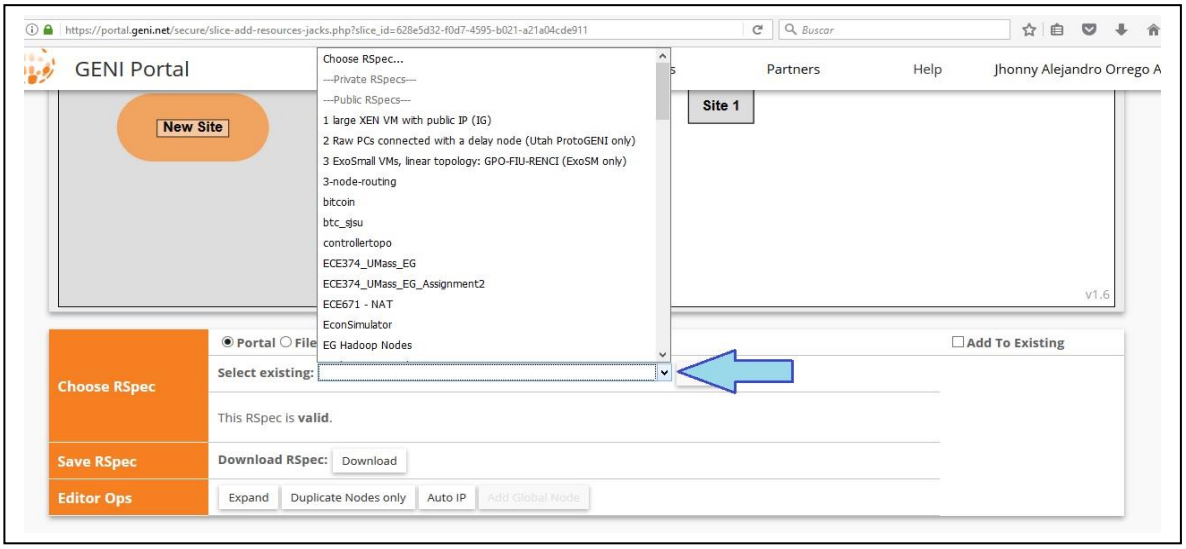


Fuente: Fuente propia captura de pantalla plataforma Geni.

3. Lista de recursos preconfigurados.

Para seleccionar los recursos preconfigurados se debe desplegar la lista de recursos existentes como se muestra en la imagen 3. En dicha lista encontramos recursos usados de manera continua o recurrente en diversos proyectos de investigación, los cuales fueron propuestos como recursos de alto uso y aplicación variada para los investigadores o proyectos.

Imagen 3. Lista de recursos

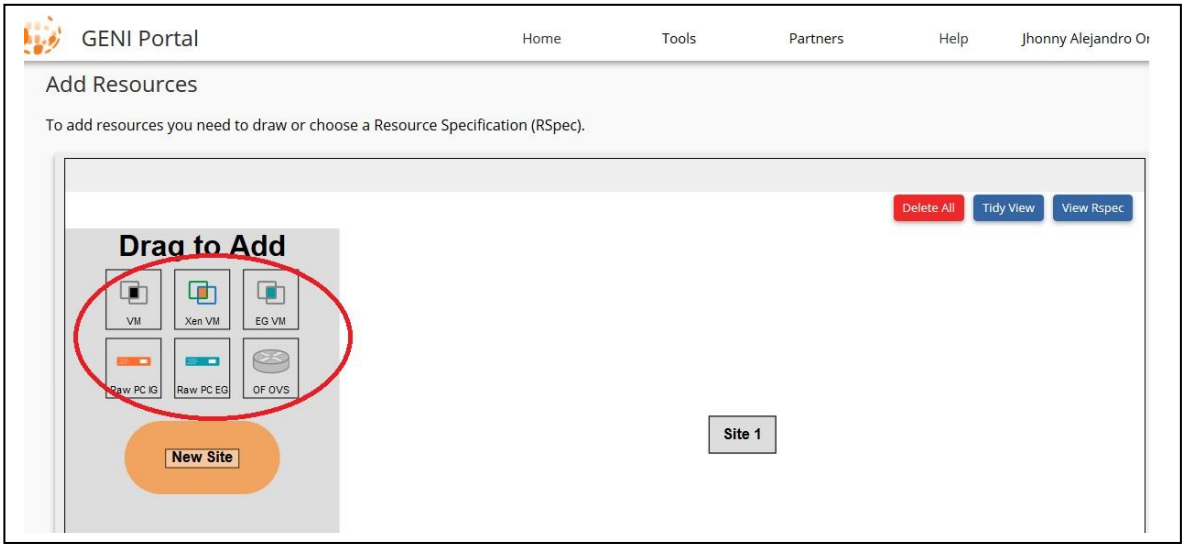


Fuente: Fuente propia captura de pantalla plataforma Geni.

4. Selección de recursos.

Para realizar la selección de recursos se debe seleccionar de la lista de elementos preconfigurados como se muestra en la imagen 3, o se puede realizar de manera manual seleccionando los elementos que se desean del panel de equipos disponibles como se muestra en la imagen 4.

Imagen 4. Selección de recursos de forma manual.



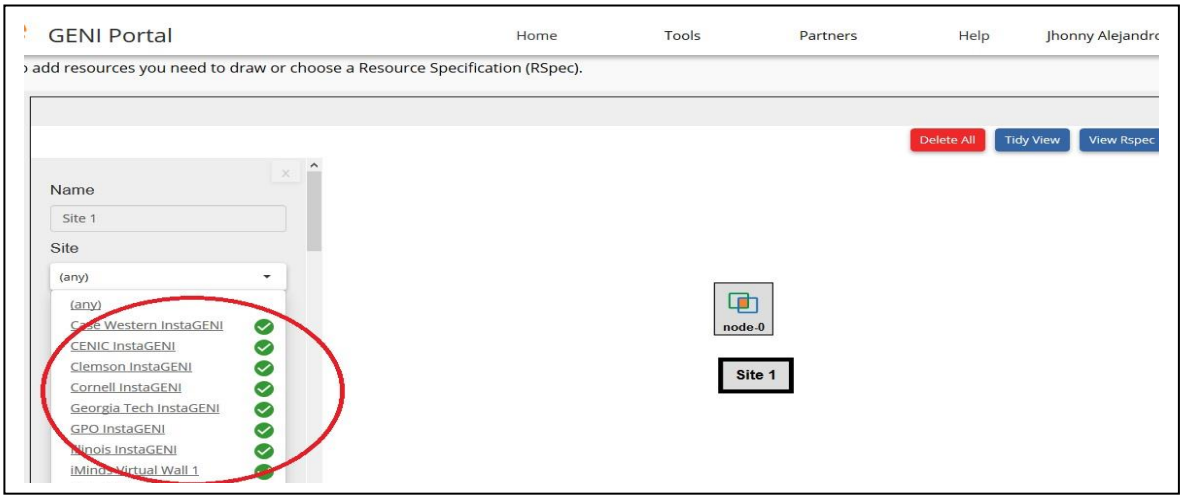
Fuente: Fuente propia captura de pantalla plataforma Geni.

Dicho panel permite arrastrar con click sostenido a la zona de selección de equipos, el tipo de recurso que se desea configurar de forma manual.

5. Selección de sitio del recurso.

Para realizar la selección del sitio al cual se va a solicitar los recursos se debe buscar en la lista de sitios disponibles como se observa en la imagen 5.

Imagen 5. Selección de sitio

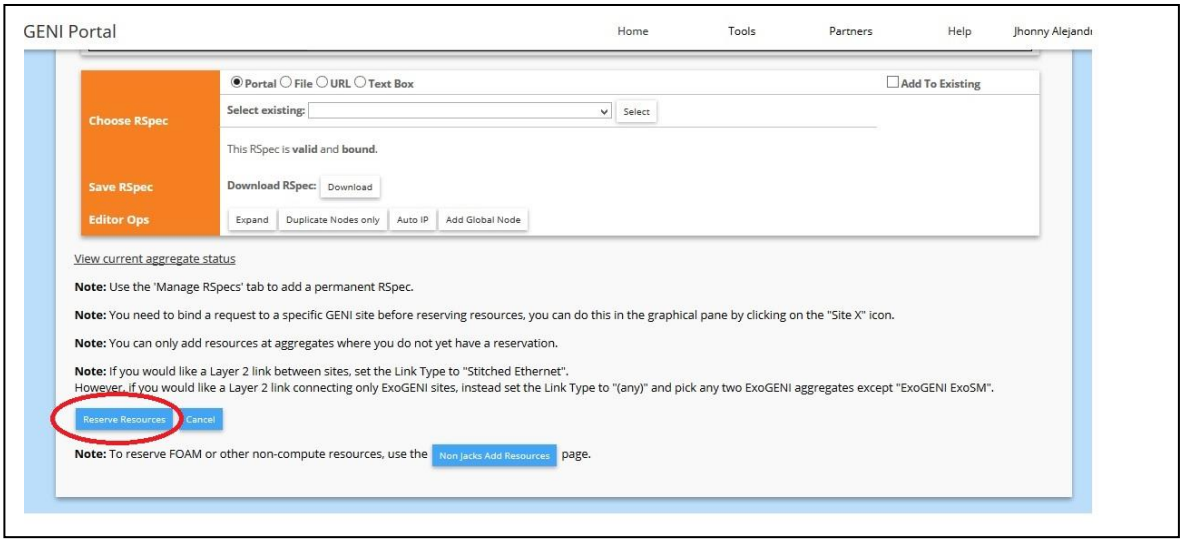


Fuente: Fuente propia captura de pantalla plataforma Geni.

6. Solicitud asignación del recurso.

Para obtener la asignación del recurso y del sitio al cual se solicitará el equipo o equipos, se debe hacer la solicitud seleccionando el botón Reserve Resources como se observa en la imagen 6.

Imagen 6. Selección de sitio.

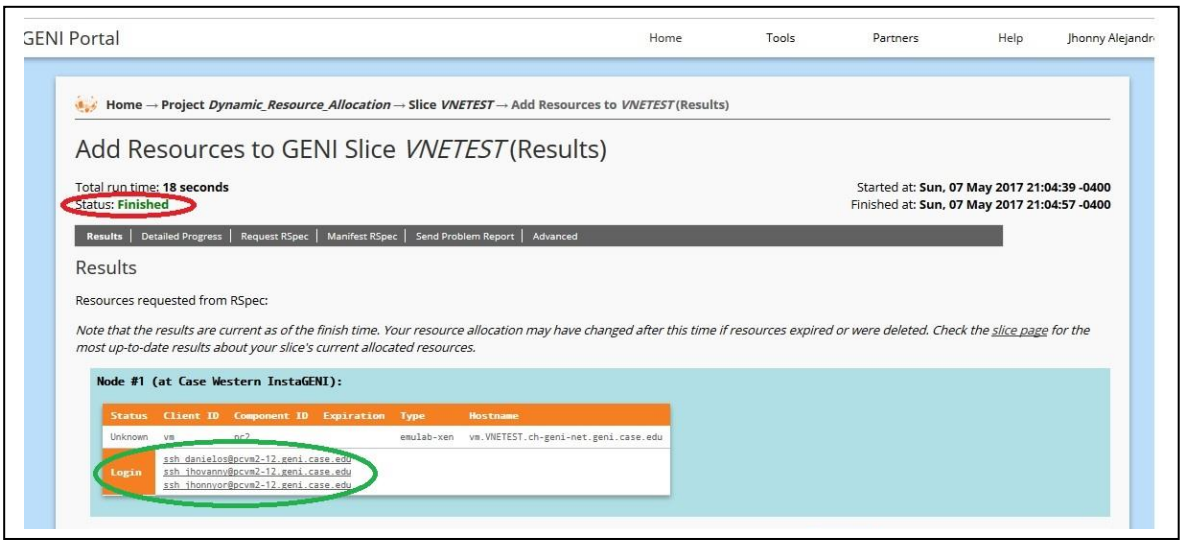


Fuente: Fuente propia captura de pantalla plataforma Geni.

7. Asignación del recurso.

Para finalizar si el recurso fue asignado correctamente se entregan los accesos y se obtiene un mensaje de confirmación de la disponibilidad del recurso como se observa en la imagen 7.

Imagen 7. Asignación del recurso.



Fuente: Fuente propia captura de pantalla plataforma Geni.

