



**UNIVERSIDAD LATINA
DE COSTA RICA**
LAUREATE INTERNATIONAL UNIVERSITIES®

Profesor:

Carlos Mendez Rodriguez

Curso:

SISTEMAS OPERATIVOS II

3 cuatrimestre

Año:

2024

Alumno:

Fabián Bone Araya

Avance 3 de proyecto

Redes definidas por software (SDN), virtualización, y soluciones de auto recuperación ante fallos (autonomic networking)

Objetivos generales:

- Explicar los conceptos básicos de SDN, virtualización y soluciones de autorrecuperación en el ámbito telemático
- Investigar y demostrar como las redes definidas por software hoy en día son una forma de garantizar la alta disponibilidad del servicio
- Justificar como la virtualización y soluciones de auto recuperación ante fallos (autonomic networking) son elementales para que los proveedores puedan entregar un servicio de alto rendimiento al usuario final

Objetivos específicos:

- Mostrar y argumentar una topología de red en donde se utilicen tecnologías actuales disruptivas
- Desarrollar ventajas y desventajas de la automatización en procesos de Telemáticos utilizando nuevas tecnologías emergentes
- Profundizar sobre el impacto positivo que se obtiene de la virtualización en el campo de las redes
- Estudiar la tendencia de los proveedores de servicio en el uso de nuevas tecnologías ligadas a la telemática
- Presentar un caso de uso real

Plan de trabajo

- Investigar sobre las tecnologías de SDN, virtualización y soluciones de auto recuperación ante fallos.
- Revisión de artículos académicos, libros y recursos online sobre los conceptos básicos y funcionamiento de estas tecnologías.

- Compilar ejemplos y estudio de casos actuales relacionados con la alta disponibilidad, auto recuperación y virtualización siempre con enfoque en redes
- Obtener una comprensión sólida de los fundamentos de SDN, virtualización y autonomic networking en la aplicabilidad.
- Identificar las tendencias actuales en la implementación de estas tecnologías y profundizar en estas técnicas disruptivas
- Investigar las arquitecturas de red basadas en SDN y sus ventajas/desventajas en términos de alta disponibilidad
- Estudiar las soluciones emergentes en virtualización y cómo impactan en la eficiencia y rendimiento de las redes.
- Construir, diseñar o buscar una topología de un caso real de una red que integre SDN, virtualización y soluciones de auto recuperación.
- Argumentar las ventajas y desventajas de cada tecnología o tecnica en la topología a mostrar
- Incluir justificación técnica de cada componente y su aporte al rendimiento, estabilidad, escalabilidad y alta disponibilidad
- Concluir cómo estas soluciones permiten a los proveedores ofrecer un servicio de alto rendimiento.
- Concluir sobre la efectividad de las tecnologías estudiadas en función de la alta disponibilidad y recuperación ante fallos.

Definición de la metodología de investigación

La investigación será descriptiva y exploratoria, enfocándose en identificar, analizar y describir las tecnologías de SDN, virtualización y auto recuperación ante fallos en redes telemáticas. Además, se realizarán estudios de caso de proveedores de servicios para demostrar la aplicación real de estas tecnologías.

El enfoque de la investigación será cualitativo y cuantitativo:

Cualitativo: Para explorar en profundidad los conceptos, principios y teorías relacionadas con las tecnologías emergentes, así como las ventajas y desventajas en la práctica.

Cuantitativo: Para recolectar y analizar datos sobre el impacto de la virtualización y auto recuperación en el rendimiento de redes, basándose en métricas teóricas reales de disponibilidad, tiempo de recuperación y eficiencia.

Identificación de herramientas y recursos necesarios.

- Herramientas técnicas de simulado de redes (opcional)

- Herramientas técnicas de modelado de redes (opcional)
- Herramientas para automatización de auto recuperación de fallos (opcional)
- Herramientas de virtualización (opcional)
- Herramientas de análisis y pruebas (opcional)
- Recursos académicos
- Documentación (imprescindible)
- Casos de uso reales (imprescindible)
- Infraestructura tecnológica a nivel de hardware (Imprescindible)
- Infraestructura tecnológica de software a nivel de maquina (Imprescindible)

Marco teórico

Redes Definidas por Software (SDN)

Las Redes Definidas por Software (SDN, por sus siglas en inglés) son un paradigma y un conjunto de técnicas que separan el plano de control (gestión de la red) del plano de datos (encargado de transportar el tráfico). Este enfoque permite a los administradores gestionar y configurar dinámicamente la red mediante software, lo que facilita la adaptabilidad, la optimización y la alta disponibilidad del servicio.

SDN se basa en una arquitectura que incluye:

Controladores centralizados: Gestionan la red de manera central, lo que permite una visión global de la infraestructura y control total de la topología.

Protocolos de comunicación como OpenFlow, NETCONF, PCEP, etc: Facilitan la interacción entre los controladores y los dispositivos de red.

Dispositivos de red programables: Que permiten modificar su comportamiento sin necesidad de cambiar hardware o visitar el lugar donde se encuentran los componentes físicos

Esta tecnología se ha convertido en un elemento clave para garantizar la alta disponibilidad y rendimiento, ya que permite una rápida adaptación a fallos y cambios en la demanda.

Virtualización

La virtualización es la creación de versiones virtuales de recursos tecnológicos, como servidores, sistemas operativos, redes, dispositivos de almacenamiento, etc. En redes, la virtualización

permite crear entornos aislados y optimizar la utilización de los recursos físicos disponibles en conjunto con recursos virtuales que ofrecen otro tipo de ventajas.

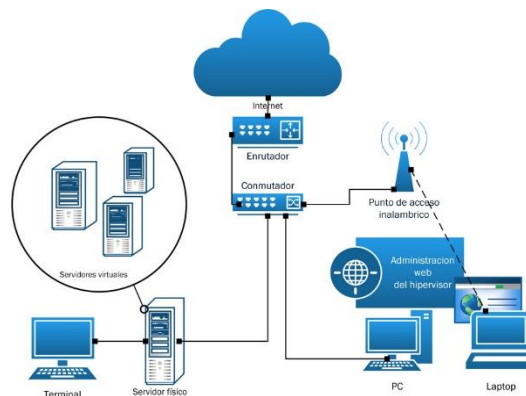
Algunos tipos de virtualización relevantes incluyen:

NFV (Network Function Virtualization): Sustituye dispositivos físicos dedicados (como firewalls o balanceadores de carga) por funciones de red virtualizadas.

Virtualización de servidores: Mejora la eficiencia mediante la consolidación de múltiples máquinas virtuales en un único hardware físico.

SD-WAN (Software-Defined WAN): Una extensión de SDN que virtualiza redes de área amplia para mejorar el rendimiento y reducir costos.

La virtualización aporta flexibilidad y escalabilidad, esenciales para soportar el alto rendimiento y la adaptabilidad requerida en servicios modernos.



Soluciones de Auto Recuperación (Autonomic Networking)

El networking autónomo busca desarrollar redes capaces de autorregularse, autoconfigurarse, y recuperarse de fallos sin intervención humana significativa. Estas redes se inspiran en sistemas biológicos, implementando mecanismos de autoaprendizaje y respuesta adaptativa.

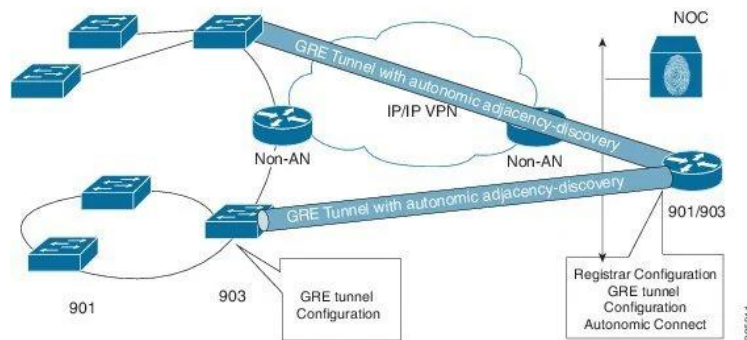
Componentes clave del autonomic networking:

Detección de fallos: Monitoreo constante para identificar problemas de manera proactiva y automatizada.

Automatización de respuesta: Acciones automáticas para mitigar impactos, como la redirección de tráfico en caso de fallos.

Escalabilidad dinámica: Ajuste automático según las necesidades del usuario o la carga de trabajo.

Esto asegura no solo la continuidad del servicio, sino también la mejora continua del rendimiento y la estabilidad en las redes con el fin de brindar la mejor experiencia al usuario final.



Interrelación entre SDN, Virtualización y Autonomic Networking

La combinación de estas tecnologías crea un ecosistema robusto y adaptable que permite a los proveedores de servicios ofrecer soluciones de alta disponibilidad y rendimiento. Por ejemplo:

SDN proporciona un control centralizado y flexible.

La virtualización optimiza los recursos y reduce los costos operativos.

El autonomic networking asegura la recuperación y continuidad del servicio ante fallos.

