Emulación de SNDs usando Mininet

Rodrigo Manríquez Peralta, Estudiante, Universidad Técnica Federico Santa María.

***Resumen*—En la actualidad, la actualización de la infraestructura de red hacia nuevas tecnologías y protocolos es una tarea engorrosa y costosa, ya que si los cambios son significativos se requiere reemplazar los equipos por completo. En ese sentido, se puede decir que hoy en día los equipos de red tienen poca flexibilidad. Un desafío de los desarrolladores e investigadores hoy en día radica en pensar cómo solucionar ésta problemática. Con éste enfoque en mente es que nacen las denominadas Redes definidas por software (SDN), que són mucho menos dependientes del hardware, y por tanto otorgan mayor flexibiliad. Dentro de éste nuevo paradigma es que nace llamada Mininet, la cuál permite a los desarrolladores emular redes de forma virtual. Para evaluar las capacidades de Mininet, se realizarán pruebas de emulación, interactividad, y escalabilidad (benchmarking) sobre una red emulada.**

***Keywords*—SND, Mininet, emulación, benchmarking**

I. Introducción

Una forma de otorgar flexibilidad a los equipos de red, de forma que no se requiera reemplazarlos en caso de querer implementar un nuevo protocolo o tecnología a la red, es el hacer que éstos cambios se puedan introducir cambiando la programación de los equipos. Esto los hace dependientes del software, y no del hardware. Una SDN (Software Defined Network) es una red que implementa elementos programables en su configuración, de forma que éstas pueden ser reprogramadas y actualizadas para implementar nuevos protocolos o tecnologías. En este contexto aparece *Mininet*, que se define como una platadorma que permite emular redes de forma virtual, y en que su configuración puede ser programada de forma externa. Entre otras características, las redes creadas por Mininet ofrecen escalabilidad, interactividad y flexibilidad. La relación entre Mininet y las SNDs se establece por su similitud al trabajar con nodos que físicamente no existen, pudiendo en ambos casos definir sus configuraciones por software. Una característica importante de Mininet es que permite emular redes definidas por software con todas sus características. En particular, permite trabajar con el protocolo *Openflow*, usado por las SDNs.

II. Software-defined Networks

Una SND (Red definida por software) es un paradigma de redes de computadores que introduce el uso de más recursos programables y menos dependencia del hardware. Ésta característica permite otorgar flexibilidad a la redes, pudiendo realizar cambios a éstas de forma gradual, sin costos y de forma más simplificada.

Una SND consta de 3 elementos importantes. En primer lugar, está el *Controlador*, el cuál actúa como interfaz entre el administrador/desarrollador y los elementos de la red (como routers y switches). El controlador se encarga, además, de ejecutar comandos para control de direccionamiento y ruteo. Esto hace que la administración de la red se centralice, dependiendo más del controlador, otorgando además un mayor nivel de abstracción de la red desde el punto de vista del administrador. Los comandos de control permiten implementar, por ejemplo, reglas de monitoreo o nuevas políticas de seguridad.

El segundo elemento de una SND son los *programable switching elements*, o elementos de direccionamiento/enrutamiento programables. Éstos son los routers y switches en la red, que además de realizar los enlaces a nivel de capa de red o enlace, tienen la característica de ser programables.

Por último, se encuentran los *protocolos de comunicación entre* controlador y switches. Uno de los protocolos más populares de comunicación en éste contexto es OpenFlow. Éste protocolo permite que el ruteo de "bajo nivel" se siga realizando en los switches, pero el de "alto nivel" sea realizado por un servidor central (controlador). OpenFlow permite que los switches y el controlador se comuniquen en éste contexto, y haciendo que además el reenvío de paquetes sea centralizado.

III. Mininet

Mininet es una plataforma que permite crear redes virtuales a gran escala de forma rápida y eficiente gracias al uso de una característica conocida como virtualización ligera (light-weight virtualization). Ésta plataforma es muy práctica para el estudio de SNDs, debido a que permite implementar nodos con protocolo OpenFlow. Entre las características de Mininet se encuentran:

o Flexibilidad: Topologías y características nuevas se pueden setear por software usando Lenguajes de programación y SO comunes.
o Aplicabilidad: Correctas implementaciones (prototipos) se pueden implementar en redes basadas en Hardware sin cambiar su código fuente.
o Interactividad: Administración y simulación ocurren en tiempo real.
o Escalabilidad: Ambiente de prototipado es escalable a redes grandes con un solo ccomputador.
o Realista: Comportamiento del prototipo representa comportamiento real con alto grado de confianza. Esto permite que aplicaciones y pilas de protocoles se puedan usar sin cambiar código.
o Compartible: Prototipos pueden ser compartidos para que otros colaboradores y desarrolladores puedan correr y modificar el experimento.