ANEXO B. INSTALACIÓN HIPERVISOR FLOWVISOR EN GENI

REQUISITOS PARA LA INSTALACION DEL HIPERVISOR FLOWVISOR FLOWVISOR

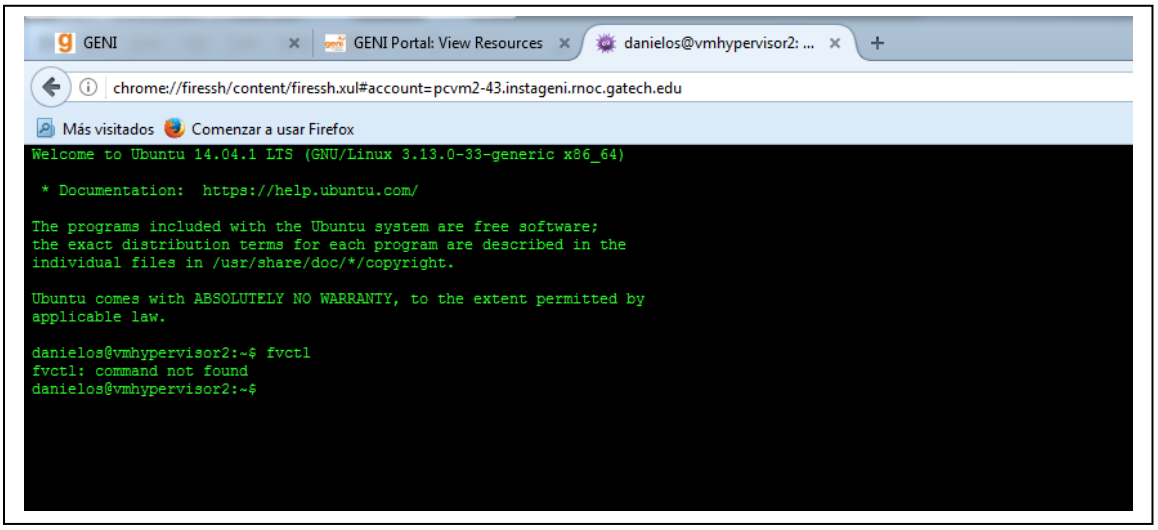
Esta instalación se realiza sobre un recurso virtual en la plataforma Geni con las siguientes características:

- Ubuntu 14.04.01 LTS

- GNU/Linux 3.13.0-33-generic x86_64

Antes de proceder con la instalación verificamos que Flowvisor no se encuentre instalado con el comando **$ fvctl** debe de retornar “comando not found” como se observa en la imagen 1.

Figura1. Captura de pantalla comando $ fvctl

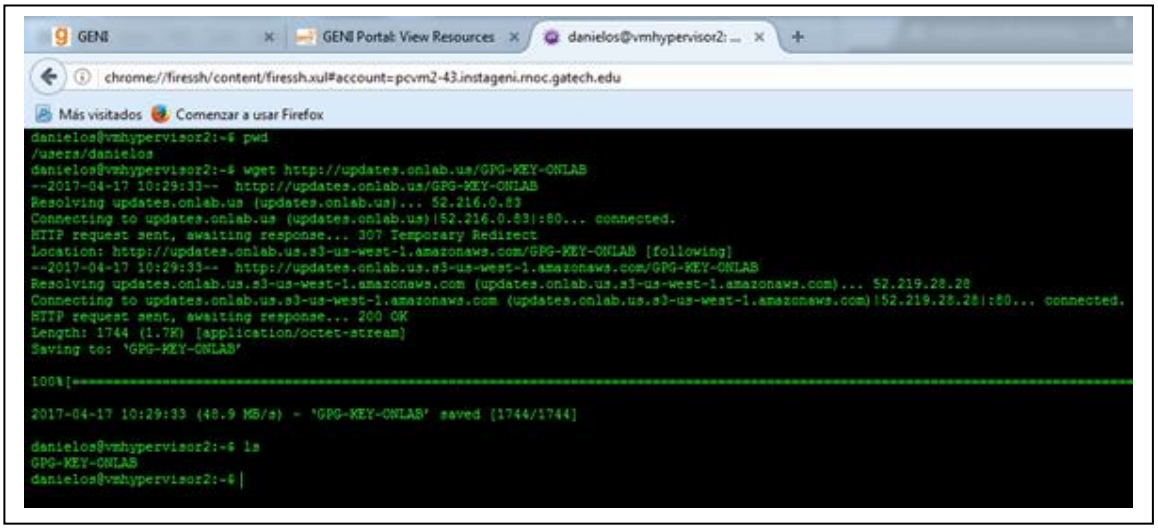


Fuente: Fuente propia captura de pantalla plataforma Geni.

1.1 Paso 1. Descarga de la llave del repositorio.
Se debe realiza la descargar de la llave del repositorio, para esto se verifica en que directorio se está actualmente con el comando **$ pwd** antes de realizar la descarga y poder acceder a ella posteriormente.

Se utiliza el comando **$ wget http://updates.onlab.us/GPG-KEY-ONLAB** para descargar la llave y se verifica con el comando $ ls que si haya sido correctamente descargada como se muestra en la imagen 2.

Imagen 2. Captura de pantalla ejecución de comando $ wget http://updates.onlab.us/GPG-KEY-ONLAB

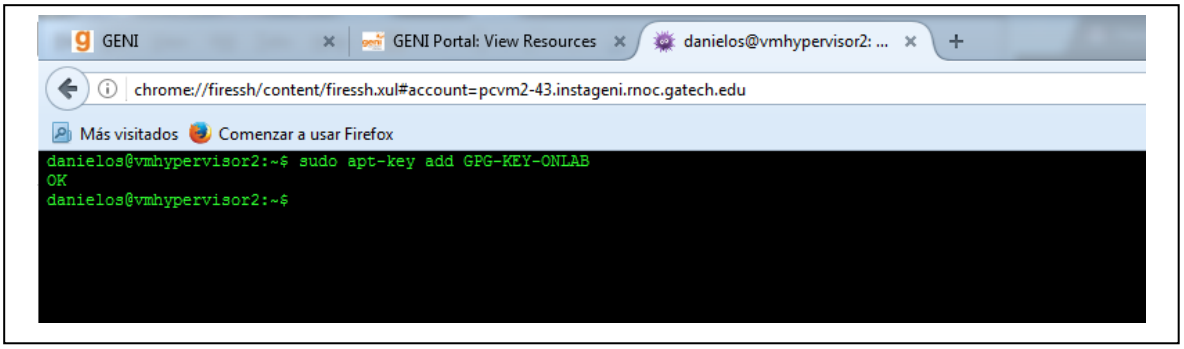


Fuente: Fuente propia captura de pantalla plataforma Geni.

1.2 Paso 2. Instalar la llave del repositorio.

Se procede a instalar la llave del repositorio con el comando **$ sudo apt-key add GPG-KEY-ONLAB** y como resultado se obtiene un mensaje de “OK” en la consola después de ejecutar este comando como se muestra en la imagen 3.

Imagen 3. Ejecución de comando $ sudo apt-key add GPG-KEY-ONLAB



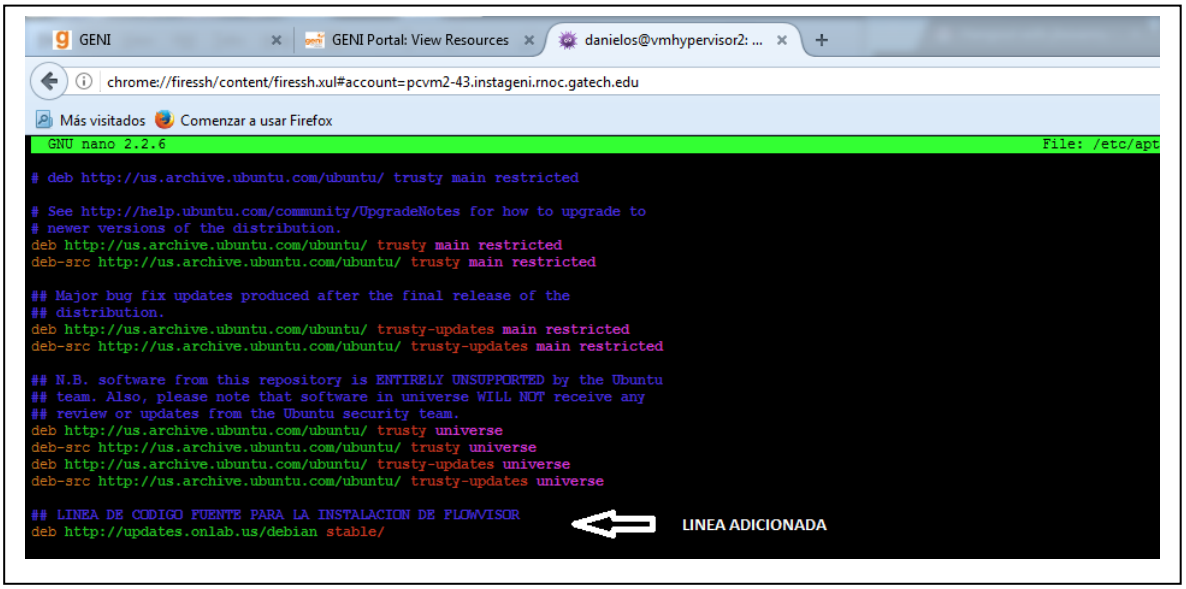
Fuente: Fuente propia captura de pantalla plataforma Geni.

1.3 Paso 3. Modificación de archivo source.list.

Para obtener lo repositorios más actualizados de descarga se debe añadir la siguiente línea **deb http://updates.onlab.us/debian stable/** en **/etc/apt/sources.list** para esto se utiliza el editor de texto nano, el comando completo a ejecutar seria:

$ sudo nano /etc/apt/sources.list se adiciona la línea como se muestra en la imagen 4 y se guardan los cambios con la combinación de teclas con Ctrl X---> Y---> Enter.

Imagen 4. Modificación de archivo source.list. Fuente: Fuente propia captura de pantalla



plataforma Geni.

1.4 Paso 4. Actualización de la base de datos de apt.

Para poder actualizar la base de datos de apt se debe ejecutar el comando **$ sudo apt-get update** y como se puede observar en la imagen 5 la forma en que se toma la actualización.

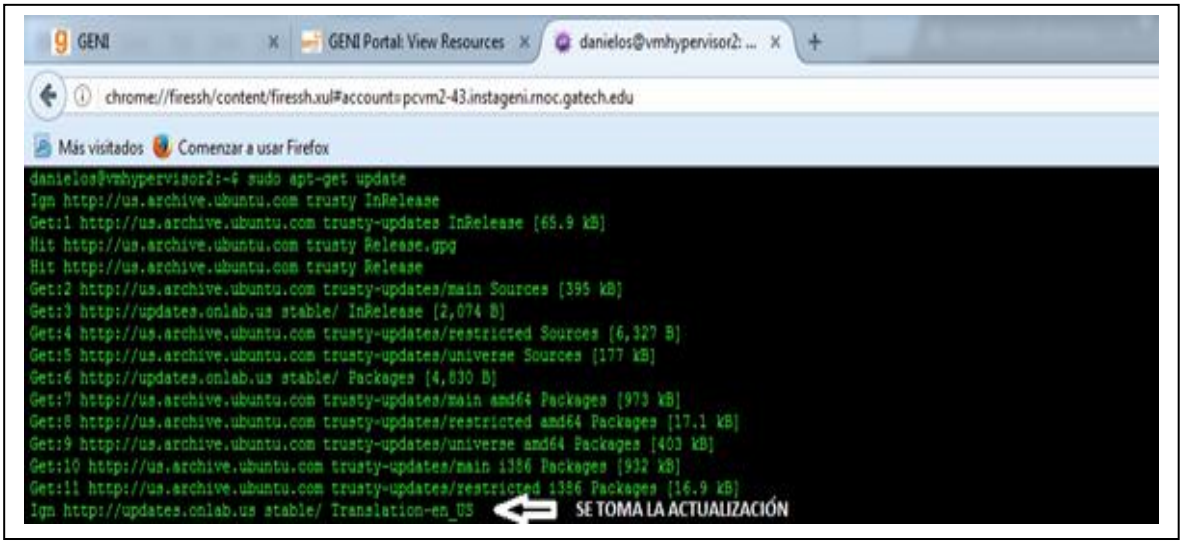


Imagen 5. Actualización de la base de datos de apt. Fuente: Fuente propia captura de pantalla plataforma Geni.

1.5 Paso 5. Instalación Flowvisor.

Para instalar Flowvisor se usa el comando **$ sudo apt-get install flowvisor**, este comando crea un usuario y grupo llamado flowvisor en el sistema.

