



Instituto Politécnico Nacional
Escuela Superior de Cómputo
Ingeniería en Sistemas Computacionales



Proyecto Final GNS3

Profesora: Henestrosa Carrasco Leticia

Grupo: 4CV12

Alumnos:

- Ayala Segoviano Donaldo Horacio
- Sánchez Valdivia Natalia Lisset
- Salvador Bucio Pedro Armando
- Sulvarán Solache Bernabé Guadalupe

1 Introducción

El presente reporte explica el desarrollo de implementación de una topología emulando la red y simulando equipos tales como los servidores SNMP, DNS, DHCP, HTTP y TFTP, además de utilizar el router 7200 de cisco en nuestra simulación, se pretende tener al menos 5 routers del mismo tipo interactuando y usando el protocolo de enrutamiento RIPv2,

Se planteaba usar al inicio una máquina física dado que la capacidad que se requiere para simular toda la topología en GNS3 es bastante y el costo en Azure, Amazon u otras plataformas como Google Cloud Platform son muy caras, pero finalmente se decidió usar Azure con una máquina virtual Ubuntu Server 20.04.3 sin entorno gráfico, dado que permite a los usuarios conectarse remotamente y trabajar en conjunto sobre el mismo proyecto en GNS3 que debe ser instalado en la máquina Virtual en Azure.

1.1 Azure

Microsoft Azure (anteriormente Windows Azure y Azure Services Platform) es un servicio de computación en la nube creado por Microsoft para construir, probar, desplegar y administrar aplicaciones y servicios mediante el uso de sus centros de datos. Proporciona software como servicio (SaaS), plataforma como servicio (PaaS) e infraestructura como servicio (IaaS) y es compatible con muchos lenguajes, herramientas y marcos de programación diferentes, incluidos software y sistemas específicos de Microsoft y de terceros.

Microsoft Azure utiliza un sistema operativo especializado, llamado de la misma forma, para correr sus "capas" (en inglés "fabric layer") un cluster localizado en los servidores de datos de Microsoft que se encargan de manejar los recursos almacenados y procesamiento para proveer los recursos (o una parte de ellos) para las aplicaciones que se ejecutan sobre Microsoft Azure.

Estos funcionan bajo la versión 2008 de Windows Server y una versión personalizada de Hyper-V, conocido como el Hipervisor de Microsoft Azure que provee la virtualización de los servicios. La capa controladora de Microsoft Azure se encarga de escalar y de manejar la confiabilidad del sistema evitando así que los servicios se detengan si alguno de los servidores de datos de Microsoft tiene problemas y a su vez maneja la información de la aplicación web del usuario dando como ejemplo los recursos de la memoria o el balanceo del uso de esta.

1.2 Docker

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos. Docker utiliza características de aislamiento de recursos del kernel Linux, tales como cgroups y espacios de nombres (namespaces) para permitir que "contenedores" independientes se ejecuten dentro de una sola instancia de Linux, evitando la sobrecarga de iniciar y mantener máquinas virtuales.

Cómo funciona Docker

Docker le proporciona una manera estándar de ejecutar su código. Docker es un sistema operativo para contenedores. De manera similar a cómo una máquina virtual virtualiza (elimina la necesidad de administrar directamente) el hardware del servidor, los contenedores virtualizan el sistema operativo de un servidor. Docker se instala en cada servidor y proporciona comandos sencillos que puede utilizar para crear, iniciar o detener contenedores.

1.3 Servidor HTTP Apache

El servidor HTTP Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), que implementa el protocolo HTTP/1.1 y la noción de sitio virtual según la normativa RFC 2616. Cuando comenzó su desarrollo en 1995 se basó inicialmente en el código del popular NCSA HTTPd 1.3, pero más tarde fue reescrito por completo.

El servidor Apache es desarrollado y mantenido por una comunidad de usuarios bajo la supervisión de la Apache Software Foundation dentro del proyecto HTTP Server (httpd).