Estas tecnologías no solo garantizan un servicio más eficiente, sino que también sientan las bases para redes del futuro, donde la adaptabilidad y la resiliencia serán esenciales.

Justificación

La creciente dependencia de los servicios digitales exige que las redes telemáticas sean más robustas, escalables y resilientes. Las tecnologías disruptivas como SDN, la virtualización y las soluciones de auto recuperación permiten a los proveedores afrontar estos desafíos, asegurando un rendimiento constante y adaptándose rápidamente a las necesidades cambiantes de los usuarios.

Conceptos básicos

- Redes SDN:

Las redes definidas por software (SDN) son una categoría de tecnologías que permiten gestionar una red mediante software. La tecnología SDN permite que los administradores de TI configuren sus redes mediante una aplicación de software. El software SDN es interoperable, lo que significa que debería poder funcionar con cualquier enrutador o conmutador, independientemente del proveedor que lo haya fabricado, ya que todo la agrupación de componentes está sincronizada mediante software y controlada por ese medio.

Las redes definidas por software se utilizan cada vez más en los grandes centros de datos. Un centro de datos es un conjunto de servidores y equipos de red, normalmente dentro de un mismo edificio, que almacena, procesa e intercambia datos. Casi todos los servidores web están ubicados dentro de centros de datos, y muchas empresas cuentan con sus propios centros de datos para almacenar datos corporativos y ejecutar aplicaciones internas (por ejemplo, el correo electrónico de la empresa). Como los centros de datos utilizan mucho equipo físico de red, la SDN facilita mucho el trabajo administrativo en los mismos.

SDN también permite que las empresas puedan conectar más fácilmente su infraestructura local con su infraestructura en la nube, como en una implementación de nube híbrida. Las nubes corporativas pueden conectarse con el software mucho más fácilmente que con el hardware; el hardware suele introducir problemas de compatibilidad, mientras que el software en la nube y el software SDN pueden integrarse independientemente del hardware subyacente. De hecho, muchos proveedores ofrecen tanto servicios en la nube como un producto SDN, lo que simplifica todavía más las integraciones en la nube híbrida.

Para comprender mejor cómo funcionan las SDN, sería bueno definir los componentes básicos que crean el ecosistema de la red. Los componentes utilizados para construir una red definida por software pueden o no estar ubicados en la misma área física. Éstas incluyen:

- Aplicaciones : se encargan de transmitir información sobre la red o solicitudes de disponibilidad o asignación de recursos específicos.
- Controladores SDN : controlan la comunicación con las aplicaciones para determinar el destino de los paquetes de datos. Los controladores son los balanceadores de carga dentro de las SDN.
- Los dispositivos de red reciben instrucciones de los controladores sobre cómo enrutar los paquetes.
- Tecnologías de código abierto : protocolos de red programables, como OpenFlow, dirigen el tráfico entre los dispositivos de red en una red SDN. Open Networking Foundation (ONF) ayudó a estandarizar el protocolo OpenFlow y otras tecnologías SDN de código abierto.

Al combinar estos componentes, las organizaciones obtienen una forma más sencilla y centralizada de gestionar las redes. Las SDN eliminan las funciones de enrutamiento y reenvío de paquetes, conocidas como plano de control, del plano de datos o de la infraestructura subyacente. Las SDN luego implementa controladores, considerados el cerebro de la red SDN y los coloca sobre el hardware de red en la nube o en las instalaciones. Esto permite a los equipos utilizar la gestión basada en políticas, un tipo de automatización, para gestionar directamente el control de la red.

Los controladores SDN indican a los switches dónde enviar paquetes. En algunos casos, los conmutadores virtuales que se han integrado en el software o el hardware reemplazarán a los conmutadores físicos. Esto consolida sus funciones en un único conmutador inteligente que puede comprobar los paquetes de datos y los destinos de sus máquinas virtuales para garantizar que no haya problemas antes de mover los paquetes, todo lo que hemos visto anteriormente resume que esta técnica contribuye en demasía con la estabilidad y la continuidad de una red, para así brindar un servicio de calidad a los usuarios finales sin tanta gestión física ya que se achican los usos de dispositivos físicos y también se suma el poder controlar la red de forma remota en su totalidad.

- Virtualización de redes:

La virtualización implica transformar una red que depende del hardware en una que se base en el software. Al igual que con todos los demás tipos de virtualización informática, el objetivo principal es introducir una capa de abstracción entre el hardware físico, las aplicaciones y los servicios que lo utilizan.

Para ser más específicos, la virtualización de la red permite ofrecer las funciones de la red y los recursos del hardware y el software, independientemente del hardware es decir, como una red virtual. Puede usarse para consolidar muchas redes físicas, subdividir una de ellas o conectando dispositivos entre sí.

Con la virtualización de la red, los proveedores de servicios digitales pueden optimizar sus recursos, es decir, menos servidores inactivos y ahorro en diferentes componentes físicos, etc. Permitiéndoles usar un ambiente virtual para las funciones que requerían un hardware propietario costoso y en general, todo esto contribuye a mejorar la velocidad, la flexibilidad y la confiabilidad de sus redes.

- Autonomic Networking:

La automatización de redes es el proceso de usar herramientas de software y scripts para configurar, monitorear y administrar dispositivos y servicios de red. Puede mejorar la eficiencia, confiabilidad y escalabilidad de las operaciones de red, pero también introduce nuevos desafíos y riesgos, en breve se comenta como se manejan los errores de red y los escenarios de recuperación en la automatización de red? Aquí hay algunos consejos y mejores prácticas a seguir.

Cuando se produce un fallo de red, el primer paso es identificar y aislar la causa raíz del problema. Esto se puede hacer mediante el uso de herramientas de diagnóstico, como los comandos ping, traceroute o show, para comprobar la conectividad, el estado y la configuración de los dispositivos y vínculos de red. También puede utilizar sistemas de registro, supervisión y alertas, como Syslog, SNMP o Grafana, para recopilar y analizar datos y eventos de red. Al aislar la causa raíz, puede evitar daños mayores y centrarse en las acciones de recuperación adecuadas.

Uno de los aspectos clave de la automatización de la red es contar con procedimientos de copia de seguridad y restauración para los dispositivos y servicios de red. Esto significa que debe realizar copias de seguridad periódicas de la configuración, el firmware y los datos de los dispositivos y servicios de red, y almacenarlos en una ubicación segura y accesible. También debe probar los procedimientos de copia de seguridad y restauración periódicamente, y documentarlos claramente. De esta manera, puede restaurar rápida y fácilmente la funcionalidad y el estado de la red en caso de falla.

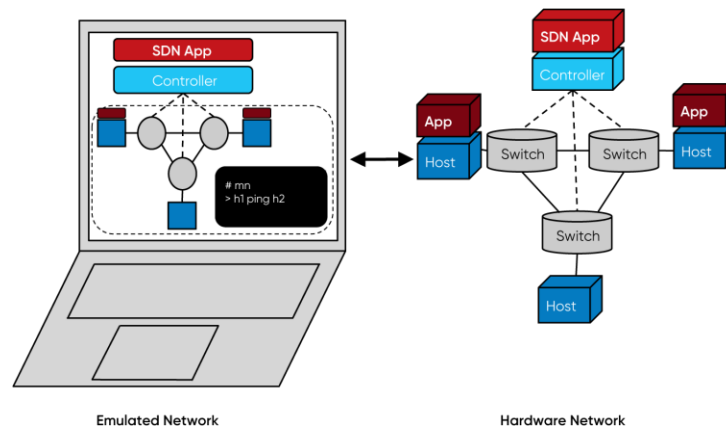
Otro aspecto importante de la automatización de la red es utilizar sistemas de control de versiones y gestión de cambios para la configuración y el código de la red. Los sistemas de control de versiones, como Git o SVN, le permiten rastrear, comparar y revertir los cambios realizados en la configuración y el código de la red. Los sistemas de gestión de cambios, como Ansible o Puppet, le permiten automatizar, validar y auditar los cambios realizados en los dispositivos y servicios de red. Mediante el uso de estos sistemas, puede evitar errores de configuración, conflictos e incoherencias, y asegurarse de que los cambios de red son compatibles y reversibles.

Después de resolver la falla de la red y restaurar la funcionalidad y el estado de la red, no debe detenerse allí. Debe aprender y mejorar del incidente realizando un análisis y revisión post-mortem. Esto implica identificar la causa raíz, el impacto y las lecciones aprendidas del incidente, y documentar los hallazgos y recomendaciones. También debe implementar las acciones correctivas y preventivas de forma automatizada guiándose con todo el estudio previo, actualizando la configuración, el código o los procedimientos, para evitar o mitigar incidentes similares en el futuro.

Implementación

La implementación y la simulación de este proyecto se llevará a cabo con 2 herramientas llamadas Mininet y Miniedit, sumando también un protocolo sumamente importante llamado openflow, con esto lograremos presentar y demostrar una red SDN con diferentes componentes virtualizados y métodos de autorrecuperación.