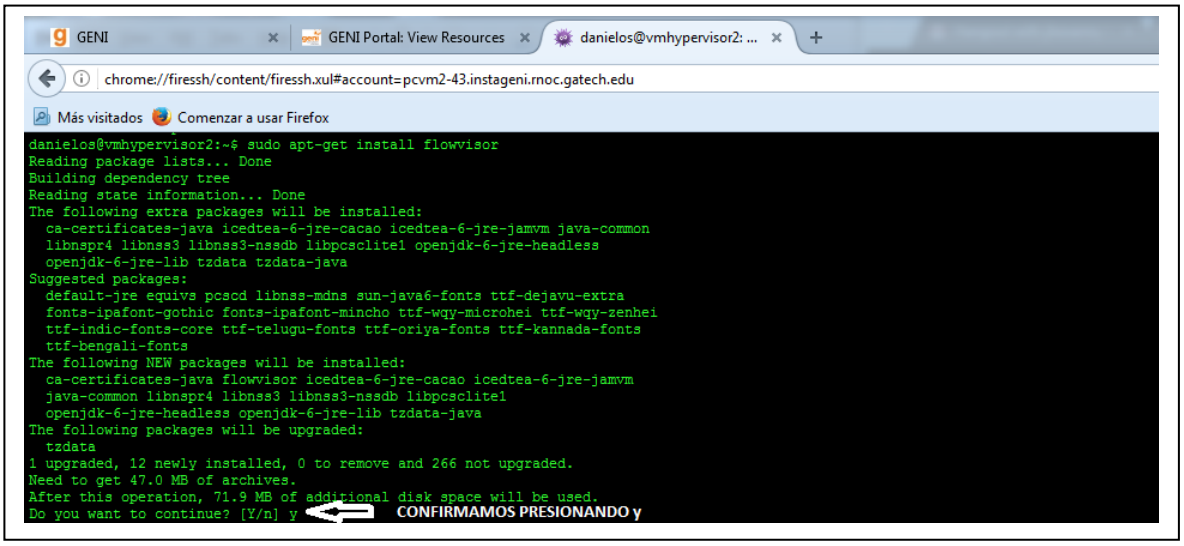


Imagen 6. Ejecución comando $ sudo apt-get install flowvisor. Fuente: Fuente propia captura de pantalla plataforma Geni.

ANEXO C. INSTALACION CONTROLADOR FLOODLIGHT EN GENI

1. Requisitos para instalación del controlador FLOODLIGHT

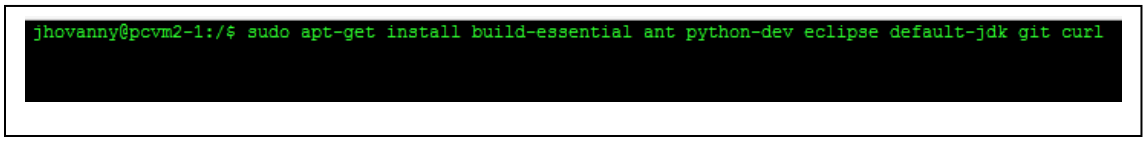
Esta instalación se realiza sobre un recurso virtual en la plataforma Geni con las siguientes características:

- Ubuntu 14.04.01 LTS
- GNU/Linux 3.13.0-33-generic x86_64

1.1 Paso 1. Instalación de paquetes esenciales.

Se instalan los paquetes esenciales necesarios para que el controlador pueda ejecutarse sin inconvenientes. Para eso se utiliza el comando: **sudo apt-get install build-essential ant python-dev eclipse default-jdk git curl** como se observa en la imagen 1

Imagen 1. Instalación de paquetes esenciales.



Fuente: Fuente propia captura de pantalla plataforma Geni

1.2 Paso 2. Instalación del controlador.

Una vez completado el paso 1 correctamente, se realiza la ubicación de la carpeta tmp y su correspondiente ingreso. Posterior a esto se comienza la instalación del controlador, con el siguiente comando: **git clone git://github.com/floodlight/floodlight.git** como se observa en la imagen 2.

Imagen 2. Instalación del controlador.

Fuente: Fuente propia captura de pantalla plataforma Geni.

1.3 Paso 3. Ejecución del comando git submodule init.

Al utilizar git clone se instala la última versión de Floodlight, que para este caso era la versión Master. Este creara una carpeta llamada floodlight sobre la cual se debe ingresar con el comando **cd floodlight**. Posterior a esto se ejecuta el comando: **git submodule init** como se observa en la imagen 3 para inicializar los submodulos, tales como la interfaz gráfica.

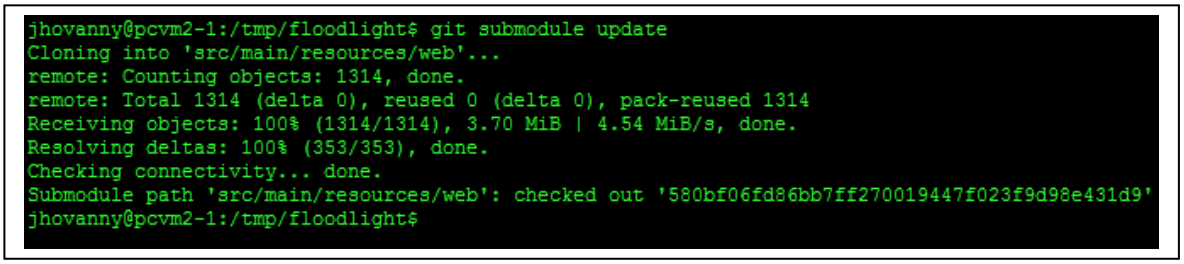
Imagen 3. Comando git submodule init.

Fuente: Fuente propia captura de pantalla plataforma Geni

1.4 Paso 4. Actualización de submodulos Floodlight.

Para tener el software más actualizados y funcional se actualizan los submodulos de Floodlight mediante el comando: **git submodule update** como se observa en la imagen 4.

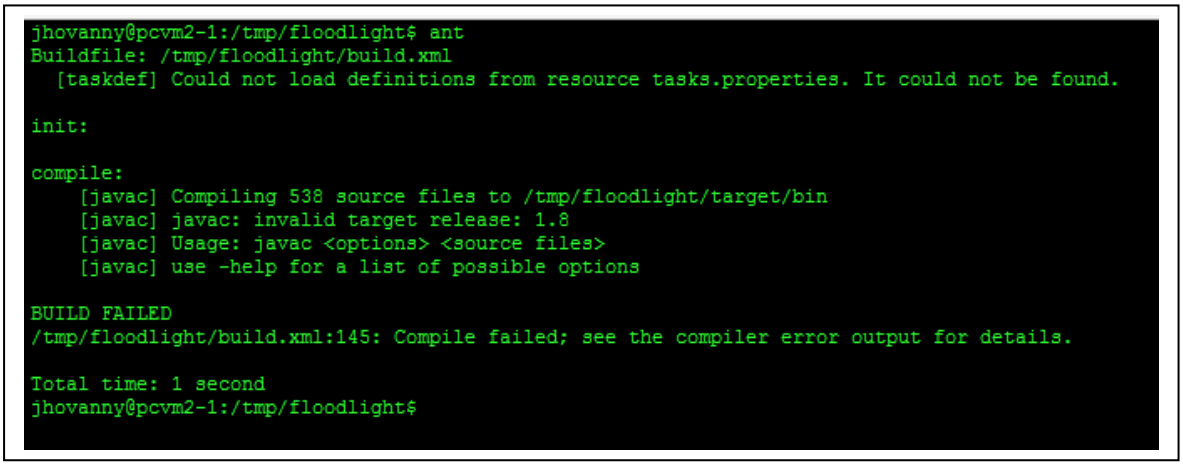
Imagen 4. Ejecución comando git submodule update.



Fuente: Fuente propia captura de pantalla plataforma Geni.

NOTA IMPORTANTE 1: Para poder ejecutar el próximo comando “ant” se debe actualizar la versión java de 7 a 8, ya que de lo contrario dicho comando arrojara un error, dicho comando sirve para construir las librerías del programa como se ve en la imagen 5.

Imagen 5. Error de Java versión 7.



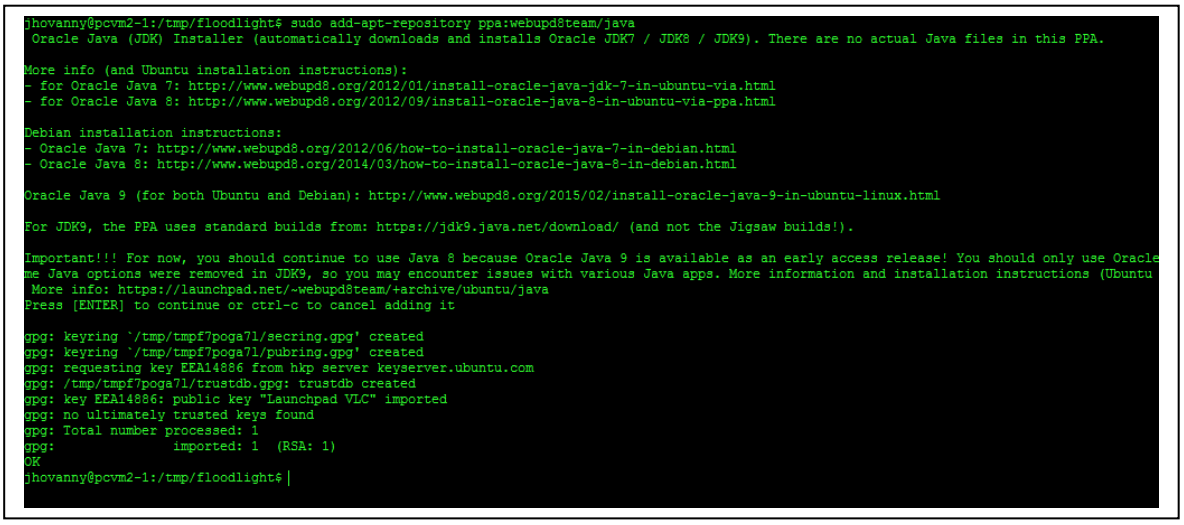
Fuente: Fuente propia captura de pantalla plataforma Geni.

Para actualizar la versión de java se utilizan los siguientes 3 comandos:

Comando 1:

Primera parte de la actualización **sudo add-apt-repository ppa:webupd8team/java** como se observa en la imagen 6.

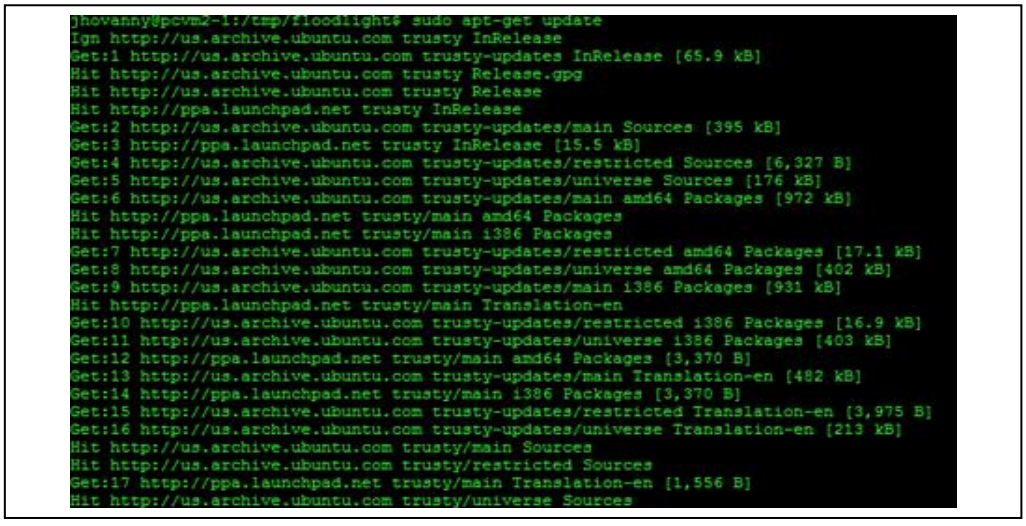
Imagen 6. Actualización java versión 8.



Fuente: Fuente propia captura de pantalla plataforma Geni.

Comando 2:
Después se ejecuta el siguiente comando: **sudo apt-get update** como se observa en la imagen 7 con el cual se actualizan archivos del repositorio.

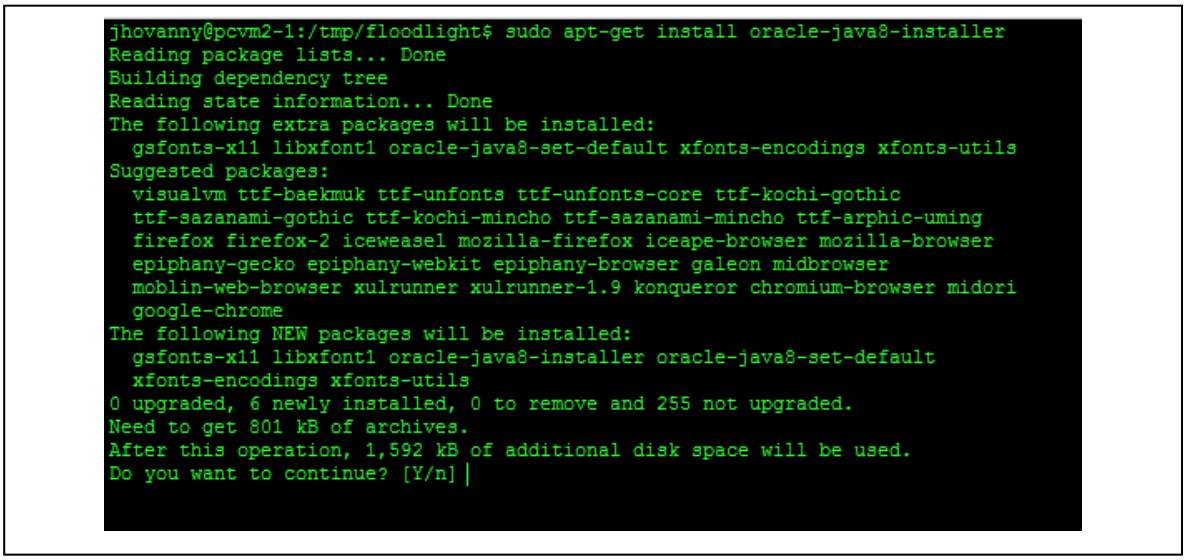
Imagen 7. Comando sudo apt-get update.