Apache tiene amplia aceptación en la red: desde 1996, Apache es el servidor HTTP más usado. Jugó un papel fundamental en el desarrollo de la World Wide Web y alcanzó su máxima cuota de mercado en 2005, siendo el servidor empleado en el 70% de los sitios web en el mundo. Sin embargo, ha sufrido un descenso en su cuota de mercado en los últimos años (estadísticas históricas y de uso diario proporcionadas por Netcraft). En 2009, se convirtió en el primer servidor web que aloja más de 100 millones de sitios web.

1.4 SNMP (Simple Network Administration Protocol)

SNMP significa protocolo simple de gestión de red, por sus siglas en inglés. Se trata de un protocolo para la gestión de la transferencia de información en redes, especialmente para uso en LAN, dependiendo de la versión elegida. Su utilidad en la gestión de redes proviene del hecho de que permite recopilar la información sobre los dispositivos conectados a la red de una forma estandarizada en una gran variedad de tipos de hardware y software. Los mensajes SNMP se envían y reciben entre los administradores y los agentes. Por lo general, el administrador de SNMP de la red se instala en la entidad administradora, y los agentes SNMP, en los dispositivos administrados.

1.5 DNS (Domain Name Service)

El servicio DNS (Domain Name Service) es un servicio de Internet que traduce los nombres de los dominios (direcciones por nombre, p. ej. www.unizar.es) en direcciones IP (direcciones numéricas, p. Ej. 155.210.3.32) y viceversa. Este servicio es imprescindible para poder iniciar cualquier comunicación con otro computador accediendo al mismo por su nombre.

En este sistema estructurado en árbol participan tres partes diferenciadas:

- Clientes DNS: es un programa cliente DNS usado por cualquier persona usuaria a través de su equipo o dispositivo para realizar una petición a través de la red (una web, un envío de correo, etc).
- Servidores DNS: es el software o máquina encargada de atender y dar respuesta a la petición de los clientes DNS, por ejemplo indicando en qué dirección se aloja una determinada página web. Los servidores recursivos tienen la capacidad de reenviar la solicitud hacia otro servidor en caso de que ellos no dispongan de la dirección solicitada.
- Zona de autoridad: servidores o grupos de servidores que se encargan de resolver un conjunto de dominios determinado (por ejemplo .ES, .COM, etc).

1.6 DHCP (Dynamic Host Configuration Protocol)

El DHCP es una extensión del protocolo Bootstrap (BOOTP) desarrollado en 1985 para conectar dispositivos como terminales y estaciones de trabajo sin disco duro con un Boot Server, del cual reciben

su sistema operativo. El DHCP se desarrolló como solución para redes de gran envergadura y ordenadores portátiles y por ello complementa a BOOTP, entre otras cosas, por su capacidad para asignar automáticamente direcciones de red reutilizables y por la existencia de posibilidades de configuración adicionales. La dirección asignada se guarda en la base de datos del servidor junto con la dirección MAC del cliente, con lo cual la configuración se hace permanente, es decir, el dispositivo se conecta a la red siempre con esa dirección que le ha sido asignada automáticamente y que ya no está disponible para ningún otro cliente, lo que significa que los clientes DHCP nuevos no pueden recibir ninguna dirección si ya están todas asignadas, incluso aunque algunas IP ya no se usen activamente.

1.7 TFTP (Trivial File Transfer Protocol)

TFTP son las siglas de Trivial file transfer Protocol (Protocolo de transferencia de archivos trivial). TFTP está destinado a las aplicaciones que no necesitan las interacciones sofisticadas que proporciona el protocolo de transferencia de archivos (FTP). TFTP, junto con el protocolo Bootstrap (BOOTP), proporciona soporte para los clientes de un producto System i. También proporcionan soporte para otros clientes que usan los protocolos TFTP y BOOTP.

2 Objetivo

El objetivo de la realización del presente proyecto es el de aplicar los conocimientos adquiridos en la materia de administración de servicios en red en el emulador GNS3, el cual nos permitirá familiarizarnos con la implementación y administración de servicios en un ambiente muy cercano al real, y además nos permitirá reafirmar los conocimientos adquiridos.