Gracias a sus capacidades de emulación, Mininet se convierte en una herramienta práctica y útil a la hora de hacer pruebas con redes definidas por software. A continuación se presentarán diversas pruebas para demostrar las capacidades de emulación de Mininet.

III-A. Emulación de Redes

Mininet incluye, entre otras características, una *interfaz por línea de comandos* (CLI), bibliotecas y una API para *Python*, y aplicaciones útiles para el análisis de tráfico de redes (Wireshark).

Para crear una red simple, se debe ejecutar el siguiente comando como super usuario en la terminal:

sudo mn –topo (topología),(N. de hosts)

Donde el campo *topología* especifica la forma de la red, que puede ser *tree* (árbol), *simple*, *linear*, etc.

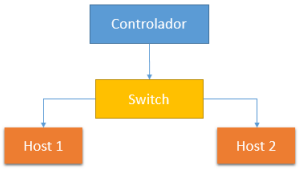

Figura 1. Red simple con estructura *linear*

El comando *mn* también admite otras entradas para caracterizar la red según se desee. Por ejemplo, se puede limitar el ancho de banda en los enlaces o el número de puertos de los switches, o se puede asignar un delay a los enlaces, etc. Notar que, por defecto, h1 es asignado con la IP 10.0.0.1, y h2 con 10.0.0.2.

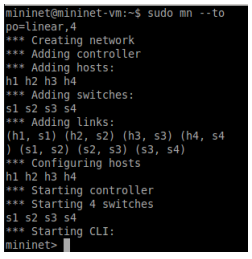

Figura 2. Creación de una red en Mininet

Los hosts creados de forma virtual pueden interactuar entre sí. Por ejemplo, se puede hacer un ping desde el host *h1* al host *h2* usando el comando *h1 ping h2*. Esto también se puede realizar abriendo una terminal en el host *h1* usando el comando *xterm*.

h1 xterm

Mininet admite más opciones sobre los nodos. Por ejemplo, se puede montar un servidor web en uno de los hosts, mientras que el otro hace un requerimiento HTTP al servidor creado.

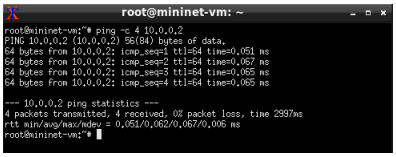

Figura 3. Ping a h2 usando *xterm*

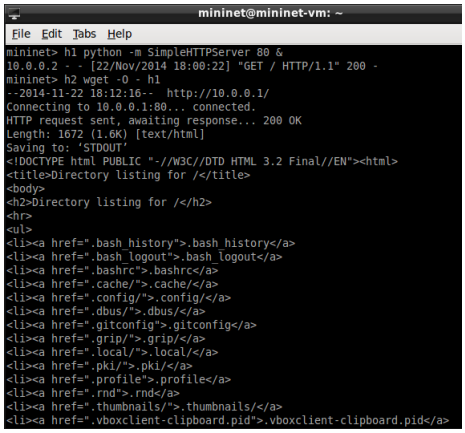

Figura 4. Servidor web montado en h1

Es importante aclarar que ésto es una emulación de hosts creados de forma virtual, por lo que los paquetes enviados son reales y pueden ser vistos por un sniffer como Wireshark.

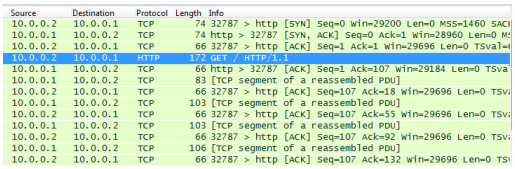

Figura 5. Requerimiento GET en Wireshark

Como se ve, las IPs del requerimiento GET de HTTP corresponden a las asignadas por Mininet. El hecho de que Wireshark pueda ver los paquetes sugiere que los hosts pueden tener conectividad a Internet. Esto se comprobará a continuación.