Hoy en día resulta casi imprescindible realizar una simulación previa de todo proyecto llevado a cabo, ya sea la optimización de una red, la automatización de procesos, la investigación de nuevos entornos y protocolos, etc. Centrándonos en el ámbito que nos atañe, aún resulta más importante realizar estas simulaciones, ya que nos permiten realizar un desarrollo previo, una depuración de errores y la opción de realizar pruebas en un entorno seguro.



En este caso nos centraremos en el entorno de Mininet, un emulador de redes basada en Linux, que da la posibilidad de crear redes virtuales (switches, routers, hosts, etc) bajo el dominio de un controlador basado en SDN. Tras conocer este sistema se realizará un estudio de uno de sus componentes, MiniEdit.

Cabe mencionar la diferencia entre un emulador y un simulador, pese a que ambos softwares son muy parecidos, un emulador es un software con capacidad de ejecutar programas en una plataforma distinta a la que se había programados, mientras que un simulador únicamente se centra en representar o simular el comportamiento del programa.

Además de Mininet existen otras alternativas para la simulación de arquitecturas SDN:

- Ns-3, es un potente simulador empleado en ámbitos educativos e investigación

de código abierto y gratuito, que persigue el desarrollo de simulaciones realistas, pudiendo ser implementado a tiempo real. [16]

- EstiNet, este simulador perteneciente a la compañía con el mismo nombre, enfocada a soluciones de red dedicadas a productos basados en SDN; proporciona un entorno grafico en el que poder llevar a cabo las simulaciones. [17]
- VNX, o Virtual Networks over linuX, es una herramienta de código abierto que permite la creación de escenarios de prueba virtuales, desarrollada en la Universidad Politécnica de Madrid.

Estas son algunas de las alternativas a Mininet que se pueden encontrar en el mercado, pese a ello, el emulador sujeto a estudio, además de ser el más empleado, cuenta con numerosas ventajas, tales como su velocidad de simulación, la capacidad de crear redes personalizadas en las que implementar programas de uso reales, tales como Wireshark en su versión para Linux

programabilidad en la transmisión de paquetes, entre otras. Una ventaja añadida es el hecho de al ser un emulador, permite su ejecución en un amplio abanico de dispositivos (ordenadores, servidores, etc.), además de dar una gran capacidad de difusión, permite compartir y replicar resultados de forma sencilla y la emulación por medio de Python scripts. Una característica más a tener en cuenta es el hecho de ser un proyecto de código abierto en constante desarrollo.

La limitación más restrictiva de Mininet es el hecho de únicamente tener soporte para Linux, siendo necesario emplear versiones con los kernels más modernos, siendo Ubuntu la recomendación de los desarrolladores, garantizando su buen funcionamiento en dicho sistema. Otra limitación a tener presente es el no tener tiempo virtual en sus simulaciones, es decir, las mediciones se basarán en tiempo real, lo que imposibilita emplear redes de alta velocidad.

Instalación

A la hora de instalar el entorno de trabajo se nos presentan dos opciones, el empleo de una máquina virtual Mininet o, por el contrario, una instalación nativa. Ambas opciones están disponibles para sistemas Linux, pero podemos emplear la primera opción para emplear este entorno en otros sistemas operativos, ya sea Windows o Mac Os. A continuación, se describirán los pasos a seguir para ambas instalaciones.

Instalación por medio de una máquina virtual, será necesario contar con una instalación previa de máquinas virtuales, como podría ser el caso de Virtual Box o VMWare. Tras esto se deberá de proceder con la descarga de la imagen de Mininet VM, la cual podremos obtener en la web de Github [21], tras descargar y descomprimir el documento descargado, obtenemos dos archivos, el disco de

datos virtual y un segundo archivo que contiene los parámetros de configuración. Tras los pasos anteriores se debe abrir el archivo correspondiente desde nuestra máquina virtual, y con ello ya podremos empezar a trabajar con Mininet. Cabe la posibilidad de poder acceder a este sistema vía SSH.

Para ello debemos de comprobar que el adaptador de red este configurado, se recomienda crear un segundo adaptador configurado como “host-only”, el procedimiento a seguir puede variar en función del programa para máquinas virtuales que se esté empleando.

Para seleccionar el adaptador, desde la máquina virtual comprobaremos si está activo este nuevo elemento, mediante el comando **ifconfig -a**, en la siguiente figura se aprecia un ejemplo del adaptador sin configurar:

```
mininet@mininet-vm:~$ ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:0c:29:f1:c2:db
          inet addr:192.168.234.145  Bcast:192.168.234.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:269 errors:0 dropped:0 overruns:0 frame:0
          TX packets:279 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:25177 (25.1 KB)  TX bytes:24644 (24.6 KB)

eth1      Link encap:Ethernet  HWaddr 00:0c:29:f1:c2:e5
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Comprobada la configuración, se debe ejecutar el comando **sudo dhclient eth1**, con ello el adaptador quedará configurado y tras comprobar su dirección IP, se podrá realizar el enlace por medio de un enlace SSH.

```
mininet@mininet-vm:~$ sudo dhclient eth1
mininet@mininet-vm:~$ ifconfig -a
eth1      Link encap:Ethernet  HWaddr 00:0c:29:f1:c2:e5
          inet addr:192.168.67.129  Bcast:192.168.67.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:746 (746.0 B)  TX bytes:684 (684.0 B)
```

Tras esto desde el terminal del ordenador se podrá acceder a la máquina virtual Mininet empleando el comando **ssh -Y mininet@192.168.67.129**, siendo la IP la configurada en el adaptador. En caso querer acceder desde un terminal OS se debe remplazar la **Y** por una **X**; en ambos casos, tras identificarnos, se podrá hacer completo uso de las prestaciones de Mininet..

En cualquier caso, es sistema pedirá una acreditación por parte del usuario, y con ello poder hacer uso de sus funciones, para ello deberemos de emplear las credenciales de **mininet** tanto para el campo de usuario, como para el campo de contraseña.

Otra opción es Instalación nativa, este método solo puede ser llevado a cabo en sistemas basados en Linux, preferentemente en el sistema Ubuntu.

Para poder realizar esta instalación, se debe tener acceso a internet desde el equipo donde se va a proceder a la instalación, hecha las comprobaciones previas emplearemos los siguientes comandos:

- **git clone git://github.com/mininet/mininet**, por medio del cual se importa el contenido de dicho repositorio.
- **cd mininet y git checkout -b**, cumple la funcionalidad de comprobar las versiones disponibles para instalar.

Como se puede apreciar en la imagen se dispone de diversas opciones de instalación, siendo recomendable para un uso genérico emplear la primera opción, la cual realiza una instalación completa en el directorio principal. Para ello se debe emplear la instrucción **mininet/útil/install.sh**

-a o en caso de no ser necesario especificar la ruta, emplear únicamente **install.sh -a**.

```
git clone git://github.com/mininet/mininet
cd mininet
git tag # list available versions
git checkout -b 2.2.1 2.2.1 # or whatever version you wish to install
cd ..
install.sh -h
To install everything (using your home directory): install.sh -a
To install everything (using another directory for build): install.sh -s mydir -a
To install Mininet + user switch + OVS (using your home dir): install.sh -nfv
To install Mininet + user switch + OVS (using another dir:) install.sh -s mydir -nfv
mininet/util/install.sh -a
```

Tras la instalación es recomendable emplear el comando **sudo mn -test pingall** con el único propósito de comprobar las funcionalidades básicas de Mininet, y con ello la correcta instalación.

Es probable que a la hora de seguir estos pasos nos encontremos con que el sistema nos devuelva "*command not found*" al ejecutar el primer comando **git clone git...**, en caso de ser así la solución llega tras la ejecución de las siguientes tres líneas de comandos:

```
sudo apt-get
update sudo apt-
get upgrade sudo
apt-get install git
```

Con ello comprobaremos las actualizaciones del sistema e instalaremos la versión correspondiente de Git necesarias para poder obtener el software requerido para llevar a cabo el estudio de este entorno de trabajo.

Entorno de trabajo ya preparado:

Una vez implementado el sistema resulta necesario hacer un repaso de los comandos básicos para poder implementar una estructura de red. Por medio del comando “help” podremos acceder a un pequeño resumen de las instrucciones y su sintaxis:

```
mininet> help

Documented commands (type help <topic>):
=====
EOF      gterm  iperfudp  nodes      pingpair    py      switch
dpctl    help   link      noecho     pingpairfull  quit    time
dump     intfs  links     pingall    ports       sh      x
exit     iperf  net       pingallfull px          source  xterm

You may also send a command to a node using:
  <node> command {args}
For example:
  mininet> h1 ifconfig

The interpreter automatically substitutes IP addresses
for node names when a node is the first arg, so commands
like
  mininet> h2 ping h3
should work.