3 Desarrollo

3.1 Configuración del proyecto en Azure

3.1.1 Creación de una Máquina Virtual en Azure

Para crear una máquina virtual, es necesario tener una cuenta en Azure, y además tener créditos en esa cuenta para poder hacer uso de los distintos servicios ofrecidos. Una vez que tengamos una cuenta con suficientes créditos, podemos seguir los siguientes pasos para crear la máquina virtual.

Desde el portal en máquinas virtuales hacemos clic en '+Crear > +Máquina Virtual' como se muestra en la figura 1.

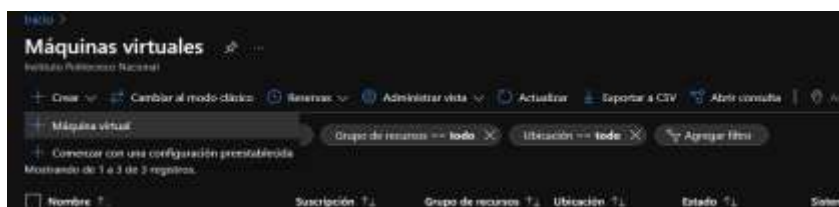


Figura 1. Creación de máquina Virtual.

Agregamos grupo de recursos (opcional, se puede crear por defecto), nombre de la máquina, imagen (en este caso Ubuntu Server 20.04 LTS) y capacidad de la máquina en este caso 2 vcpu y 8 de RAM. Como se puede observar en la figura 2.

Figura 2. Asignación de recursos.

Más abajo, agregamos usuario y contraseña y dejamos puerto de entrada ssh (22) por defecto, como se muestra en la figura 3.

Figura 3. Configuración de credenciales.

En la pestaña discos, preferentemente cambiamos a HDD por cuestión de costos.

Figura 4. Configuración de almacenamiento.

Deshabilitamos el diagnóstico de arranque en pestaña administración:

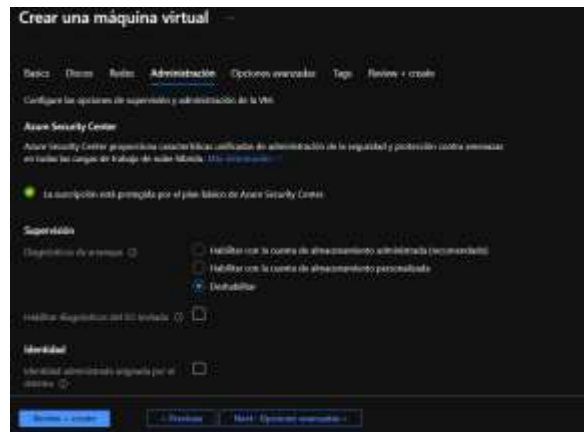


Figura 5. Deshabilitar del diagnóstico de arranque.

Por el momento es suficiente, damos en revisar y crear y si todo está bien en crear:

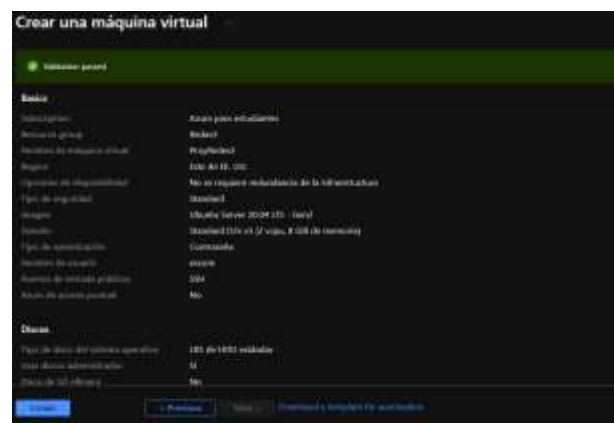


Figura 6. Verificación de configuraciones.

Después nos aparece la máquina y podremos trabajar con ella:



Figura 6. Visualización de la máquina virtual.

Una vez que tengamos nuestra máquina virtual creada, podremos conectarnos vía SSH a esta. En la siguiente figura se muestra el prompt remoto de la máquina virtual.