Fuente: Fuente propia captura de pantalla plataforma Geni.

Comando 3:
Para finalizar se ejecuta el siguiente comando: **sudo apt-get install oracle-java8-installer** como se observa en la imagen 8.

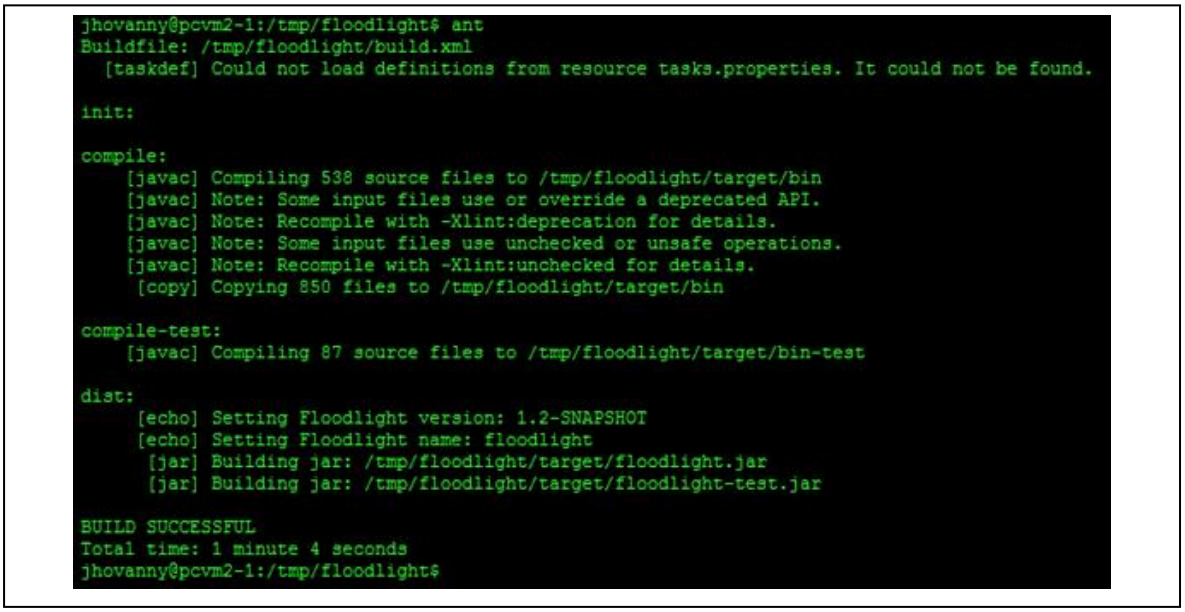
Imagen 8. Comando sudo apt-get install oracle-java8-installer.



Fuente: Fuente propia captura de pantalla plataforma Geni.

Como última parte del paso 4 presionamos “Y”, y seguimos las instrucciones para finalizar la actualización de Java. Una vez finalizada la instalación se ejecuta el comando “ant” y el resultado obtenido se puede observar en la imagen 9.

Imagen 9. Ejecución de comando ant.



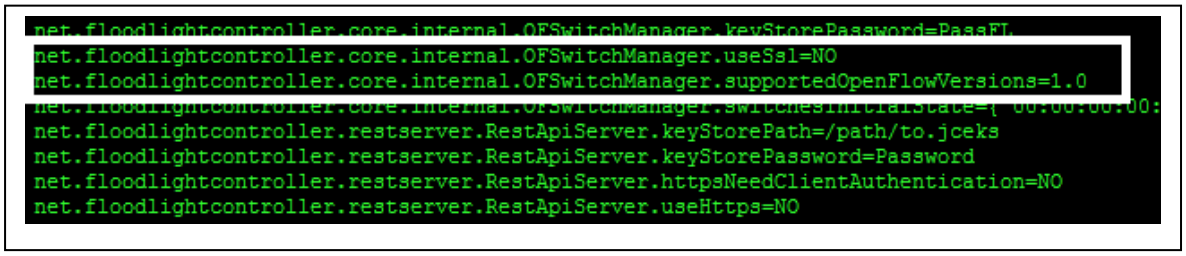
Fuente: Fuente propia captura de pantalla plataforma Geni.

Como se observa BUILD SUCCESSFUL, lo que indica que se ejecutó correctamente.
Y por último se cambian los permisos de la carpeta Floodlight para tener un acceso sin restricciones a las carpetas contenidas en dicho fichero mediante el comando: **sudo chmod 777 /var/lib/floodlight**

1.5 Paso 5. Modificación de script de compatibilidad Openflow versión 1.0 en controlador floodlight.

Para tener una conectividad adecuada entre el controlador Floodlight y el hipervisor Flowvisor, es necesario que ambos componentes funcionen bajo con la versión 1.0 del protocolo OpenFlow, para esto se debe ajustar el script **floodlightdefault.properties** en el controlador el cual se encuentra en la siguiente ruta dentro de la carpeta Floodlight: **src/main/resources/floodlightdefault** y modificar la línea:
net.floodlightcontroller.core.internal.OFSwitchManager.supportedOpenFlowVersions=1.0, 1.1, 1.2, 1.3, 1.4, 1.5
Para que solamente utilice la versión 1.0 del protocolo OpenFlow y se vea de la siguiente manera:
net.floodlightcontroller.core.internal.OFSwitchManager.supportedOpenFlowVersions=1.0 como se observa en la imagen 10.

Imagen 10. Línea a editar para cambiar versión de protocolo Openflow.

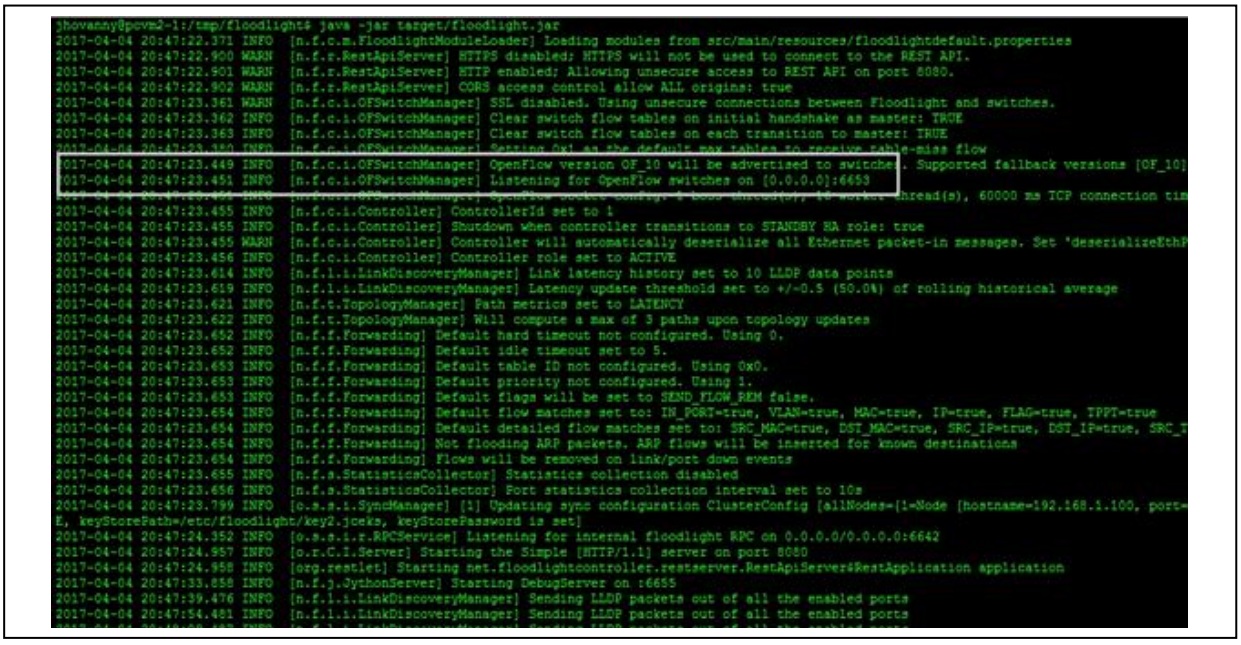


Fuente: Fuente propia captura de pantalla plataforma Geni.

1.6 Paso 6. Ejecución de controlador floodlight.

Para ejecutar el controlador Floodlight se usa el comando:
java –jar target/floodlight.jar y su correcta ejecución genera un log que permite ver que se está funcionando correctamente escuchando por el puerto 6653 como lo muestra la imagen 11.

Imagen 11. Log de funcionamiento controlador Floodlight.



Fuente: Fuente propia captura de pantalla plataforma Geni.

ANEXO D. CONFIGURACION DE SWITCH PARA IMPLEMENTACION DE COMUNICACIÓN CON HIPERVISOR O CONTROLADOR

1. Requisitos para la implementación de comunicación entre el switch switch y el hipervisor o controlador en la plataforma Geni.

Para realizar la implementación de comunicación es necesario contar con los recursos necesarios asignados por la plataforma Geni, ya sea el caso que se desee conectar el switch a un controlador o a un hipervisor. Para este ejemplo se realizará la conexión entre el switch y el hipervisor. Aclarando que el procedimiento para realizar la conexión entre el switch y el hipervisor, es el mismo para conectar el switch a un controlador.

Puntualmente para llevar a efecto ejemplo de conexión entre un switch compatible con protocolo Openflow y el hipervisor con software Flowvisor se solicitan los siguientes equipos en la plataforma Geni:

- Openflow OVS all XEN el cual es un recurso compuesto por un switch virtual Openvswitch compatible con el protocolo Openflow y a su vez este cuenta con tres nodos los cuales son

máquinas virtuales con un sistema operativo linux 14.04 preinstalado.

- 1 large XEN VM with public IP (IG) este consta de una máquina virtual con un sistema operativo Linux Ubuntu 14.04 preinstalado y una ip pública para el ejercicio de la experimentación. En este equipo se realizará la instalación del software Flowvisor como se muestra en el anexo B, con el cual dicho equipo cumplirá la función de hipervisor de red.

2. Configuración de los elementos de red.

Para obtener la convergencia de comunicación entre los elementos asignados por la plataforma Geni, es necesario que dichos componentes de la red se configuren de acuerdo a la documentación expuesta en el anexo C para el caso del hipervisor. Prestando especial cuidado a la versión del Protocolo Openflow, ya que la versión sobre la cual se realiza esta implementación de comunicación es la versión 1.0 del ya mencionado, esto debido a que el software Flowvisor no soporta versiones superiores de dicho protocolo. Por esta razón todos los elementos de red que se comunique con el hipervisor Flowvisor deben poder manejar dicha versión del protocolo.

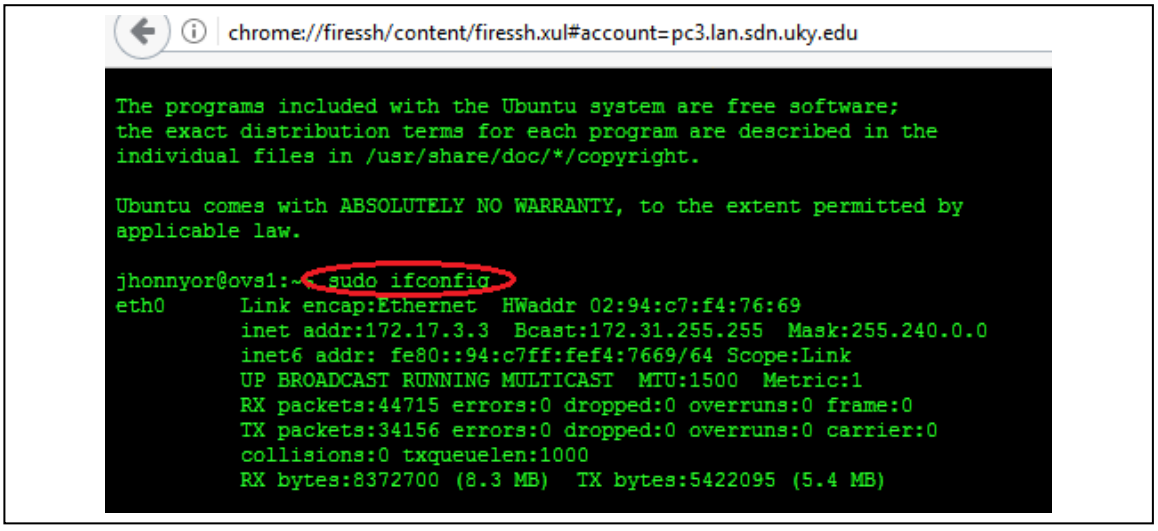
3. Configuración del Switch ovs.