Some character-oriented interactive commands require
noecho:
  mininet> noecho h2 vi foo.py
However, starting up an xterm/gterm is generally better:
  mininet> xterm h2
```

En la imagen podemos apreciar que la sintaxis básica para programar los diferentes nodos que pueden conformar una red “(nombre del nodo) comando”, de esta forma el sistema reconoce el nodo en concreto sin necesidad de espaciarlo por su IP, cabe destacar que para ejecutar comandos interactivos orientados a las características de los nodos necesitan el empleo de “noecho”, tal y como se muestra en la figura anterior.

Al iniciar la plataforma de Mininet se crean diversos nodos con los que poder practicar los comandos anteriores. Estos nodos estarían formados por dos hosts, un switch OpenFlow y un controlador.

En caso de querer añadir un nuevo host con el que poder ampliar las simulaciones de la red, se debe emplear el comando **py net.addHost(<nombre del host>)** para incluirlo en el escenario de trabajo.

Una vez incluido se deben de configurar las conexiones que le permitirán comunicarse con el resto de la red. Para ello se empleara la instrucción **py net.addLink(s1, net.get(<nombre del host>))**, tras aplicar los cambios en el switch, empleando **py (nombre del switch).attach((nombre del puerto de enlace))**; y configurar una ip valida al nuevo host, quedará listo para la comunicación con el resto de la red.

Para configurar el nuevo terminal se debe emplear ***py net.get('nombre del terminal').cmd('ifconfig (puerto del terminal) (ip)')***, de esta forma quedara asociada la IP deseada al puerto que se está empleando.

En la figura se puede observar un ejemplo de la creación del host 3 y de su correcta comunicación, con los otros hosts, por medio del comando ping:

```
mininet> py net.addHost('h3')
<Host h3: pid=3954>
mininet> py net.addLink(s1, net.get('h3'))
<mininet.link.Link object at 0x7f85f29668d0>
mininet> py s1.attach('s1-eth3')
mininet> py net.get('h3').cmd('ifconfig h3-eth0 10.0.0.10')
mininet> h1 ping h3
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=1 ttl=64 time=13.0 ms
--- 10.0.0.10 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9188ms
rtt min/avg/max/mdev = 0.072/1.418/13.079/3.888 ms
mininet> h2 ping h3
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=1 ttl=64 time=9.40 ms
--- 10.0.0.10 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4077ms
```

Si se desean añadir otros elementos a la red como por ejemplo un switch se debe emplear el comando ***net.addSwitch('nombre del nuevo nodo')***.

Pese a ello la forma más habitual de trabajar con esta tecnología es haciendo uso del amplio abanico de herramientas que ofrece, como la creación de una red por medio del comando “***topo***”, que nos permite crear redes sencillas en función de su topología ya sea single, linear, en árbol o personalizada.

Un ejemplo de ello sería la estructura resultante de emplear ***\$ sudo mn -topo linear, (número de switches y hosts)***. De la ejecución de esta instrucción se obtendría una red con una topología simple formada por el número de nodos indicados.

El inconveniente al método anterior es que las redes resultantes carecen de controlador

de red.

Por medio del comando **controller**, se implementarían topologías con un controlador, pudiendo especificar el tipo de controlador, la topología, el número de componentes y el tipo de switch.

Un ejemplo de ello sería la ejecución del argumento:

```
$ sudo mn --controller=remote --topo single, 3 --mac --switch ovsk, protocols=OpenFlow10 --nat
```

Obteniendo una arquitectura con un controlador remoto, con una topología formada por un switch conectado a tres hosts en una estructura single. Los terminales tendrían su propia dirección MAC, mientras que, en el caso del controlador, al no especificarse la IP recibirá una por defecto. Para especificar una IP para el controlador, bastaría con añadir a la instrucción el argumento: **ip=(dirección ip)**. En el caso del switch, se ha especificado el tipo que se va a implementar, ovsk, y su protocolo, OpenFlow en su versión 1.0.

Una de las herramientas que ofrece mininet es Miniedit, la cual permite la creación y simulación de redes de una forma muy simplificada y aun alto nivel mediante una interfaz gráfica.

Miniedit:

Dentro del propio editor podemos encontrar la carpeta de *examples*, donde podemos encontrar diversos ejemplos y herramientas, en la que destaca la herramienta de Miniedit, una extensión de Mininet, que nos permite crear redes de forma sencilla sobre un terminal gráfico.

Esta interfaz facilita la creación de redes, realizándose la programación de las mismas en un según plano a priori oculto para el usuario, pese a ello esta plataforma presenta ciertas limitaciones en comparación con todas las capacidades que presenta el propio Mininet.

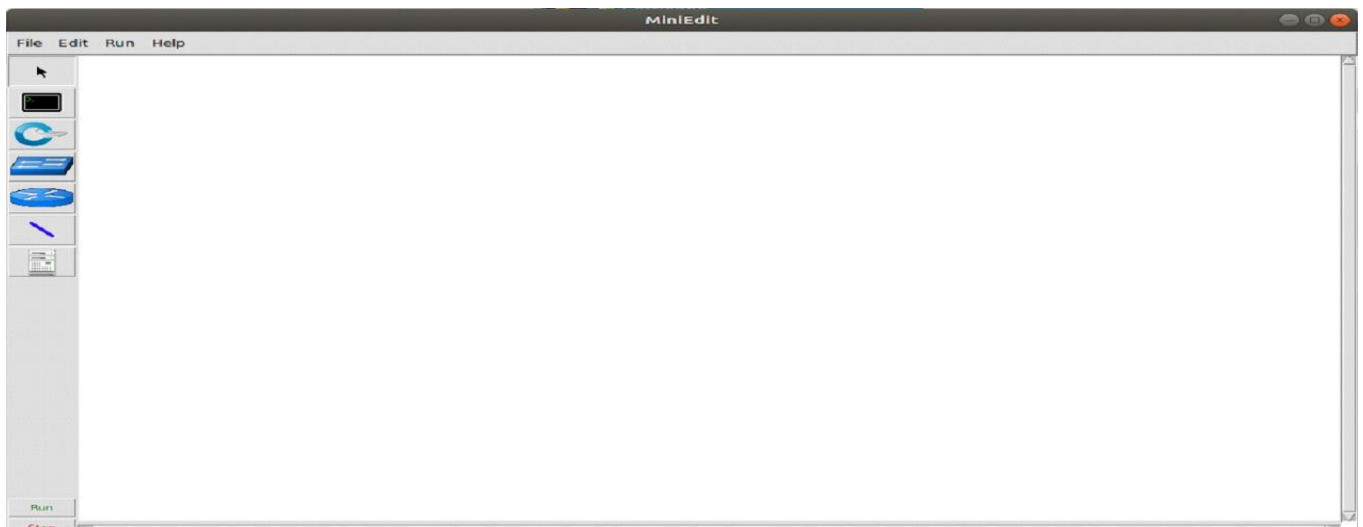
Para poder ejecutar esta función deberemos cerrar las simulaciones que se estén ejecutando en el entorno de Mininet, siendo necesario emplear el comando **exit**, o similar, para cerrar la simulación creada en la interfaz principal, mediante el comando **killall controller**, se finaliza los procesos del controlador de la red, y por último se debe ejecutar **sudo mn -c**, con el cual se limpian todos los elementos que hayan podido ser creados en la simulación anterior.

Una vez cerradas todas las simulaciones o con un terminal nuevo, se pueden emplear estas dos instrucciones para arrancar el entorno de trabajo:

sudo ~/mininet/examples/miniedit.py

sudo python ./mininet/examples/miniedit.py

Una vez arrancado Miniedit presenta una interfaz de usuario simplificada, con una zona de trabajo y una fila de herramientas representadas por iconos, en el lado izquierdo, además de una barra de menú en la parte superior.



En el lateral del lienzo de trabajo, podemos observar las herramientas de trabajo, concretamente son las siguientes:



FLECHA DE SELECCIÓN: Se emplea para mover los nodos por el escenario de trabajo, esta función no es estrictamente necesaria para seleccionar los nodos, puesto que podemos seleccionarlos con el resto de herramientas. Empleando el botón derecho para poder emplear el menú de configuración,



HOST: Permite añadir elementos host o terminales al lienzo, mientras este seleccionada nos permite añadir múltiples elementos, los cuales pueden ser configurados mediante las propiedades en el menú desplegado con el botón derecho.



CONMUTADOR: Crea un Open vSwitch, que implementa una plataforma de cálda, capaz de hacer una gestión estándar y habilitar de forma programable las funciones de forwarding o transmisión de la información, pudiendo ser configurados de la misma forma que el elemento anterior.



CONMUTADOR TRADICIONAL: Crea un switch Ethernet con una configuración predeterminada, funcionando de forma independiente al controlador. Este equipo no puede ser configurado, parten con el protocolo Spanning Tree deshabilitado, por lo que no se deben conectar en bucle.



ROUTER TRADICIONAL: o Router Legacy, crea enrutadores básicos, que funcionan de forma independiente al controlador. En definitiva, es un host con la IP Forwarding habilitada. Este nodo no puede ser configurado desde MiniEdit.



ENLACES: Crea los diferentes enlaces entre los elementos de la red. Las propiedades de cada enlace se pueden configurar, mediante el botón derecho del ratón.

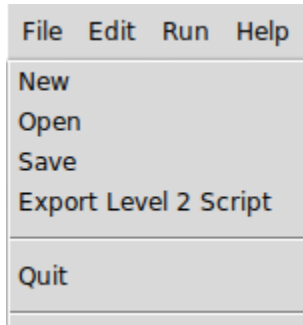


CONTROLADOR: Permite implementar diversos controladores de red, por defecto se crea un controlador de openFlow. Se pueden configurar otro tipo de controladores modificando las propiedades o configuración de los mismos, desde el menú desplegable con el botón derecho sobre él.



EJECUTAR / PARAR: Por medio de este elemento se tiene control sobre la simulación de la red conformada. Mientras se está ejecutando una simulación, por medio del botón derecho del ratón sobre un elemento, se muestran funciones operativas como abrir un terminal, ver su configuración o establecer el estado de un enlace (activándolo o desactivándolo) exclusivas del mudo de emulación.

Conociendo las funciones de los botones de la interfaz de trabajo, resulta conveniente familiarizarse con el empleo de los menús de la parte superior. Comenzado por el lado izquierdo, el primer menú es el menú de archivo o “File”

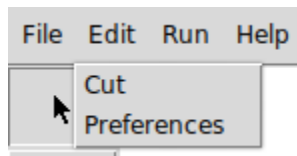


Desde este menú se puede acceder a un nuevo lienzo de trabajo, abrir uno ya creado o, guardar o exportar el actual. Estas dos últimas funciones resultan de gran utilidad a la hora de simular en mininet entornos de trabajo creados con la herramienta grafica.

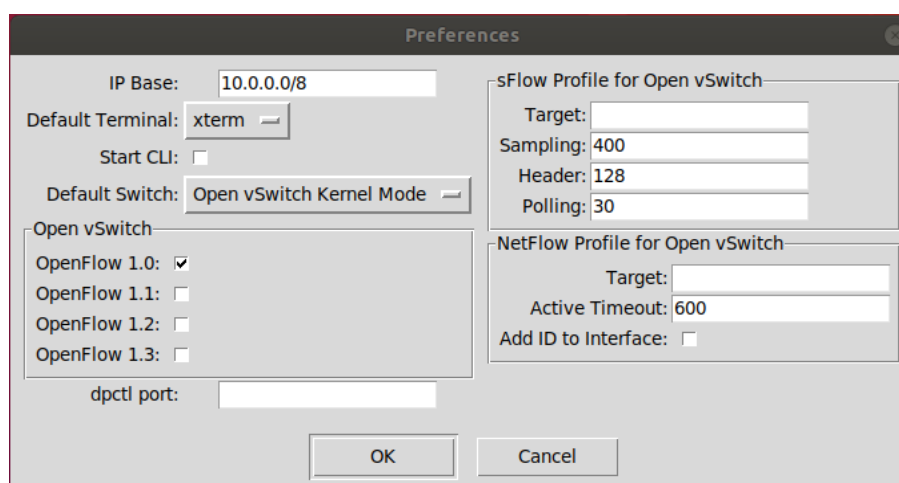
Con la opción de exportar se genera un archivo con extensión “.py”, la cual puede ser abierta desde el programa de edición Python.

Por medio de la última opción se cerraría el escenario de trabajo actual.

El segundo menú que se presenta es el menú de edición o “Edit”, un sencillo menú que cuenta con la herramienta “Cut”, su funcionalidad es semejante a una herramienta de borrado, su método de empleo se resume en seleccionar el elemento que se quiera cortar o borrar y seleccionar la función en dicho menú.



La otra opción que figura en este menú es la ventana de *Preferences* o preferencias, donde se pueden realizar las configuraciones generales de la zona de trabajo.



Desde la interfaz que se abre tras seleccionar *Preferences* se pueden editar valores globales del área de trabajo, tales como la dirección IP por defecto, incluyendo su máscara de red, el terminal de trabajo de los hosts, el modelo y la versión de los switch creados con tecnología OpenFlow; además del puerto de dpctl a emplear.

En la parte derecha de la pantalla se habilita la edición de las propiedades de los protocolos sFlow y NetFlow, los cuales se pueden habilitar en las propiedades de cada switch.

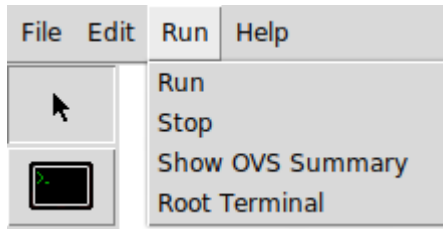
NetFlow [23], es un protocolo de red propiedad de Cisco Systems, empleado para capturar información acerca del tráfico IP.

sFlow [24], es un estándar de muestreo que permite el análisis de los flujos de datos, en la capa de aplicación, de forma simultánea en todas las interfaces.

El ultimo campo editable desde esta ventana es el arranque de la línea de comandos o CLI. Dicha funcionalidad resulta muy útil a la hora de comprobar la configuración de las redes creadas en este entorno de trabajo, puesto que nos permite hacer uso de las funcionalidades de la plataforma Mininet, sin la necesidad de guardar y abrir la estructura creada desde otra ventana de comandos.

Una vez activa esta función se puede observar como en la línea de comandos o CLI desde la que se ha arrancado la herramienta Miniedit pasa a tener el prompt “mininet>”.

Esta habitación nos permite seguir los pasos de la creación de la red una vez se iniciada la simulación, así como de todas las modificaciones referentes a la misma, algunos ejemplos serían la modificación de las propiedades de un enlace, el tipo de switch o de un controlador.

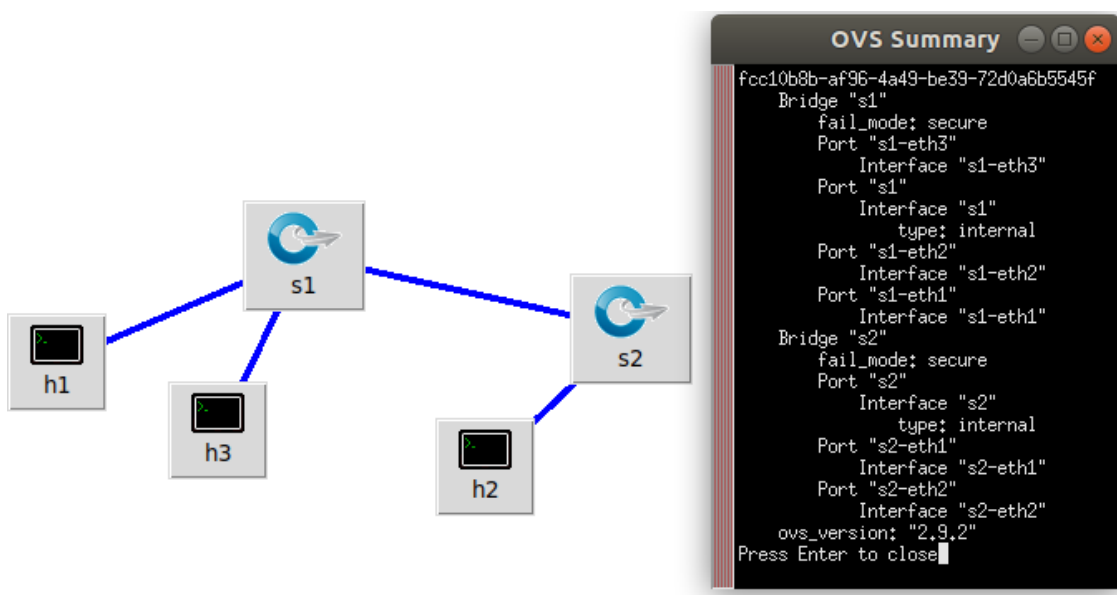


El siguiente menú presente en Miniedit es el menú de "Run" o de ejecución, desde el cual podremos iniciar o detener la simulación de la estructura abierta en el lienzo de trabajo.

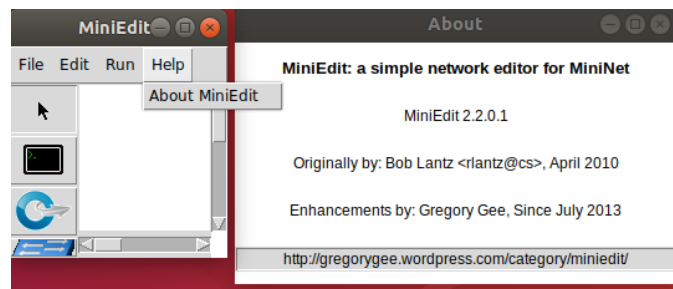
Además de dotar de la opción de abrir un nuevo terminal Root, desde el que poder ejecutar el entorno de Mininet.

La opción más importante de este menú es "Show OVS Summary", desde la cual se tiene acceso a la ventana de comandos llamada "OVS Summary", desde la cual se puede observar la configuración de todos los puertos de los nodos de comunicación de la red implementada.

En la siguiente figura se puede apreciar un ejemplo de la funcionalidad de este terminal, en el cual se puede apreciar la configuración de todos los puertos en uso en ambos switches.



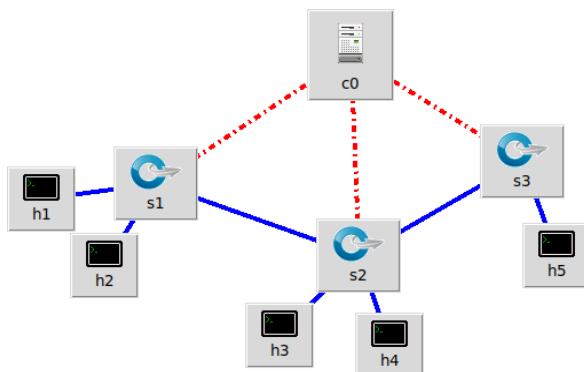
El ultimo menú de la barra de herramientas es el destinado a dar soporte, reconocible por el nombre de “Help”, su única funcionalidad es dar información al usuario acerca del editor que se está empleando.



Tras haber adquirido una visión ampliada y conocer las funcionalidades básicas del entorno de trabajo, resulta conveniente crear y emular una sencilla arquitectura de red con la que poder comprobar el funcionamiento de las funcionalidades ya descritas, así como las que hacen referencia a los distintos componentes de una red.

Con esta idea presente, se propone la creación de una arquitectura sencilla compuesta por tres switches, con tecnología OpenFlow e interconectados; por comunicación directa con un controlador de red. Además de diversos usuarios o host, conectados a cada uno de los vSwitches.

Siguiendo las características anteriormente mencionadas se realiza la siguiente red:



En primer lugar se comprobaran las configuraciones por defecto de cada uno de los componentes de la red, así como de las opciones que presentan. En primer lugar se accederá a las propiedades del controlador de red:

Entre los campos editables que presenta el controlador se encuentran el nombre o etiqueta que se emplea en mininet para hacer referencia al mismo, el puerto que emplea, el protocolo a emplear (TCP o SSL), su IP remota y el tipo de controlador, pudiendo elegir entre:

- OpenFlow Reference.
- In-Band Controller.
- Remote Controller.
- OVS Controller

En el caso de los Switches las opciones son mayores, además de permitir la edición de los campos básicos, como serían el nombre del nodo, su DPID, su dirección IP y su puerto DPCTL, permite la habilitación de los protocolos NetFlow y SFlow, configurables desde la ventana de preferencia en el menú de edición.

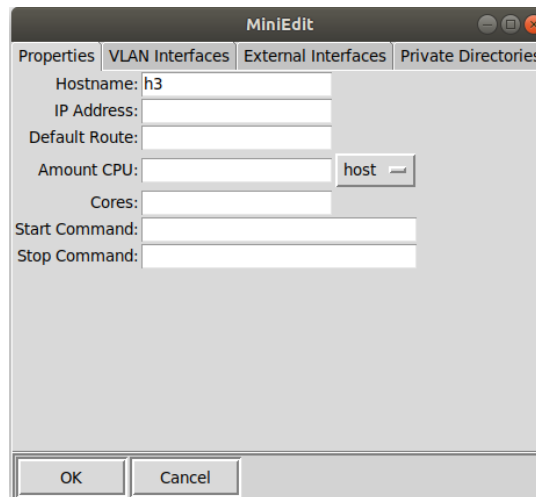
También se dispone de la posibilidad de establecer un comando de arranque y de parada, así como de establecer una interfaz externa.

Pese a disponer de varios tipos de switch, para esta simulación se empleará la configuración por defecto, la cual está establecida en la ventana de preferencias del entorno, Figura 25. Dicha configuración corresponde con un “Open vSwitch”

Las propiedades configurables básicas de cada host son semejantes a las vistas en los switches, nombre del nodo, dirección IPE y la capacidad de establecer comandos de inicio y paro. Estos terminales también cuentan con la posibilidad de configurar una ruta por defecto, indicar la cantidad de cores y de CPU disponible. Además de poder configurar un directorio privado, una interface VLAN y otra externa.

Al igual que para los casos anteriores dejaremos sus valores por defecto. Al no indicar una IP específica para cada host estarán configurados con la dirección por defecto acabada por su número de host, es decir, en el caso del terminal de la Figura 32, el “h3”, su IP

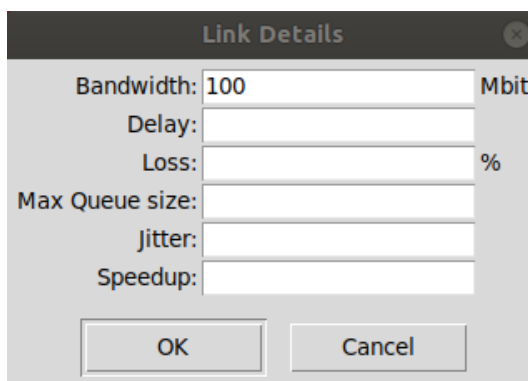
por predeterminada sería 10.0.0.3 con una máscara de 8 bits (255.0.0.0).



The image shows a window titled "MiniEdit" with four tabs: "Properties", "VLAN Interfaces", "External Interfaces", and "Private Directories". The "Properties" tab is active. It contains several input fields: "Hostname" with the value "h3", "IP Address", "Default Route", "Amount CPU" with a dropdown menu showing "host", "Cores", "Start Command", and "Stop Command". At the bottom are "OK" and "Cancel" buttons.

Por último, en la configuración de los enlaces es posible establecer condiciones o calidad de enlace de una forma más detallada, permitiendo realizar simulaciones lo más realistas posibles, pudiendo configurar el speedup, el retraso o delay, el porcentaje de pérdidas, el tamaño máximo de la cola y la fluctuación del retardo o jitter.

Para la simulación emplearemos un ancho de banda de 100Mbits.



The image shows a window titled "Link Details" with a close button in the top right corner. It contains several input fields: "Bandwidth" with the value "100" and a unit dropdown set to "Mbit", "Delay", "Loss" with a unit dropdown set to "%", "Max Queue size", "Jitter", and "Speedup". At the bottom are "OK" and "Cancel" buttons.

Tras repasar la configuración de todos los elementos, se debe comprobar la correcta configuración de los puertos empleados en los diferentes equipos, para ello hacer uso del terminal OVS Summary, donde se puede apreciar la correcta configuración, así como el controlador al que se encuentra asociado cada switch, en este caso todos hacen referencia al mismo, C1, con IP 127.0.0.1.