Desde: 'Edit>preferences>Server' en el menú superior izquierdo:

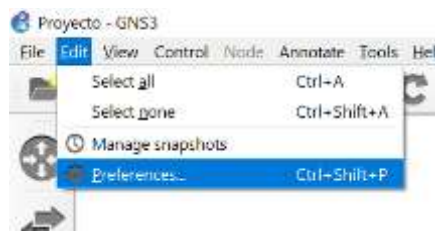


Figura 9. Acceder a la configuración.

Nos dirigimos a la sección 'Server', y deshabilitamos la casilla 'Enable local server'. Una vez deshabilitada, se nos mostrará un formulario que donde insertamos los datos y credenciales de nuestra máquina virtual:

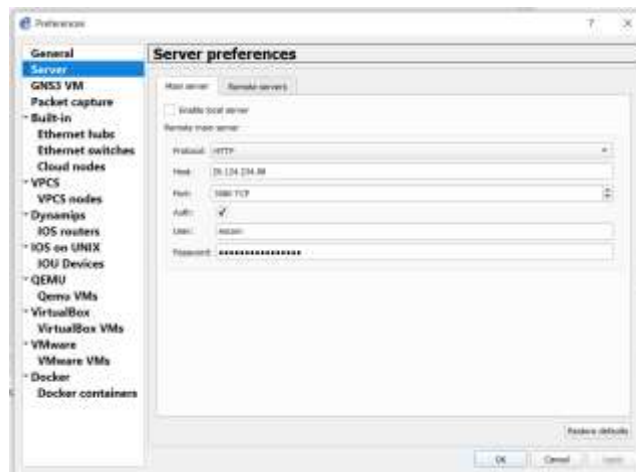


Figura 10. Configuración de credenciales de la máquina virtual.

Una vez realizada esta configuración, podremos trabajar remotamente en el proyecto creado de forma remota y paralela.

3.2 Instalación de Docker

Para la realización del proyecto, se decidió usar imágenes y contenedores de Docker, puesto que son más livianos que una imagen de sistema operativo completa. Para instalar Docker en la máquina virtual de Azure, debemos conectarnos vía SSH con las credenciales necesarias. Una vez conectados a la Máquina Virtual de Azure vía SSH, ejecutamos los siguientes comandos para instalar Docker.

```
sudo apt-get update
sudo apt-get install \ apt-transport-https \ ca-certificates \ curl \ gnupg-agent \ software-properties-
common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository \ "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) \ stable"
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

Una vez ejecutados, podemos ejecutar el comando '`sudo docker version`' para verificar que docker se instaló adecuadamente.

```
escom@gnslvm:~$ sudo docker version
Client: Docker Engine - Community
Version: 20.10.11
API version: 1.41
Go version: go1.18.9
Git commit: 0603b6
Built: Thu Nov 18 00:37:00 2021
OS/Arch: linux/amd64
Context: default
Experimental: true

Server: Docker Engine - Community
Engine:
Version: 20.10.11
API version: 1.41 (minimum version 1.12)
Go version: go1.18.9
Git commit: 0603b6
Built: Thu Nov 18 00:37:00 2021
OS/Arch: linux/amd64
Experimental: false
containerd:
Version: 1.4.12
GitCommit: 8f4b969140690b48f3b03c1e7205e3a47b62459e94
runc:
Version: 1.0.2
GitCommit: v1.0.2-0-g52b3a62
docker-init:
Version: 0.19.0
GitCommit: 060ba00
```

Figura 11. Verificación de instalación de Docker.

Una vez que tenemos docker instalado, podemos comenzar a crear contenedores para gestionar los programas instalados y hacer las configuraciones necesarias. Podemos crear contenedores con diferentes imágenes en el repositorio de Docker. El siguiente comando crea un contenedor a partir de una imagen de Ubuntu versión 18.04.

```
sudo docker run -it ubuntu:18.04 /bin/bash
```

Ahora, para iniciar el contenedor, usaremos el id del contenedor previamente creado. Para ver los contenedores actualmente creados, se puede ejecutar el siguiente comando.

```
escom@gnslvm:~$ sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS          NAMES
74fc556fb6ee   ubuntu:18.04   "/bin/bash"             6 minutes ago   Exited (0)   About a minute ago   focused_turing
```

Figura 12. Consulta de los contenedores creados.

Se pueden usar los siguientes comandos para ejecutar un contenedor y para poder acceder a una consola dentro de este contenedor.