Para realizar la configuración del Switch virtual ovs de manera tal que realice una comunicación exitosa con el hipervisor o el controlador, es necesario acceder a dicho switch por medio de la conexión ssh proporcionada por la plataforma Geni y seguir los siguientes pasos:

a) Identificación de interfaces de red del Switch

Para identificar las interfaces de red y el direccionamiento ip que posee el switch es necesario ejecutar el comando: **sudo ifconfig**
Tal como se observa en la imagen 1.

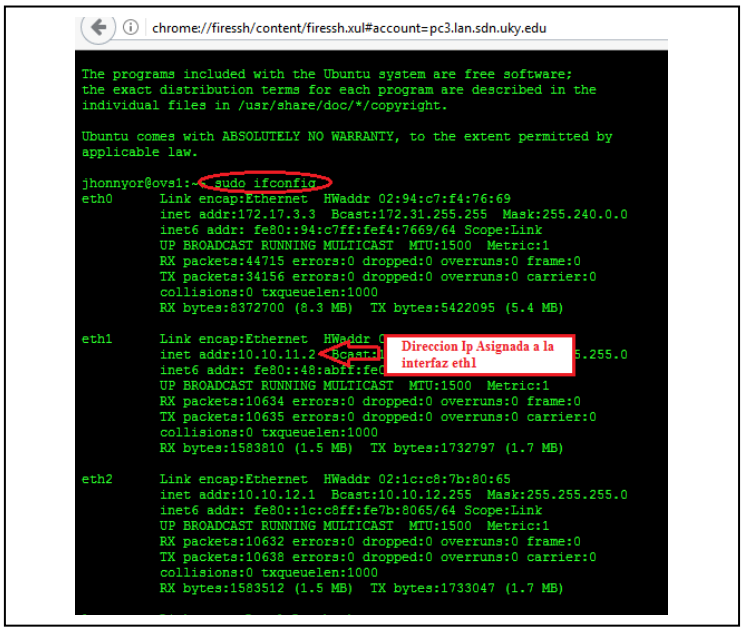
Imagen 1. Identificación de interfaces



Fuente: Fuente propia captura de pantalla plataforma Geni.

Dentro de la información obtenida con el comando ifconfig se observa las interfaces que posee el switch con sus respectivas direcciones ip, tal como se observa en la Imagen 2.

Imagen 2. Información de red del switch



Fuente: Fuente propia captura de pantalla plataforma Geni.

Dicha información de la red es importante y debe ser guardada, ya sea por medio de una captura de pantalla o la creación de un documento de texto, donde se relacione la información de las interfaces de red y su correspondiente dirección ip asignada.

b) Desacoplado el plano de datos del Switch

Para desacoplar el plano de datos del Switch se debe realizar el cambio de las interfaces de red de ipv4 a flujo de datos. Esto se realiza ejecutando el siguiente comando:

sudo ifconfig eth1 0

Este comando cambia el funcionamiento de la interfaz de red de ipv4 a manejo por flujos. Y se debe ejecutar en cada una de las interfaces que serán conectadas y controladas ya sea al hipervisor o al controlador de red. Para este ejemplo las interfaces **eth1** y **eth2**.

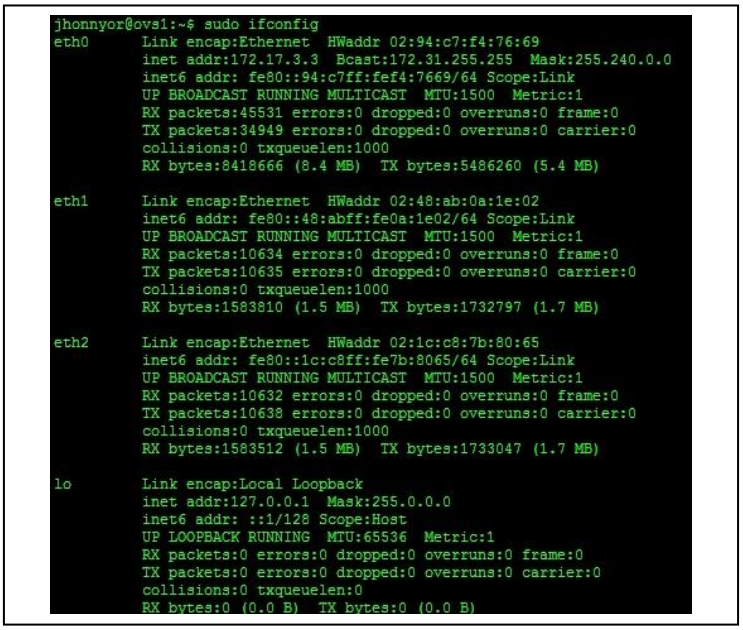
Es importante que bajo ninguna circunstancia se realice cambios a la interfaz de red **eth0** del switch, ya que esta interfaz es la encargada de realizar la conexión desde y hacia internet. Y a su vez es la que permite poder acceder al switch para realizar configuraciones como la que se está haciendo.

Para verificar la correcta ejecución del cambio en la interface se hace uso del comando:

sudo ifconfig

Obteniendo como resultado la perdida direcciones ipv4 en las interfaces modificadas, como se observa en la Imagen 3.

Imagen 3. Perdida de direccionamiento ipv4



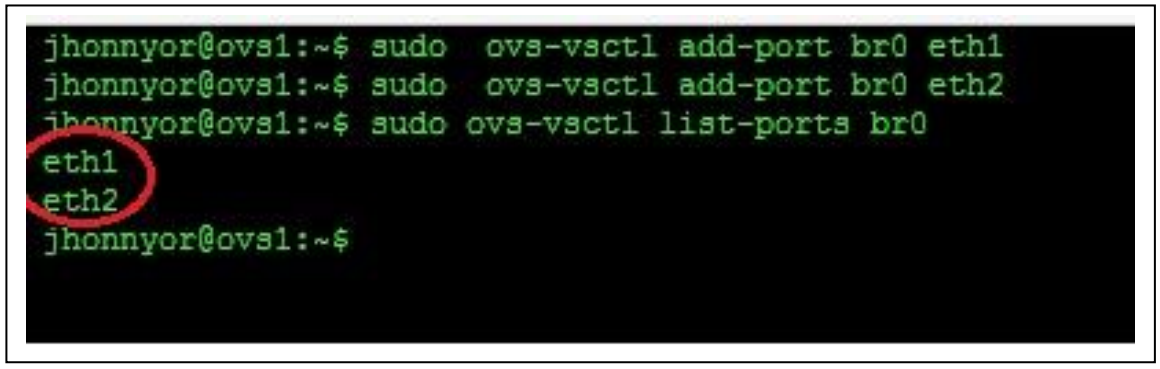
Fuente: Fuente propia captura de pantalla plataforma Geni.

c) Creación y configuración del Bridge

Una vez realizado el cambio de ipv4 a manejo por flujos en las interfaces del switch se ejecuta el siguiente comando:

sudo ovs-vsctl add-br br0
Este comando permite la creación de un puente que será el medio de comunicación, a través del cual las interfaces de red enviarán y recibirán información hacia el controlador o el hipervisor. Cabe aclara que **br0** es el nombre que se le asigno al bridge para este ejemplo.
Al bridge creado **br0** se le deben asociar las interfaces de red en el switch que serán controladas por el hipervisor o el controlador. Esto se realiza con el comando:
sudo ovs-vsctl add-port br0 eth1
Para este ejemplo se adicionan las dos interfaces de red a las cuales previamente se les deshabilito el direccionamiento de Ipv4. Estas fueron la **eth1** y la **eth2**.
Para verificar la correcta adición de las interfaces ya mencionadas al bridge se ejecuta el comando:
sudo ovs-vsctl list-ports br0
Como resultado se obtiene un listado de las interfaces de red agregadas al bridge **br0** para este ejemplo, como se observa en la Imagen 4.

Imagen 4. Adición de interfaces de red al bridge



Fuente: Fuente propia captura de pantalla plataforma Geni.

Posteriormente se ejecuta el comando:
sudo ovs-vsctl set bridge br0 other-config:datapath-id=0000000000000001
Este comando permite la asignación de un identificador al bridge ya configurado. Dicho identificador es necesario para realizar la identificación del switch en el hipervisor o controlador, y a su vez asociar las interfaces de red que se están conectado por medio de determinado bridge en el switch.
Para este ejemplo el bridge **br0** de este switch se identificará como **0000000000000001**. En el caso de tener más de un bridge en el switch lo cual es posible, es importante siempre tener un id identificador por cada

bridge, y de esta manera saber que interfaces son las que se están conectando al hipervisor o controlador.

Como se mencionó previamente para realizar la implementación de comunicación de forma adecuada, es necesario que todos los elementos de red funcionen bajo la versión 1.0 del protocolo Openflow. Para realizar esto en el caso del switch es necesario parametrizar el bridge, para que funcione bajo dicha versión del protocolo Openflow, ya que el equipo viene configurado por defecto para funcionar con la versión 1.3 del protocolo.

Para este ejemplo se debe configurar el **br0** de forma tal que funcione con la versión 1.0 del protocolo Openflow, y esto se logra ejecutando el siguiente código:

sudo ovs-vsctl set bridge br0 protocols=OpenFlow10

Es importante aclarar que la configuración expuesta en este anexo funciona para realizar la comunicación hacia un hipervisor o un controlador.

Dicho esto, como paso final para realizar la conexión del switch al hipervisor o al controlador, se debe configurar el bridge con la dirección ip del dispositivo con el cual se desea establecer comunicación sea el caso de un hipervisor o un controlador, y de igual manera el puerto de comunicación por medio del cual se realizará la conexión. Toda esta información se ingresa en la siguiente línea de comando:

sudo ovs-vsctl set-controller br0 tcp:<controller_ip>:6633

Para este ejemplo el bridge **br0** se configura apuntando hacia el hipervisor, al cual previamente se le instaló el software Flowvisor, como se explica en el anexo B de este proyecto, dicho recurso solicitado y asignado por la plataforma Geni cuenta con la ip publica **192.86.139.64** y realiza la escucha de los dispositivos que se desean conectar a él por medio del puerto **6633**, el cual fue asignado en la instalación del software Flowvisor.

El comando de configuración para bridge **br0** del switch hacia el hipervisor Flowvisor para este ejemplo quedaría de la siguiente forma:

sudo ovs-vsctl set-controller br0 tcp:192.86.139.64:6633

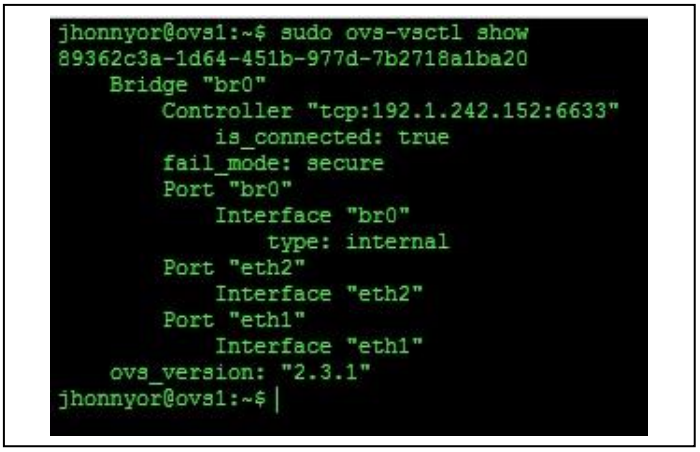
Como último paso de la configuración del switch es necesario configurar el modo de falla ante eventualidades, en otras palabras, si el controlador o el hipervisor se encuentran fuera de línea, el switch tiene dos modos de funcionamiento:

- Modo standalone
  En este caso el switch toma la responsabilidad del envió de los paquetes si el controlador se encuentra fuera de línea.
- Modo secure
  En este caso el controlador es el responsable del envió de los paquetes, y si este se encuentra fuera de línea, todos los paquetes son descartados.

Para configurar el modo de falla del switch se debe definir con cuál de las opciones se configurará el bridge.

Se recomienda el uso del modo secure, ya que este permite realmente saber cuándo el controlador o el hipervisor se encuentran fuera de línea debido a la constante perdida de paquetes.
Para configurar el modo de fallos **secure** en el switch se usa el comando:
sudo ovs-vsctl set-fail-mode br0 secure
Y en el caso de querer realizar la configuración de fallos en modo **standalone se** ejecuta el siguiente comando:
sudo ovs-vsctl set-fail-mode br0 secure
Para realizar la verificación del bridge, corroborar su correcta configuración, y conexión con el hipervisor o controlador se ejecuta el comando:
sudo ovs-vsctl show
Este comando entrega una descripción resumida de la configuración que contiene el switch, el estado de la conexión del mismo con la ip del hipervisor o el controlador según sea el caso, su modo de configuración de fallos y las interfaces habilitadas en el bridge. La descripción se observa en la Imagen 6.