```

OVS Summary
fcc10b8b-af96-4a49-be39-72d0a6b5545f
  Bridge "s3"
    Controller "tcp:127.0.0.1:6633"
    fail_mode: secure
    Port "s3"
      Interface "s3"
        type: internal
    Port "s3-eth2"
      Interface "s3-eth2"
    Port "s3-eth1"
      Interface "s3-eth1"
  Bridge "s1"
    Controller "tcp:127.0.0.1:6633"
    fail_mode: secure
    Port "s1-eth1"
      Interface "s1-eth1"
    Port "s1-eth2"
      Interface "s1-eth2"
    Port "s1"
      Interface "s1"
        type: internal
    Port "s1-eth3"
      Interface "s1-eth3"
  Bridge "s2"
    Controller "tcp:127.0.0.1:6633"
    fail_mode: secure
    Port "s2"
      Interface "s2"
        type: internal
    Port "s2-eth1"
      Interface "s2-eth1"
    Port "s2-eth3"
      Interface "s2-eth3"
    Port "s2-eth2"
      Interface "s2-eth2"
    Port "s2-eth4"
      Interface "s2-eth4"
  ovs_version: "2.9.2"
Press Enter to close

```

Antes de comenzar una transmisión, y con las configuraciones previas realizadas, se puede observar que no hay contenido en las tablas de flujos de los diferentes enrutadores, para poder comprobar esta afirmación se debe emplear el comando ***dpctl dump-flows*** en el terminal con las funciones de mininet activas. Para ello se ha habilitado la función CLI en la configuración de las preferencias del entorno de trabajo.