```
docker start 74fc556fb6ee
docker attach 74fc556fb6ee
```

Dentro del contenedor se puede instalar un servicio tal cual fuese una máquina virtual de Ubuntu.

```
escom@gnslvm:~$ sudo docker attach 74fc556fb6ee
You cannot attach to a stopped container, start it first
escom@gnslvm:~$ sudo docker start 74fc556fb6ee
74fc556fb6ee
escom@gnslvm:~$ sudo docker attach 74fc556fb6ee
root@74fc556fb6ee:/#
```

Figura 13. Conexión a un contenedor.

De igual forma, para acceder a una terminal de un contenedor, se puede ejecutar el siguiente comando.

```

maco@ms3ver:~$ sudo dockerps
sudo: dockerps: command not found
maco@ms3ver:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
73b85ceff69c   gns3:v1   "/bin/bash"   23 hours ago   Up 6 minutes   split
maco@ms3ver:~$ sudo docker exec 73b85ceff69c /bin/bash
root@4467a5b583c:/# ls
bin  dev  home  lib32  libx32  mnt  proc  run  srv  tmp  var
boot  etc  lib   lib64  media  opt  root  sbin  sys  usr
root@4467a5b583c:/#

```

Figura 14. Conexión a un contenedor.

3.3 Diseño de la topología

Se decidió implementar una simulación la topología de la red del Politécnico, tomando los routers más importantes de cada región, que en este caso son UPIICSA, ZACATENCO, SANTO TOMÁS, y adicionalmente, se extendieron algunos routers, como son CECyT 2 y CECyT 1. A continuación se muestra la tabla de direcciones de la topología.

Dispositivo	Interfaz	Dirección IP	Máscara de subred
Zacatenco	Fa0/0	187.190.16.21	255.255.255.252
	Fa1/0	187.190.4.1	255.255.252.0
	Se4/0	187.190.16.6	255.255.255.252
	Se4/1	187.190.16.10	255.255.255.252
Laboratorio	Fa0/0	187.190.4.13	255.255.252.0
	Fa1/0	192.168.16.1	255.255.255.0
Santo Tomas	Se4/0	187.190.16.2	255.255.255.252
	Se4/2	187.190.16.5	255.255.255.252
	Se4/3	187.190.16.13	255.255.255.252
CECyT 2	Fa0/0	187.190.0.1	255.255.248.0
	Se4/0	187.190.16.14	255.255.255.252
UPIICSA	Se4/0	187.190.16.1	255.255.255.252
	Se4/1	187.190.16.9	255.255.255.252
	Se4/2	187.190.16.17	255.255.255.252
CECyT 1	Fa0/0	187.190.2.1	255.255.255.128
	Fa0/0.2	187.190.2.129	255.255.255.128
	Fa0/0.3	187.190.3.1	255.255.255.128
	Se4/0	187.190.16.18	255.255.255.252

Una vez teniendo la tabla de configuración de los routers, se procede a implementar la topología en GNS3. La figura 15 muestra la topología implementada en el emulador GNS3.

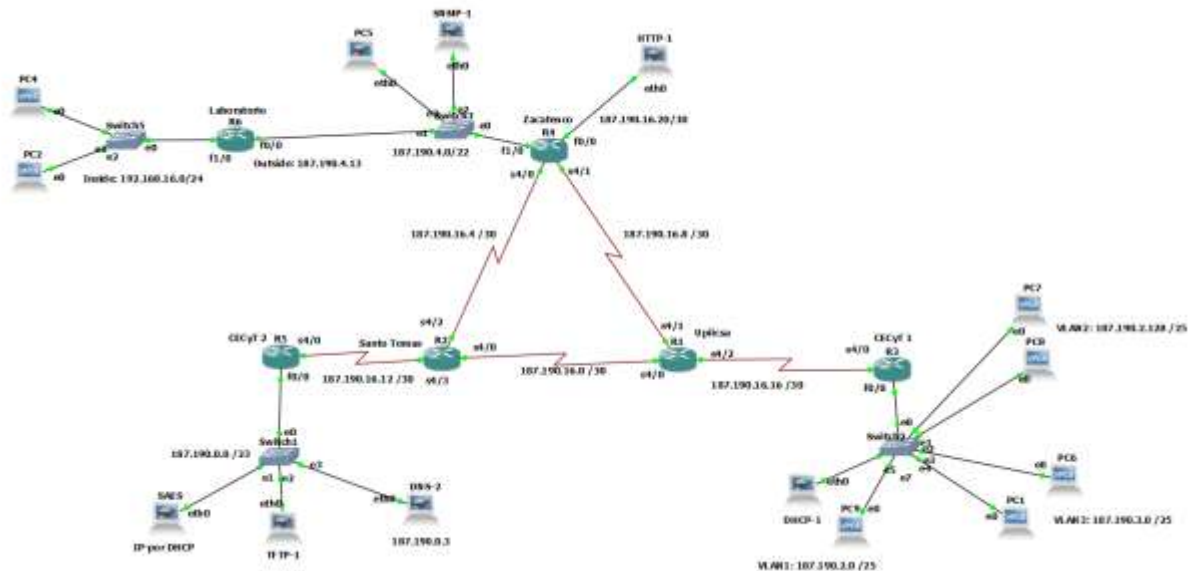


Figura 15. Topología del proyecto.

3.4 Configuración de routers

En la topología mostrada utilizamos 5 routers, a los cuales se le realizaron las configuraciones básicas. A continuación se muestran los scripts de configuración de los routers.