Imagen 6. Estado del switch y su configuración.



Fuente: Fuente propia captura de pantalla plataforma Geni.

Como se mencionó al inicio las configuraciones expuesta en este anexo, pueden ser usadas para realizar la comunicación entre un switch y un controlador o un switch y un hipervisor.

ANXO E. EJECUCION DE INCRUSTACION MANUAL DE UNA RED VIRTUAL EN LA PLATAFORMA GENI

Requisitos para la incrustación de una red virtual en una red definida por software en la plataforma Geni.

Para realizar la incrustación de la red virtual es necesario contar con los recursos necesarios asignados por la plataforma Geni. Aclarando que la incrustación de la red virtual se realizara de manera manual.

Con el fin de llevar a efecto el ejemplo de incrustación se deben solicitar los siguientes equipos en la plataforma Geni:

- 3 Switches virtuales OpenvSwitch con Openflow activo, debidamente configurados e interconectados entre ellos.
- 1 máquina virtual con ip pública y software Floodlight correctamente configurado para realizar la función de controlador de red.
- 1 máquina virtual con ip pública y software Flowvisor correctamente instalado para realizar la función de hipervisor de red.

1. Configuración de los elementos de red.

Como se ha mencionado la convergencia de comunicación entre los elementos asignados por la plataforma Geni es muy importante, por lo tanto, es necesario que dichos componentes de la red se configuren de acuerdo a la siguiente documentación:

- Anexo B. Instalación controlador floodlight en Geni.
- Anexo C. Instalación controlador floodlight en Geni.
- Anexo D. Configuración de switch para implementación de comunicación con hipervisor o controlador.

Prestando especial cuidado a la versión del Protocolo Openflow, ya que la versión sobre la cual se realiza esta implementación de comunicación es la versión 1.0 del ya mencionado, esto debido a que el software Flowvisor no soporta versiones superiores de dicho protocolo. Por esta razón todos los elementos de red que se comunique con el hipervisor Flowvisor deben poder manejar dicha versión del protocolo.

2. Implementación de incrustación de una red virtual en la plataforma Geni.

Para llevar a cabo la incrustación de la red virtual de manera exitosa es clave verificar el correcto funcionamiento y conexión entre los elementos de la red como son el switch, el controlador y el hipervisor. De la misma manera todos los dispositivos se deben encontrar en línea y ejecución.

El núcleo principal para la ejecución de la incrustación de la red virtual se encuentra en el hipervisor Flowvisor, y es en este dónde se efectuarán una serie de pasos de configuración y verificaciones para la completar la incrustación. Dichos pasos son:

A. Verificación de switches conectados

Es importante aclarar como ya se mencionó previamente, que el software Flowvisor debe estar en ejecución para que los comandos que serán usados funcionen correctamente.

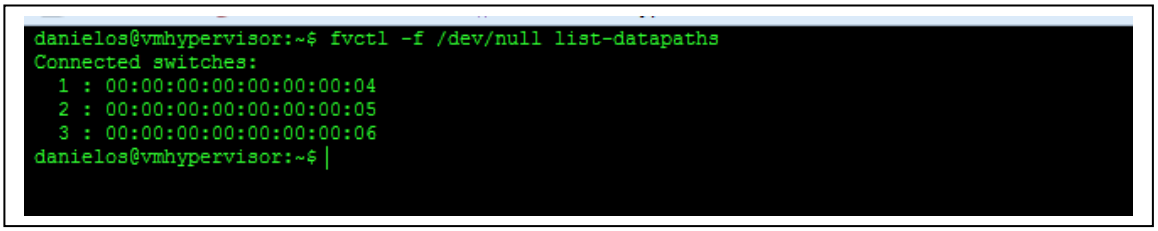
El hipervisor Flowvisor cuenta con una serie de comandos que permiten verificar que dispositivos se encuentran conectados y a su vez que interfaces de comunicación poseen.

El comando para obtener la información de los dispositivos conectados es:

fvctl –f /dev/null list-datapaths

Este comando muestra la lista de dispositivos conectados al hipervisor. Para este ejemplo se cuenta con 3 switches conectados al hipervisor como se observa en la imagen 1.

Imagen 1. Switches conectados.



Fuente: Fuente propia captura de pantalla plataforma Geni.

Como se observa en la imagen 1. Se obtienen los dpid correspondientes a los bridges de comunicación de los switches. Los dpid son asignados desde los switches tal como se explicó en el anexo D de este proyecto.

Estos dpid permiten identificar los switches que se encuentran en la red sustrato y son muy importantes, ya que por estos dpid permiten establecer donde se está realizando la incrustación de los nodos de la red virtual, en la red sustrato.

B. Información de los switches conectados

Se debe realizar la verificación de la información de cada uno de los switches conectados, ya que dicha información es vital para la incrustación de los enlaces de la red virtual.

Para este ejemplo se consulta la información del switch 1 con el dpid 00:00:00:00:00:00:00:04 con el comando:

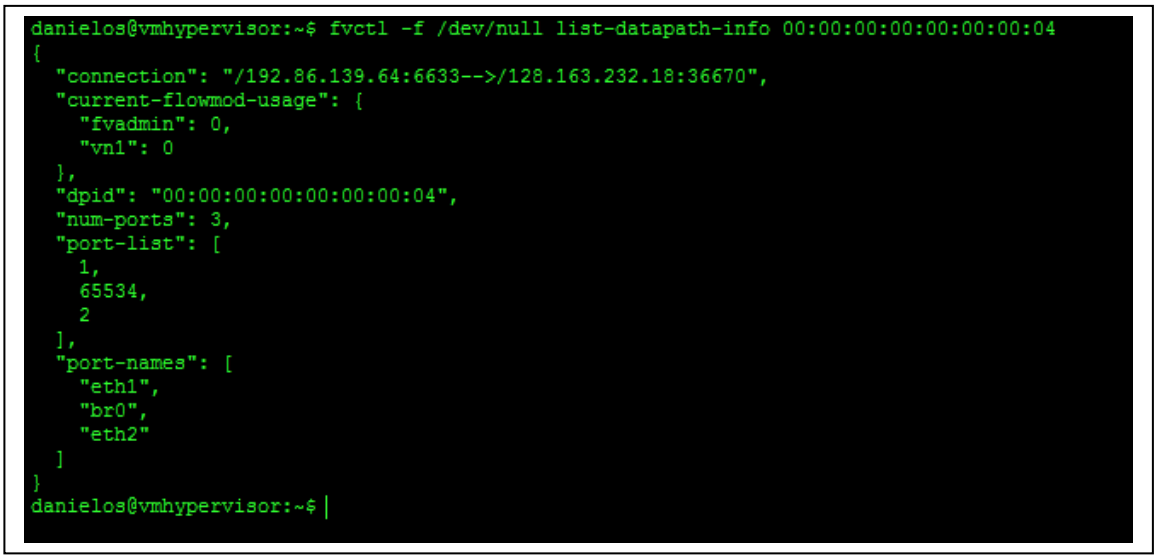
fvct fvctl –f /dev/null list-datapath-info y el **dpid** del switch a consultar.

La sentencia completa para este ejemplo seria:

fvctl –f /dev/null list-datapath-info 00:00:00:00:00:00:00:04

El resultado de la ejecución del comando se puede apreciar en la Imagen 2.

Imagen 2. Información completa del switch



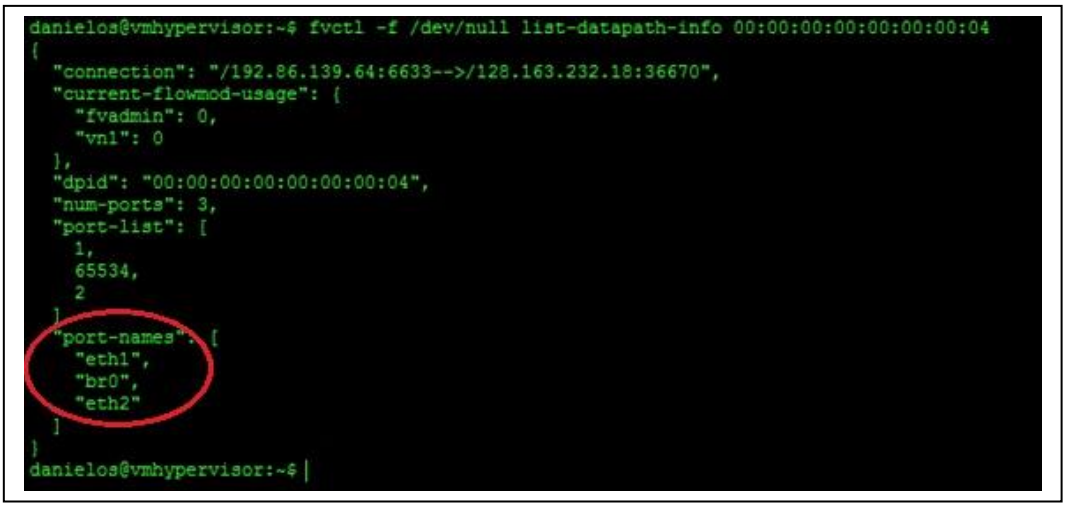
Fuente: Fuente propia captura de pantalla plataforma Geni.

Se sugiere documentar la información obtenida referente a la lista de puertos y sus respectivos nombres, ya que dicha información es el insumo para la ejecución de los comandos de creación de flowspaces en el hipervisor.

Dentro de la información obtenida con el comando anterior se destacan:

- Los nombres de las interfaces de red del switch como se observa en la Imagen 3.

Imagen 3. Nombres de las interfaces de red del switch



Fuente: Fuente propia captura de pantalla plataforma Geni.

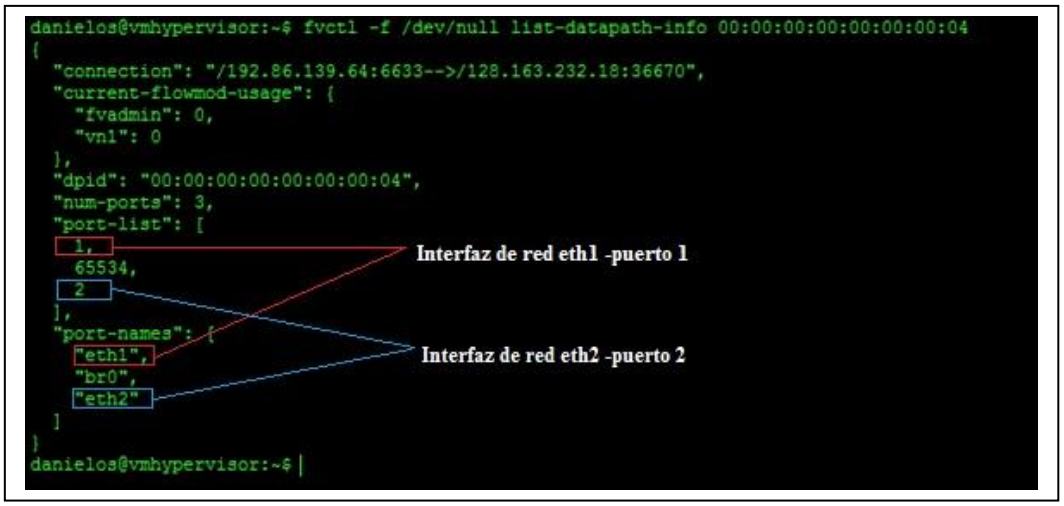
- La cantidad de puertos y el número por medio del cual se realiza la comunicación con el hipervisor Flowvisor como se observa en la Imagen 4.

Imagen 4. Puertos de comunicación con Flowvisor

Fuente: Fuente propia captura de pantalla plataforma Geni.

- Es importante entender que cada interfaz de red del switch posee un puerto por medio del cual realiza la comunicación. Tal como se observa en la Imagen 5.

Imagen 5. Puertos e interfaces de red switches.



Fuente: Fuente propia captura de pantalla plataforma Geni.