III-B. Pruebas de Interactividad

Para usar elementos más complejos dentro de Mininet se usan las librerías y API de *Python*. Los scripts creados con *Python* permiten crear redes y configuraciones más complejas, y a su vez guardarlas para que puedan ser reproducidas cada vez que se ejecute el script. Notar que una vez que se sale de Mininet, la red es destruida, y por tanto el historial de la red y los cambios realizados también se pierden.

En éste caso se usará un script llamado *nat.py* para darle conectividad a los hosts usando una tabla NAT.

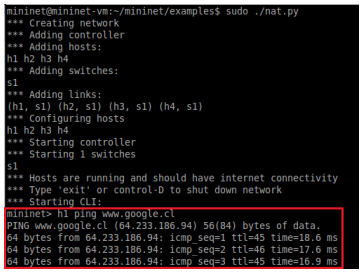

Figura 6. Host h1 con conectividad a Internet

El host *h1* es capaz de realizar un ping a una página web de Internet con éxito. También se puede montar un servidor web sobre el host *h1* como se mostró anteriormente. Ahora este host es visible desde el explorador de la máquina local.

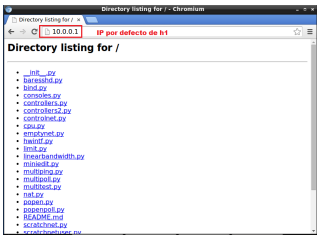

Figura 7. Página web montada por h1

III-C. Pruebas de Rendimiento

Para evaluar la capacidad de escalar grandes redes con numerosos hosts y switches es que se realizan pruebas de escalabilidad (benchmarking). Para realizar estas pruebas se usó el siguiente script en bash que crea nodos de forma concurrente con topología *tree*. El script entrega por pantalla el tiempo que tarda en crear y destruir la red, además del uso de memoria.

Las referencias [1] indican que el tiempo de creación de los nodos va aumentando de forma exponencial conforme crece el número de nodos.

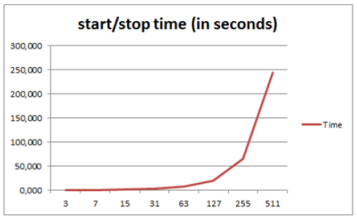

Figura 8. Tiempo de creación de los nodos según referencia

Las pruebas se realizaron sobre una máquina virtual con 1GB de memoria asignada. El equipo sobre el que se montó la máquina es un Samsung RV411 con Windows 7, 4GB de memoria RAM y procesador Intel (R) Core (TM) i3 de 2.53GHz.

III-D. Resultados

Como se puede comprobar, los tiempos efectivamente aumentan de forma exponencial, según el

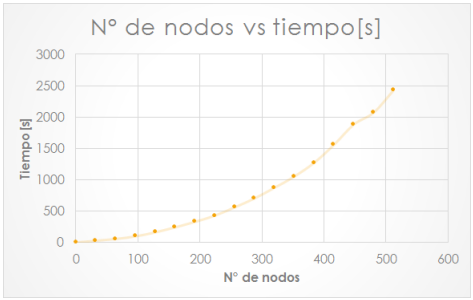

Figura 9. Tiempo de creación/destrucción vs Número de nodos

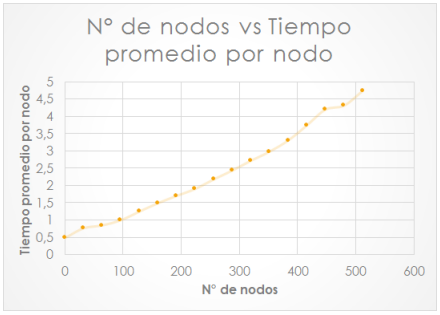

Figura 10. Tiempo de creación promedio por nodo vs Número de nodos

número de nodos de la red. El gráfico de tiempo promedio por nodo ayuda a visualizar el hecho de que el tiempo de creación por cada nodo no es constante, y depende de qué tan grande sea la red. Notar que los resultados difieren con la referencia de forma cuantitativamente, debido a que fueron realizados sobre equipos distintos. Sin embargo, se conserva el análisis cuantitativo.