```

mininet> dpctl dump-flows
*** s3 ***
*** s2 ***
*** s1 ***
mininet> h1 ping -c3 h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data:
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=14.1 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=0.841 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.077 ms

--- 10.0.0.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2020ms
rtt min/avg/max/mdev = 0.077/5.011/14.115/6.445 ms
mininet> dpctl dump-flows
*** s3 ***
*** s2 ***
*** s1 ***
mininet>

```

Una vez realizada la comprobación, se inicia una transmisión entre el host 1 y el host 4, en este caso se emplea un ping con el fin de comprobar la comunicación y la actualización de las FlowTables.

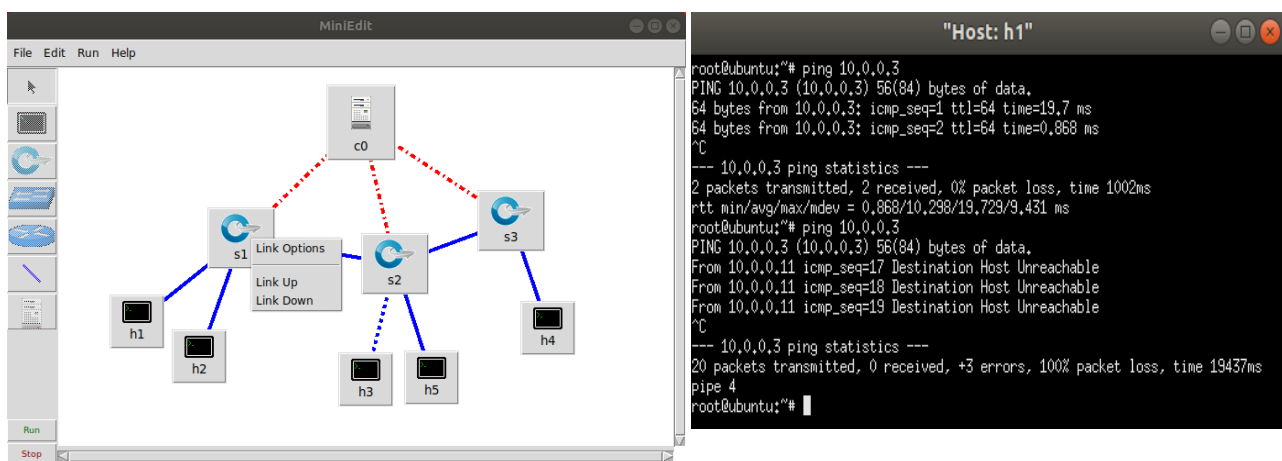
Realizada la transmisión, como se puede observar en la figura anterior, las tablas de los switch implicados en la transmisión se han actualizado, gracias a lo cual se habilita la correcta comunicación entre dichos terminales.

Comprobada la comunicación, se van a comprobar los efectos de las alteraciones de los enlaces entre los diferentes nodos. En este caso se simulará la caída de un enlace, el aumento de la tasa de retardos y el incremento del número de pérdidas en el medio.

Antes de realizar ninguna prueba con los enlaces se deben de borrar las entradas de las tablas por medio de la función ***dpctl del-flows***.

En primer lugar, se procede a deshabilitar un enlace desde la interfaz gráfica, tras lo cual se intentará realizar una comunicación entre el host 1 y el host 3. Cabe esperar que la comunicación entre ambos nodos falle y, al no tener conexión, sus respectivas entradas a las tablas de flujo no se generen.

Cabe destacar que al inutilizar un enlace desde la interfaz gráfica, este se muestra como una línea azul discontinua, a diferencia de los enlaces activos que se muestran con una línea continua.



```
mininet> dpctl dump-flows
*** s3 ***
*** s2 ***
cookie=0x0, duration=21.091s, table=0, n_packets=2, n_bytes=196, idle_timeout=60, priority=65535,icmp,in_port="s2-eth4"
w_dst=10.0.0.4,nw_tos=0,icmp_type=8,icmp_code=0 actions=output:"s2-eth2"
*** s1 ***
cookie=0x0, duration=21.095s, table=0, n_packets=2, n_bytes=196, idle_timeout=60, priority=65535,icmp,in_port="s1-eth1"
w_dst=10.0.0.4,nw_tos=0,icmp_type=8,icmp_code=0 actions=output:"s1-eth3"
mininet>
```

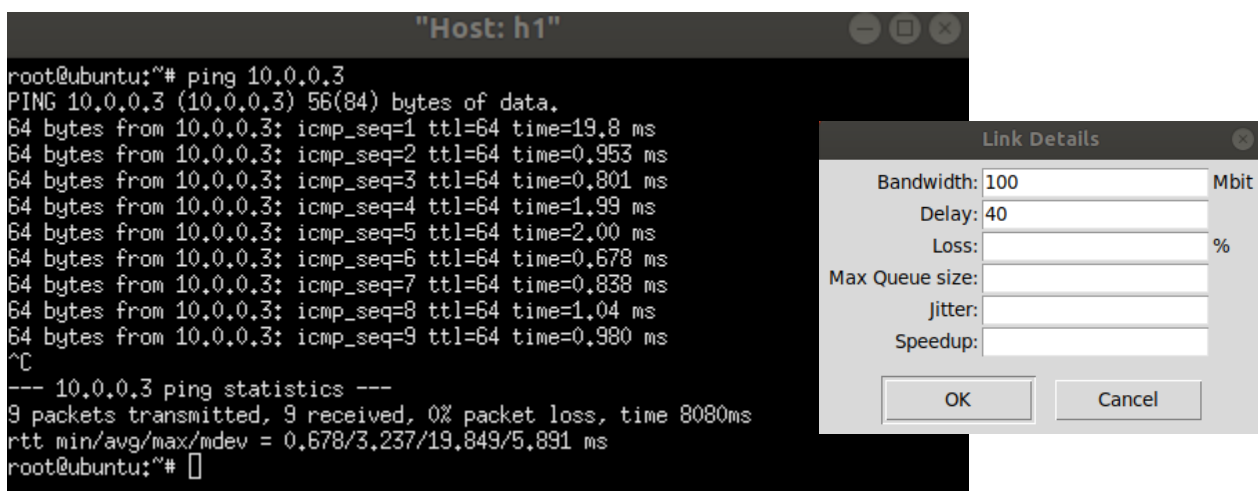
(Fallo de comunicación)

Tras realizar la simulación se puede afirmar que, al perder la comunicación por un enlace, en caso de no disponer de una alternativa de comunicación, la tabla de flujos no se actualiza, generando únicamente la ruta de comunicación con el controlador de la red.

Teniendo en cuenta que la habilitación y des habilitación de un enlace desde la interfaz gráfica única mente se puede realizar con la simulación en marcha, se vuelve a normalizar el estado del enlace caído.

Con la simulación detenida se editan las características del enlace de comunicación entre los dos switches, en un primer caso práctico se aumenta el delay o retardo en la comunicación.

Una vez hecha la modificación y la simulación en marcha se el terminal del host 1, el cual emplearemos para hacer un seguimiento de la comunicación con el host 3.