C. Creación del slice

La sintaxis de creación de un slice es la siguiente:

fvctl add-slice [options] **<nombre del slice> <ip del controlador> <email>**

La primera recomendación que se hace es consultar la documentación de cada comando con la siguiente sintaxis:

fvctl add-slice –h

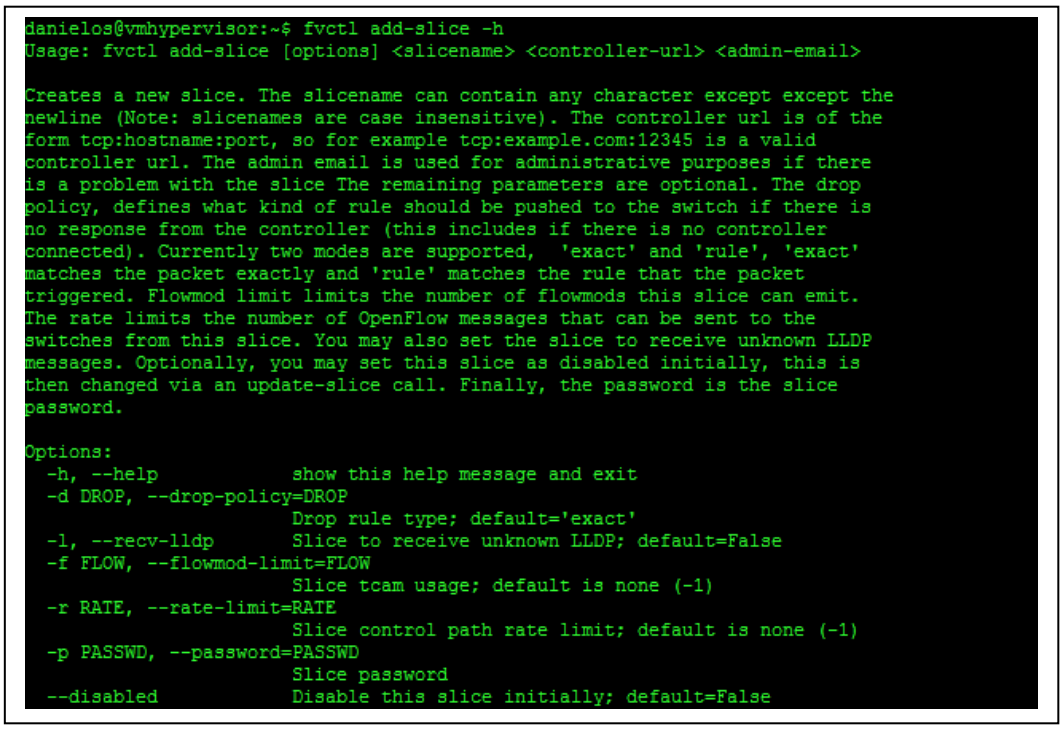
Como resultado de la ejecución se obtiene el menú de ayuda de Flowvisor, tal como se observa en la Imagen 6.

Como se puede observar se muestra una explicación de lo que el comando **add-slice** realiza, como también las opciones que se pueden utilizar con el mismo. Para una mayor eficiencia al crear un slice, con las opciones –password=”” se logró omitir el paso de ingreso de clave y hacer más fluida la consulta de información del slice.

Para este ejemplo de creación de un slice el comando completo seria:

fvctl -f /dev/null add-slice --password="" vne tcp: 128.104.159.128:6653 admin@prueba.com

Imagen 6. Sintaxis de creación slices -h.



Fuente: Fuente propia captura de pantalla plataforma Geni.

D. Visualización del slice

Para visualizar un Slice creado se ingresa el siguiente comando:

fvctl -f /dev/null list-slices

Nota: Se utiliza –f /dev/null para omitir el ingreso de la clave de Flowvisor la cual se configuro vacía, si no se utilizara esta sintaxis, Flowvisor preguntaría por la clave cada vez que se ejecute el comando, aunque esta se encuentre vacía.

E. Creación del flowspace

Los Flowspaces son las reglas que se van a definir dentro de los Slices, estas son las que se encargan de aislar las redes virtuales entre sí.

La sintaxis de creación es la siguiente:

fvctl add-flowspace [options] <nombre del flowspace> <dpid del switch> <prioridad> <in_port=puerto de la interfaz del switch> <slice=permisos>

Para este ejemplo se deben crear dos flowspaces

Flowspace 1:

fvctl -f /dev/null add-flowspace nodo1 00:00:00:00:00:00:00:04 100 in_port=1 vn1=7

Flowspace 2:
fvctl -f /dev/null add-flowspace nodo2 00:00:00:00:00:00:00:05 100 in_port=1 vn1=7

F. Visualización del flowspace

Para visualizar los flowspaces creados se utiliza la siguiente sintaxis:
fvctl -f /dev/null list-flowspace

Nota: Por último, aunque se explicó la creación de slices, flowspaces y como visualizarlos. Es importante familiarizarse con las funcionalidades propias de Flowvisor. La sintaxis del resto de comandos que posee este potente software hipervisor puede ser visualizada con la opción –h donde la cual muestra la funcionalidad de cada comando al igual que ejemplos de su uso.

ANEXO F. SCRIPT DE AUTOMATIZACION DE INCRUSTACION DE REDES VIRTUALES EN LA PLATAFORMA GENI

1. Requisitos para la implementación del script de automatización de creación de redes virtuales en la plataforma Geni.

Para realizar la implementación del script de automatización de incrustación de las redes virtuales es necesario contar con los recursos necesarios asignados por la plataforma Geni.
Con el fin de llevar a efecto el ejemplo de las incrustación se deben solicitar los siguientes equipos en la plataforma Geni:

- 3 Switches virtuales OpenvSwitch con Openflow activo, debidamente configurados e interconectados entre ellos.
- 4 máquina virtual con ip pública y software Floodlight correctamente configurado para realizar la función de controlador de red.
  Nota: Es importante tener en cuenta que para esta implementación los controladores deben cambiar su puerto de comunicación por defecto 6633 al puerto 6653, debido a que en el script de incrustación se definió al 6653 como el puerto de escucha para los controladores.
- 1 máquina virtual con ip pública y software Flowvisor correctamente instalado para realizar la función de hipervisor de red.

2. Configuración de los elementos de red.

Como se ha mencionado la convergencia de comunicación entre los elementos asignados por la plataforma Geni es muy importante, por lo tanto, es necesario que dichos componentes de la red se configuren de acuerdo a la siguiente documentación:

- Anexo B. Instalación controlador floodlight en Geni.
- Anexo C. Instalación controlador floodlight en Geni.
- Anexo D. Configuración de Switch para implementación de comunicación con hipervisor o controlador.

Prestando especial cuidado a la versión del Protocolo Openflow, ya que la versión sobre la cual se realiza esta implementación de comunicación es la versión 1.0 del ya mencionado, esto debido a que el software Flowvisor no soporta versiones superiores de dicho protocolo. Por esta razón todos los elementos de red que se comunique con el hipervisor Flowvisor deben poder manejar dicha versión del protocolo.

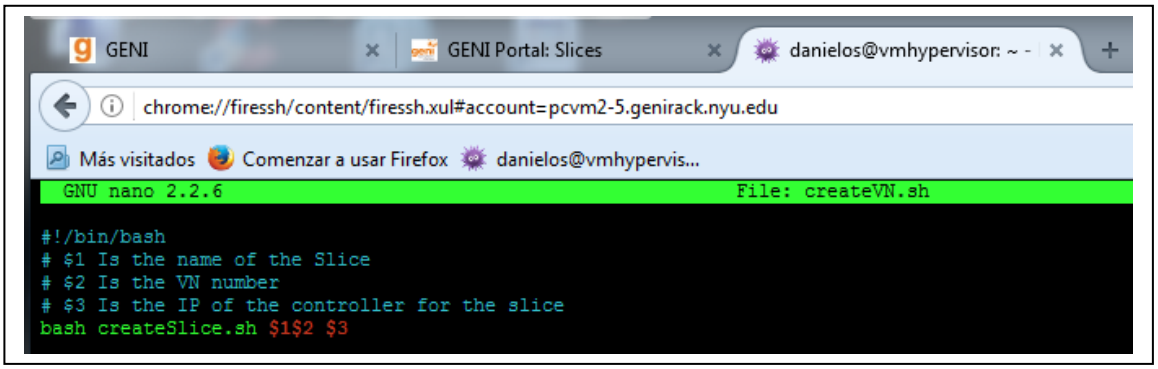
3. Script de automatización para la creación de red virtual, en la plataforma Geni.

El script desarrollado para la creación de redes virtuales **createVN.sh** empieza llamando al script **createSlice.sh** que recibe 3 parámetros los cuales son:

- Primer parámetro nombre de la red virtual.
- Segundo parámetro número de la red virtual
- Tercero parámetro IP del controlador

Como se logra observar en la Imagen 1.

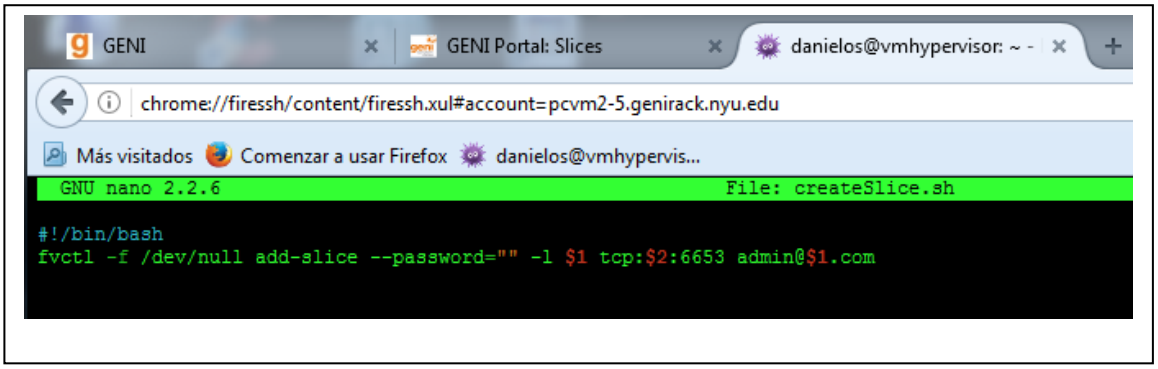
Imagen 1. Script createVN.sh captura de parámetros.



Fuente: Fuente propia captura de pantalla plataforma Geni.

El script **createSlice.sh** recibe los 3 parámetros obtenidos del script **createVN.sh** e invoca la sintaxis de creación de un slice o red virtual.
Como se ve en la Imagen 2.

Imagen 2. Script createSlice.sh creación de slice.



Fuente: Fuente propia captura de pantalla plataforma Geni.

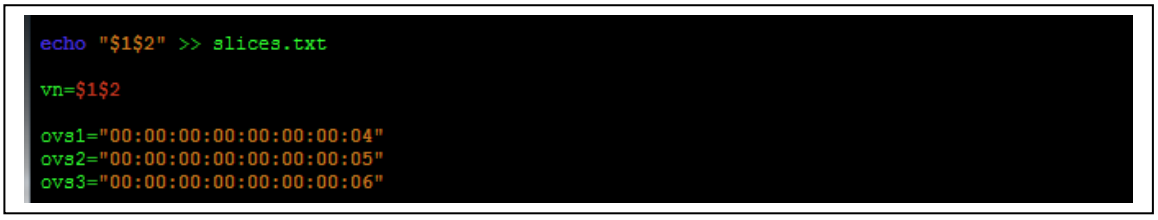
Es importante mencionar que el puerto de escucha del controlador debe ser 6653, si se configuro otro puerto de escucha en el controlador, es necesario realizar el cambio del puerto en el controlador o debe ser cambiado directamente en este script.

Después de crearse el Slice, el script createVN.sh almacena el nombre de la red virtual el cual fue el primer parámetro ingresado. El almacenamiento de los parámetros se realiza teniendo en cuenta la concatenación de los dos primeros elementos recibidos. Este almacenamiento será utilizado más adelante por el script de borrado de redes virtuales.
Se definen 4 variables que son:

I. **vn**
   Es la encargada de almacenar la red virtual y el número de esta.
II. **ovs1**
   Encargada de almacenar el dpid del switch 1
III. **ovs2**
   Encargada de almacenar el dpid del switch 2
IV. **ovs3**
   Encargada de almacenar el dpid del switch 3

Como se observa en la Imagen 3.

Imagen 3. Script CreateVN.sh variables dpid switches.