```
Router R1 UPIICSA
enable
configure terminal
hostname UPIICSA
enable secret escom
line console 0
password escom
login
exit
line vty 0 15
password escom
login
exit

interface s4/0
ip address 187.190.16.1 255.255.255.252
no shutdown
exit
interface s4/1
ip address 187.190.16.0 255.255.255.252
no shutdown
exit
interface s4/2
ip address 187.190.16.17 255.255.255.252
no shutdown
exit
```

```
Router R2 SANTOTOMAS
enable
configure terminal
hostname SANTOTOMAS
enable secret escom
line console 0
password escom
login
exit
line vty 0 15
password escom
login
exit

interface s4/2
ip address 187.190.16.5 255.255.255.252
no shutdown
exit
interface s4/0
ip address 187.190.16.2 255.255.255.252
no shutdown
exit
interface s4/3
ip address 187.190.16.13 255.255.255.252
no shutdown
exit
```

```
Router R3 CECYT1
enable
configure terminal
hostname CECYT1
enable secret escom
line console 0
password escom
login
exit
line vty 0 15
password escom
login
exit

interface fa0/0
ip address 187.190.2.1 255.255.254.0
no shutdown
exit
interface s4/0
ip address 187.190.16.10 255.255.255.252
no shutdown
exit
```

```

ROUTER R4 ZACATENCO

enable
configure terminal
hostname ZACATENCO
enable secret escom
line console 0
password escom
login
exit
line vty 0 15
password escom
login
exit
interface fa0/0
ip address 187.190.16.21 255.255.255.252
ip helper-address 187.190.2.2
no shutdown
exit
interface fa1/0
ip address 187.190.4.1 255.255.252.0
ip helper-address 187.190.2.2
no shutdown
exit
interface s4/0
ip address 187.190.16.6 255.255.255.252
no shutdown
exit
interface s4/1
ip address 187.190.16.10 255.255.255.252
no shutdown
exit

ROUTER R5 CECYT2

enable
configure terminal
hostname CECYT2
enable secret escom
line console 0
password escom
login
exit
line vty 0 15
password escom
login
exit
interface fa0/0
ip address 187.190.0.1 255.255.254.0
ip helper-address 187.190.2.2
no shutdown
exit
interface s4/0
ip address 187.190.16.14 255.255.255.252
no shutdown
exit

```

Figura 16. Scripts de configuración básica de los routers.

3.5 Protocolo de enrutamiento

Se decidió que el protocolo de enrutamiento fuera RIPv2. Su principal limitación está impuesta por la cantidad máxima de saltos que soporta: 15. RIP asume que todo lo que se encuentra a más de 15 saltos, está a una distancia infinita, y por lo tanto no tiene ruta válida. Sus principales características son la distancia administrativa que es 120, RIPv2 envía actualizaciones de enrutamiento a través de la dirección de multicast y su métrica es el número de saltos.

A continuación, se muestran algunas capturas de las tablas de enrutamiento de uno de los routers principales de la topología, así como los comandos usados para configurar el protocolo. Se puede observar que se empiezan a conocer los diferentes segmentos de red de la topología, las entradas con una 'R' al inicio son todos los segmentos que se conocieron a través de RIPv2.