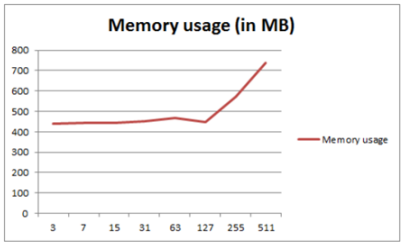

Figura 11. Uso de memoria en la creación/ destrucción de nodos según referencia

Respecto al uso de memoria, se dice que esta se mantiene constante para redes pequeñas, y esta comienza a aumentar linealmente a medida que la red crece. Cabe destacar que éste análisis no es del todo acertado, ya que el script entrega el uso de memoria total, y no sólo el de la aplicación. Es por esto que se deduce que para redes pequeñas el uso de memoria efectivamente aumenta, pero no es notorio en comparación a lo que ocupan las demás aplicaciones y SO.

Este aumento de memoria simplemente se hace significativo en cierto punto, a partir del cuál aumenta de forma lineal.

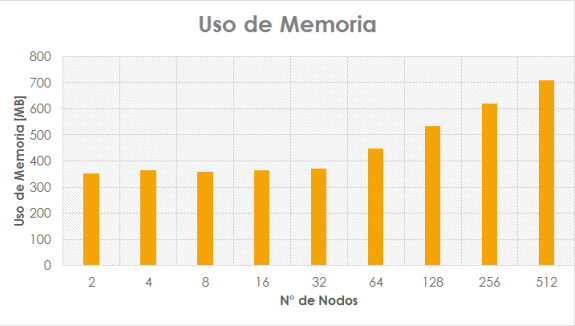

Figura 12. Uso de memoria en la creación/destrucción de nodos

Con el gráfico se comprueba que la memoria efectivamente aumenta de forma lineal conforme aumenta el número de nodos. Es preciso recalcar el hecho de que las diferencias en los valores se deben a los diferentes equipos en los cuales se realizan las pruebas. Esto es importante destacarlo también para enfatizar la dependencia de los recursos de los que dispone Mininet en la emulación. Mientras más recursos, más grande son las redes que puede escalar.

IV. Conclusión

Las Software-Defined Networks presentan una innovación en el área de las redes de computadores, y para su estudio el uso de la plataforma Mininet resulta de gran ayuda. El hecho de que Mininet realize *emulaciónes* en lugar de *simulaciónes* ofrece una gran diferencia respecto a otras herramientas. La *emulación* permite usar los prototipos y códigos usados a redes reales, además de extrapolar los resultados logrados; lo que es una ventaja frente a una red simulada.

Por otro lado, con las pruebas realizadas se demuestra la versatilidad e interactividad de Mininet. Las redes simuladas pueden lograr interacción con redes reales, pudiéndose conectar a Internet. Además, los hosts emulados son capaces de realizar acciones de hosts reales, como montar una servidor web entre otros. Además, los switchs emulados son capaces de usar nuevos protocolos de ruteo, los cuales se pueden diseñar e implementar fácilmente.

Sin embargo, Mininet también presenta desventajas. Con las pruebas de benchmarking realizadas queda claro que la capacidad de Mininet de emular redes a gran escala es limitada por los recursos disponibles en la máquina en donde corre. En éste sentido, los tiempos dependen de la capacidad de procesamiento del computador, y éstos aumentas exponencialmente al aumentar el número de nodos, mientras que el uso de memoria aumenta de forma lineal . Ésto limita claramente la escalabilidad que se puede alcanzar con Mininet. Es importante destacar además de que el hecho de que todos los nodos compartan los mismos recursos de la máquina es también una desventaja. Esto es pues los nodos emulados pueden requerir recursos que la máquina no tiene disponibles para ellos, cosa que no sucedería en equipós reales.

Queda como trabajo futuro el realizar más pruebas sobre las capacidades de emulación de Mininet. Por ejemplo, montar un servidor archivos (Apache) sobre un host, o también el que Mininet simule un switch real que comunique dos equipos reales, aislados uno del otro y que estén visibles solos a través de la interfaz emulada. Ésto último permitirá evaluar qué tanta interactividad se puede lograr con ésta plataforma.

Referencias

[1] de Oliveira, R.L.S. ; Shinoda, A.A. ; Schweitzer, C.M. ; Rodrigues Prete, L. *Using Mininet for emulation and prototyping Software-Defined Networks*, Colombian Conference on Communications and Computing (COLCOM), 2014 IEEE.

[2] Lantz, B. ; Heller, B. ; McKeown N. *A network in a Laptop: Rapid prototyping for Software-Defined Networks*, Hotnets-IX Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, ACM New York, NY, USA ©2010