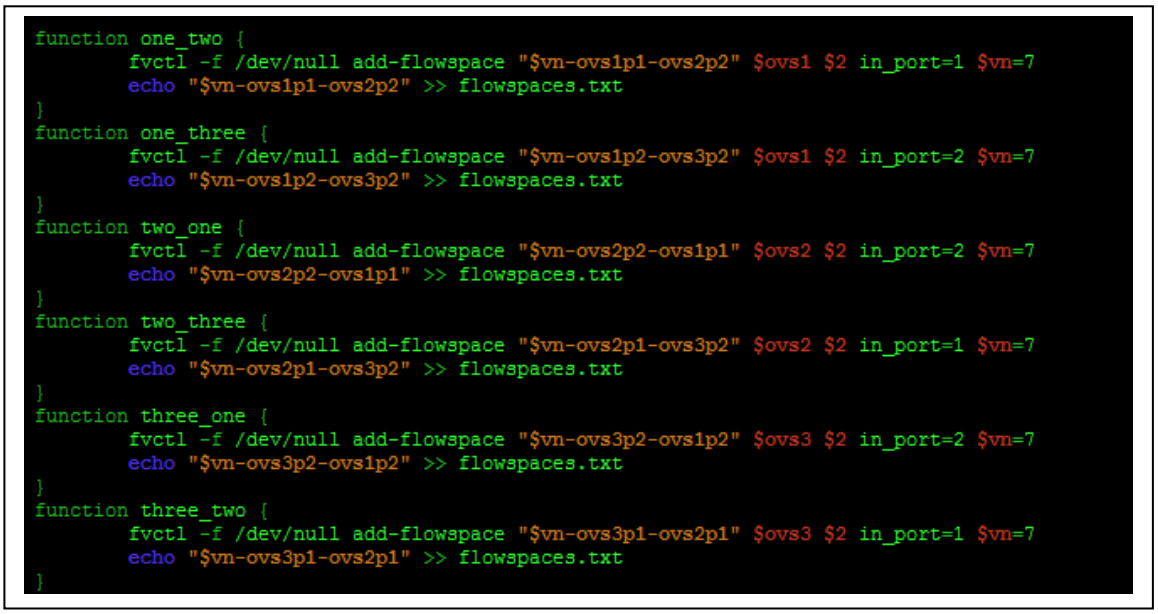


Fuente: Fuente propia captura de pantalla plataforma Geni.

Nota: Los switches pueden ser n, pero para pruebas fueron configurados previamente.

Se define una serie de funciones, las cuales serán las encargadas de realizar el mapeo de switches de cada red virtual creada, dependiendo de donde vayan a residir los nodos virtuales. Como se ve en las Imágenes 4 y 5.
El proceso de mapeo se realiza más adelante alimentando el script con los resultados del BNPA-VNE el cual es el algoritmo de incrustación de redes virtuales trabajado por el ingeniero Néstor Álzate, nuestro tutor de este proyecto.

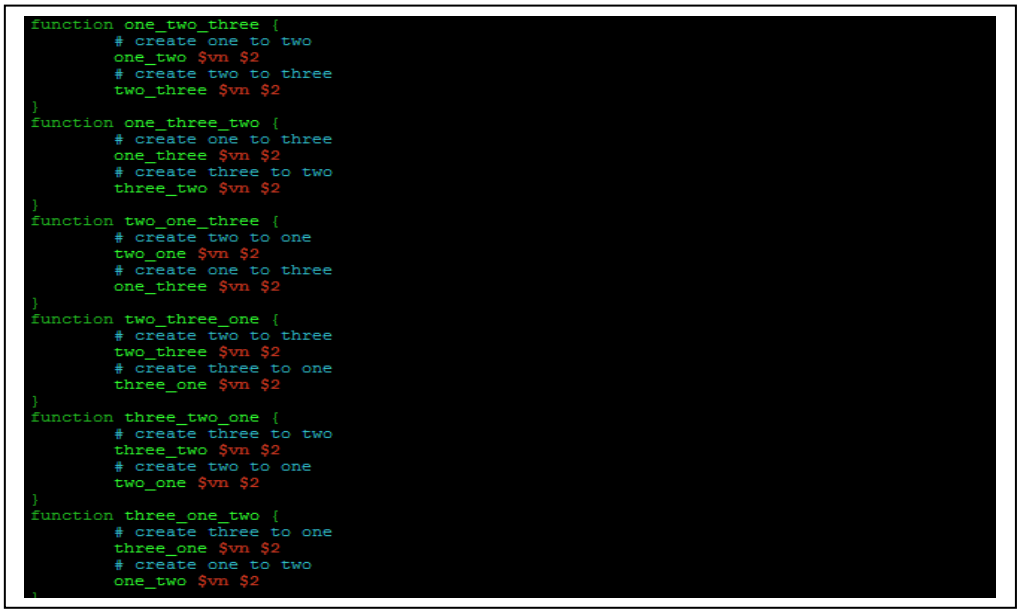
Imagen 4. Script CreateVN.sh funciones de mapeo 1.



```
function one_two {
        fvctl -f /dev/null add-flowspace "$vn-ovs1p1-ovs2p2" $ovs1 $2 in_port=1 $vn=7
        echo "$vn-ovs1p1-ovs2p2" >> flowspaces.txt
}
function one_three {
        fvctl -f /dev/null add-flowspace "$vn-ovs1p2-ovs3p2" $ovs1 $2 in_port=2 $vn=7
        echo "$vn-ovs1p2-ovs3p2" >> flowspaces.txt
}
function two_one {
        fvctl -f /dev/null add-flowspace "$vn-ovs2p2-ovs1p1" $ovs2 $2 in_port=2 $vn=7
        echo "$vn-ovs2p2-ovs1p1" >> flowspaces.txt
}
function two_three {
        fvctl -f /dev/null add-flowspace "$vn-ovs2p1-ovs3p2" $ovs2 $2 in_port=1 $vn=7
        echo "$vn-ovs2p1-ovs3p2" >> flowspaces.txt
}
function three_one {
        fvctl -f /dev/null add-flowspace "$vn-ovs3p2-ovs1p2" $ovs3 $2 in_port=2 $vn=7
        echo "$vn-ovs3p2-ovs1p2" >> flowspaces.txt
}
function three_two {
        fvctl -f /dev/null add-flowspace "$vn-ovs3p1-ovs2p1" $ovs3 $2 in_port=1 $vn=7
        echo "$vn-ovs3p1-ovs2p1" >> flowspaces.txt
}
```

Fuente: Fuente propia captura de pantalla plataforma Geni.

Imagen 5. Script CreateVN.sh funciones de mapeo 2.



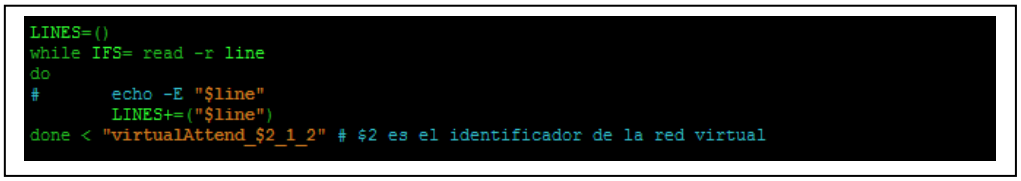
```
function one_two_three {
        # create one to two
        one_two $vn $2
        # create two to three
        two_three $vn $2
}
function one_three_two {
        # create one to three
        one_three $vn $2
        # create three to two
        three_two $vn $2
}
function two_one_three {
        # create two to one
        two_one $vn $2
        # create one to three
        one_three $vn $2
}
function two_three_one {
        # create two to three
        two_three $vn $2
        # create three to one
        three_one $vn $2
}
function three_two_one {
        # create three to two
        three_two $vn $2
        # create two to one
        two_one $vn $2
}
function three_one_two {
        # create three to one
        three_one $vn $2
        # create one to two
        one_two $vn $2
}
```

Fuente: Fuente propia captura de pantalla plataforma Geni.

Para realizar el proceso de incrustación es necesario proveer al script con la información necesaria del mapeo de nodos y enlaces, para esto en el script se crea un arreglo llamado LINES el cual se alimenta con la información

encontrada en el archivo virtualAttend, este archivo es uno de los entregados por el algoritmo BNPA-VNE e indica la ruta de enlaces de la red virtual creada. Tal como se observa en la Imagen 6.

Imagen 6. Script CreateVN.sh lectura de ruta de mapeo de enlaces.

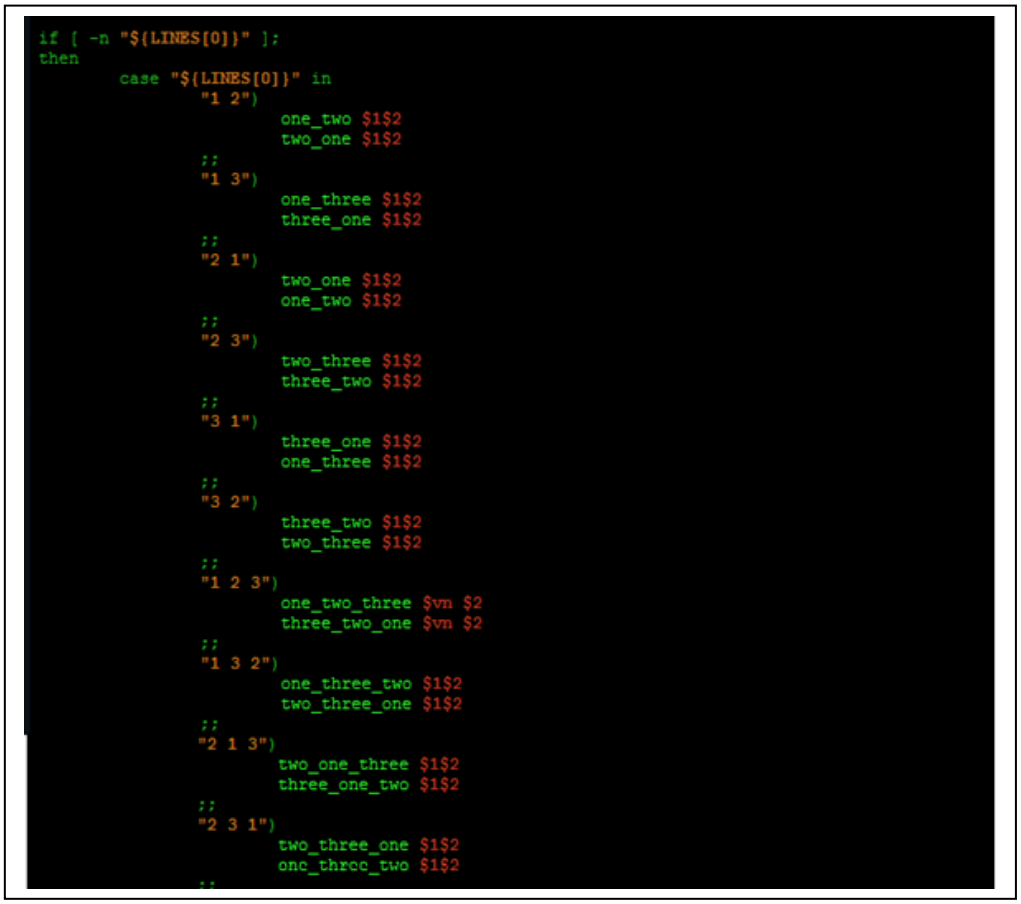


```
LINES=()
while IFS= read -r line
do
#       echo -E "$line"
        LINES+=("$line")
done < "virtualAttend_$2_1_2" # $2 es el identificador de la red virtual
```

Fuente: Fuente propia captura de pantalla plataforma Geni.

Para realizar la creación de la red virtual se utilizó un switch case el cual lee la información almacenada en el arreglo LINES, el cual fue alimentado previamente por el archivo virtualAttend y posee la ruta que debe tomar la red virtual al ser creada al igual que los nodos sustrato que va a utilizar.
La estructura del switch case usado se puede ver en la Imagen 7.

Imagen 7. Script CreateVN.sh creación de red virtual usando switch case.



```
if [ -n "${LINES[0]}" ];
then
        case "${LINES[0]}" in
                "1 2")
                        one_two $1$2
                        two_one $1$2
                ;;
                "1 3")
                        one_three $1$2
                        three_one $1$2
                ;;
                "2 1")
                        two_one $1$2
                        one_two $1$2
                ;;
                "2 3")
                        two_three $1$2
                        three_two $1$2
                ;;
                "3 1")
                        three_one $1$2
                        one_three $1$2
                ;;
                "3 2")
                        three_two $1$2
                        two_three $1$2
                ;;
                "1 2 3")
                        one_two_three $vn $2
                        three_two_one $vn $2
                ;;
                "1 3 2")
                        one_three_two $1$2
                        two_three_one $1$2
                ;;
                "2 1 3")
                        two_one_three $1$2
                        three_one_two $1$2
                ;;
                "2 3 1")
                        two_three_one $1$2
                        one_three_two $1$2
                ;;
```

Fuente: Fuente propia captura de pantalla plataforma Geni.

Para finalizar la forma de ejecutar el script es a través del comando **autovne.sh** el cual invoca el script createVN.sh, con la función time antes del script. Como se observa en la Imagen 8.

Imagen 8. Código Script autovne.sh

```
#!/bin/bash
time bash createVN.sh $1 $2 $3
```

Fuente: Fuente propia captura de pantalla plataforma Geni.

Con esta forma de ejecución y con los echo realizados al final de createVN.sh como se ven en la Imagen 9.

Imagen 9. Echo medición de tiempos.

```
echo -e "real: Tiempo que se toma en terminar" '\n'

echo -e "user: Tiempo que se toma en ejecutar el programa" '\n'

echo -e "sys: Tiempo que se demora en procesar" '\n'
```

Fuente: Fuente propia captura de pantalla plataforma Geni.

Se pueden medir los tiempos de ejecución al igual que los tiempos de procesamiento del script **autovne.sh**