```
Host: h1
root@ubuntu:~# ping 10.0.0.3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=19.8 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.953 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.801 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=1.99 ms
64 bytes from 10.0.0.3: icmp_seq=5 ttl=64 time=2.00 ms
64 bytes from 10.0.0.3: icmp_seq=6 ttl=64 time=0.678 ms
64 bytes from 10.0.0.3: icmp_seq=7 ttl=64 time=0.838 ms
64 bytes from 10.0.0.3: icmp_seq=8 ttl=64 time=1.04 ms
64 bytes from 10.0.0.3: icmp_seq=9 ttl=64 time=0.980 ms
^C
--- 10.0.0.3 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8080ms
rtt min/avg/max/mdev = 0.678/3.237/19.849/5.891 ms
root@ubuntu:~#
```

Link Details

Bandwidth:	100	Mbit
Delay:	40	
Loss:		%
Max Queue size:		
Jitter:		
Speedup:		

OK Cancel

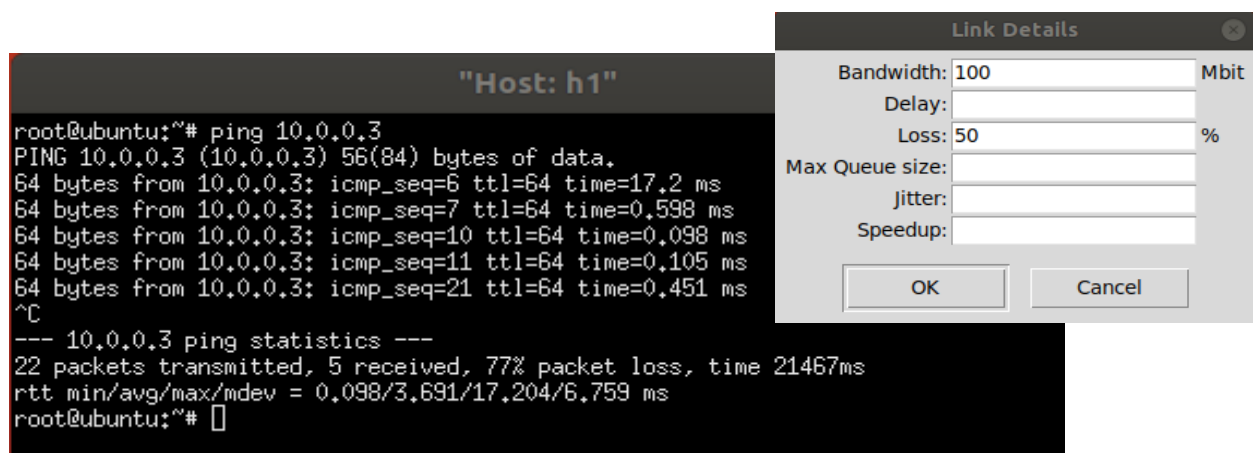
(Comunicación tasa de Delay)

Tras la visualización de nueve paquetes ICMP resultantes de realizar el ping entre ambos terminales resulta evidente la fluctuación del retardo en cada una de las transmisiones causada por el retraso establecido.

Por último, se realiza el caso práctico con el que se pretende observar el comportamiento de la red al aumentar la tasa de pérdidas en un enlace.

Para ello, y al igual que el caso anterior, se debe detener la simulación para poder realizar la configuración del enlace.

(Comunicación con pérdidas)



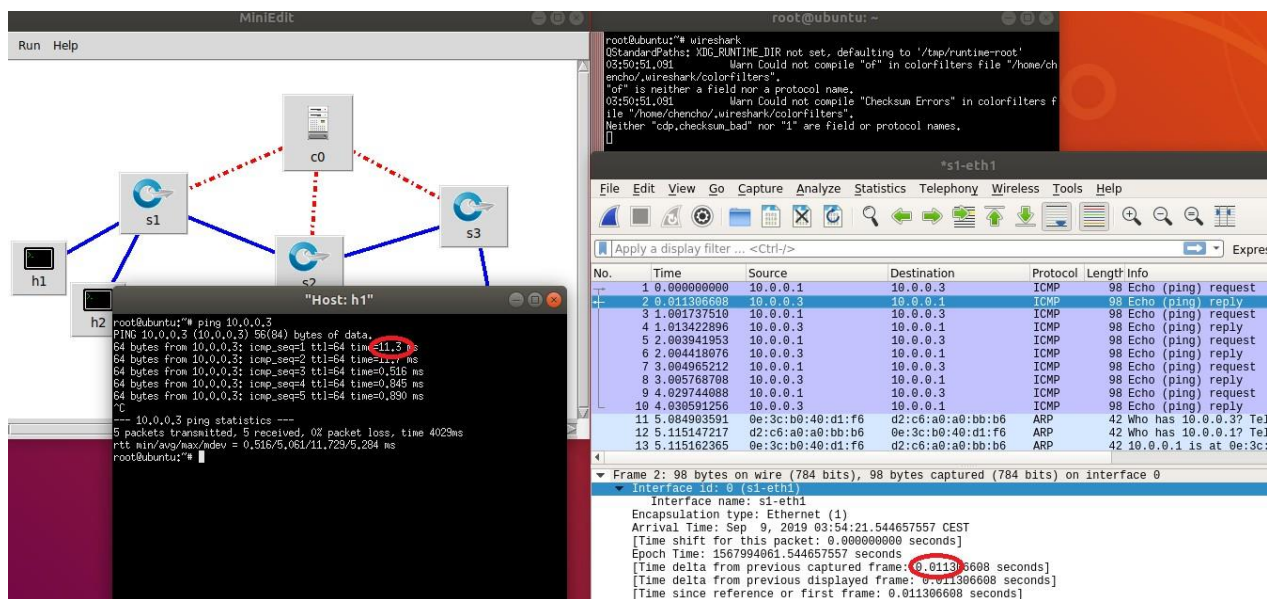
Como se puede observar en la imagen anterior de los 21 paquetes que se han transmitido entre los terminales, únicamente han alcanzado su destino cinco de ellos. Por lo que quedaría demostrado el correcto funcionamiento de las funciones de configuración de los distintos enlaces.

Para la realización de estas simulaciones, se ha decidido hacer el seguimiento desde el propio terminal, siendo posible emplear la herramienta de Wireshark, que es un analizador de tráfico, desde el cual se puede realizar el mismo procedimiento.

En caso de querer hacer el seguimiento de las mismas simulaciones desde la aplicación de Wireshark, se debe de abrir desde la línea de comandos *Root* de Miniedit, a la cual se puede acceder desde el menú de *Run*

Una vez abierta la aplicación, y tras es el escaneo de los nodos disponibles para su monitorización, se debe seleccionar el nodo o el puerto que se quiere escuchar o analizar, en este caso se hará el seguimiento al puerto ethernet 1 del primer switch, *s1-eth1*.

A modo de ejemplo se repetirá la segunda simulación, con un delay de 40 unidades. Como se puede apreciar en la siguiente figura el resultado obtenido es el mismo a los valores resumidos en la CLI del host 1.



(Muestra herramienta que pone en evidencia comunicación con perdidas)

Una vez completadas las emulaciones, para poder finalizar la misma de forma correcta, se debe finalizar primero el CLI de Mininet, por medio de la instrucción **exit**, y tras esto se puede proceder a detener la ejecución del entorno de Miniedit, por medio del botón *Run / Stop*.

Este procedimiento se especifica al habilitar la CLI, y resulta aconsejable seguirlo para evitar fallos en la emulación que puedan imposibilitar su correcta ejecución.

Una vez detenida la ejecución de la red, se puede realizar el guardado de la misma para una próxima ejecución. Siendo necesario distinguir el método de guardado en función de la plataforma desde la que se quiera volver a abrir la red creada.

En caso de querer seguir empleando la interfaz de Miniedit, se debe emplear la función **Save** en el menú de archivo, generando un archivo con extensión **.mn**, compatible con esta interfaz.

Por el contrario, si se desea poder seguir realizando las emulaciones en Mininet, se debe emplear la opción de **Export Level 2 Script**, la cual crea un documento con extensión **.py**, empleada en este entorno de trabajo.

Ambas opciones se encuentran en el menú de archivo Figura 23.

Resultados y conclusiones pendientes (Faltan)

Bibliografía

- <https://ieeexplore.ieee.org/abstract/document/9429224>
- <https://repository.udistrital.edu.co/items/93cb54ec-c169-4a2f-833a-b8b9bb922572>
- [https://d1wqtxts1xzle7.cloudfront.net/35217763/SDN-libre.pdf?1413880862=&response-content-disposition=inline%3B+filename%3DSDN Redes definidas por Software.pdf&Expires=1733526694&Signature=XjHPexDRA2-OpxGHOJh1UBsOmdi7mQl7am7dfayV46wxx~WSYYXMPW8QDXYZzEDxt4R8jIMTJ9kzOj5OkIbTbwuGJlI34W727zKuUelMLddfL-glN3YN3lyn0kow3QZx8JTY0TU8l28BR5BF7ssJyOnrgcXzCVoUzFQ1pw2rLf~w4N5rYuGYk5RBScWTWJA87spwUPHN40sz4GSCTZ8nEsZ3pkm6i9-hKtpZPyJjYuzv80LXOMC1Qi3Z1k05e1FlrCX0K1Ah3GttHsSxErvxA1o6-OoheaAzY7qe1wJI0c45vNGZRMhEB8TwGxqnl9X6of2Tr-PsmVlrCy1sqOwA0g__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA](https://d1wqtxts1xzle7.cloudfront.net/35217763/SDN-libre.pdf?1413880862=&response-content-disposition=inline%3B+filename%3DSDN+Redes+definidas+por+Software.pdf&Expires=1733526694&Signature=XjHPexDRA2-OpxGHOJh1UBsOmdi7mQl7am7dfayV46wxx~WSYYXMPW8QDXYZzEDxt4R8jIMTJ9kzOj5OkIbTbwuGJlI34W727zKuUelMLddfL-glN3YN3lyn0kow3QZx8JTY0TU8l28BR5BF7ssJyOnrgcXzCVoUzFQ1pw2rLf~w4N5rYuGYk5RBScWTWJA87spwUPHN40sz4GSCTZ8nEsZ3pkm6i9-hKtpZPyJjYuzv80LXOMC1Qi3Z1k05e1FlrCX0K1Ah3GttHsSxErvxA1o6-OoheaAzY7qe1wJI0c45vNGZRMhEB8TwGxqnl9X6of2Tr-PsmVlrCy1sqOwA0g__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA)
- <https://www.ibm.com/mx-es/topics/sdn>
- https://www.cisco.com/c/es_mx/solutions/software-defined-networking/overview.html
- <https://www.nutanix.com/mx/info/software-defined-networking>
- <https://es.linkedin.com/advice/3/how-do-you-handle-network-failures-recovery?lang=es>
- <https://investigacion.konradlorenz.edu.co/2017/02/robustez-y-auto-recuperacion-de-fallas-en-redes-complejas-utilizando-tecnicas-inspiradas-en-la-naturaleza.html>
- <https://www.e-dea.co/tendencias-2024-en-monitoreo-de-redes>
- <https://community.fs.com/es/article/software-defined-networking-sdn-types-advantages-and-applications.html>
- <https://www.itsinfocom.com/redes-inteligentes-con-cisco-sdn/>
- <https://www.kyndryl.com/mx/es/services/network/software-defined#:~:text=Las%20SDN%20le%20permiten%20transformar,y%20habilitada%20para%20la%20nube.>
- <https://www.telefonicaempresas.es/grandes-empresas/blog/casos-de-uso-de-sdn-la-busqueda-continua/>
- <https://www.tecnous.com/como-testear-sdn/>
- https://oa.upm.es/70551/1/TFG_JUAN_ANDRES_MARTINEZ_HONTANAYA.pdf