```

187.190.0.0/16 is variably subnetted, 13 subnets, 5 masks
R   187.190.0.0/23 [120/2] via 187.190.16.2, 00:00:07, Serial4/0
R   187.190.2.0/25 [120/1] via 187.190.16.18, 00:00:35, Serial4/2
R   187.190.2.128/25 [120/1] via 187.190.16.18, 00:00:35, Serial4/2
R   187.190.3.0/25 [120/1] via 187.190.16.18, 00:00:35, Serial4/2
R   187.190.4.0/22 [120/1] via 187.190.16.10, 00:00:13, Serial4/1
C   187.190.16.0/30 is directly connected, Serial4/0
L   187.190.16.1/32 is directly connected, Serial4/0
R   187.190.16.4/30 [120/1] via 187.190.16.10, 00:00:13, Serial4/1
      [120/1] via 187.190.16.2, 00:00:07, Serial4/0
C   187.190.16.8/30 is directly connected, Serial4/1
L   187.190.16.9/32 is directly connected, Serial4/1
R   187.190.16.12/30 [120/1] via 187.190.16.2, 00:00:08, Serial4/0
C   187.190.16.16/30 is directly connected, Serial4/2
L   187.190.16.17/32 is directly connected, Serial4/2

router rip
version 2
network 187.190.16.0
network 187.190.16.8
network 187.190.16.16
no auto-summary
exit

```

Figura 17. Tabla de enrutamiento y comandos de configuración de RIPv2 del router UPIICSA.

3.6 VLSM (Variable Length Subnet Mask)

VLSM permite un uso más eficiente de las direcciones ip. También permite dividir un espacio de red en partes desiguales. La red primero se divide en subredes y, a continuación, las subredes se vuelven a dividir en subredes.

Para el proyecto, se inició con una dirección ip pública de clase B. A partir de esta. A partir de esta, se implementó VLSM para poder crear las subredes necesarias. Para los enlaces WAN se utilizaron subredes con máscaras de prefijo 30, lo cual nos da exactamente las dos ip's necesarias. Para el resto de las subredes, se usaron diferentes máscaras de subred, simulando una diferente demanda de usuarios por cada subred.

3.7 VLAN (Virtual Local Area Network)

Nos permite crear redes lógicas independientes dentro de la misma red física. En la topología se implementaron 3 VLAN en el segmento de red del CECyT 1, en el switch 2 como se puede ver en la figura 18. En el switch se configuraron 3 VLAN's, VLAN 1, VLAN 2 y VLAN 3. La configuración se realizó mediante la interfaz gráfica del switch de GNS3, ya que estos switches no permiten tener acceso a una consola.

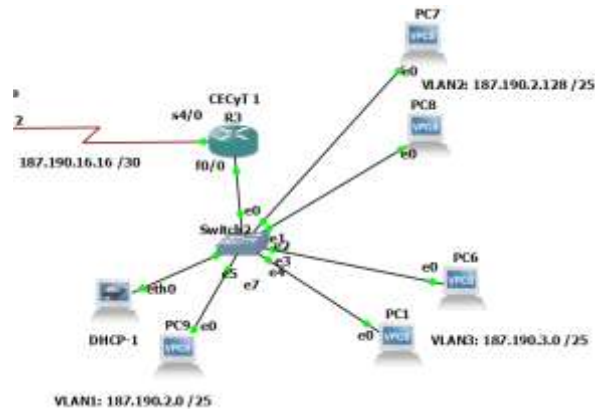


Figura 18. VLAN's configuradas en el Switch 2.

A continuación, en la figura 19, se muestra la configuración realizada mediante la interfaz. Fue necesario configurar el puerto 0 con el protocolo dot1q, ya que es el protocolo usado para VLAN tagging e InterVLAN. Esto para que estas VLAN puedan tener acceso al resto de la red.

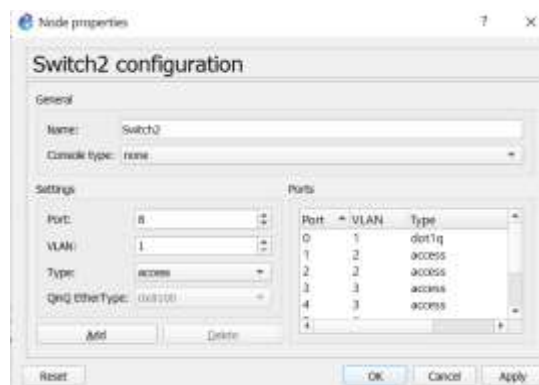


Figura 19. Configuración de VLAN's.

Adicionalmente, se implementó la configuración *Router on a Stick* la cual nos permite conectar todas las VLAN implementadas del switch usando solamente una interfaz. Esta configuración se realiza en el router del CECyT 2. A continuación, en la figura 20, se muestran los comandos para configurar router on a stick en el router.

```

CECVT1(config)#int fa0/0
CECVT1(config-if)#ip address 187.190.2.1 255.255.255.128
CECVT1(config-if)#exit
CECVT1(config)#int fa0/0.2
CECVT1(config-subif)#encapsulation dot1q 2
CECVT1(config-subif)#ip address 187.190.2.129 255.255.255.128
CECVT1(config-subif)#exit
CECVT1(config)#int fa0/0.3
CECVT1(config-subif)#encapsulation dot1q 3
CECVT1(config-subif)#ip address 187.190.3.1 255.255.255.128
CECVT1(config-subif)#exit
CECVT1(config)#

```

Figura 20. Configuración de router on a stick.

Ahora, las VLAN de este switch pueden acceder a la red, y por lo tanto, entre ellas. Sin embargo, se planteó el caso donde la comunicación está prohibida entre la VLAN 2 y la VLAN 3. Por lo que se implementó la siguiente ACL y se asignó a las subinterfaces de las VLAN 2 y 3. En la figura 21 se muestra la configuración de la ACL.

```

CECVT3(config)#ip access-list extended vlan23
CECVT3(config-ext-nacl)#deny ip 187.190.2.0 0.0.0.127 187.190.3.0 0.0.0.127
CECVT3(config-ext-nacl)#deny ip any any
CECVT3(config-ext-nacl)#exit
CECVT3(config)#int fa0/0.2
CECVT3(config-subif)#ip access-group vlan23 in
CECVT3(config-subif)#exit
CECVT3(config)#int fa0/0.3
CECVT3(config-subif)#ip access-group vlan23 in
CECVT3(config-subif)#exit

```

Figura 21. Creación y configuración de ACL's para VLAN's 2 y 3.

La figura 22 muestra como las VLAN tienen acceso a toda la red, sin embargo, cuando se trata de hacer comunicación entre las VLAN, esta no es posible, y, por último, se comprueba la conexión entre las computadoras pertenecientes al mismo segmento de red.

```

PCB> ping 187.190.4.1
64 bytes from 187.190.4.1: icmp_seq=1 ttl=253 time=56.184 ms
64 bytes from 187.190.4.1: icmp_seq=2 ttl=253 time=56.189 ms
64 bytes from 187.190.4.1: icmp_seq=3 ttl=253 time=56.444 ms
64 bytes from 187.190.4.1: icmp_seq=4 ttl=253 time=56.212 ms
64 bytes from 187.190.4.1: icmp_seq=5 ttl=253 time=57.118 ms

PCB> ping 187.190.3.1
*187.190.2.129 icmp_seq=1 ttl=253 time=6.409 ms (ICMP type=8, code=13, communication administratively prohibited)
*187.190.2.129 icmp_seq=2 ttl=253 time=7.383 ms (ICMP type=8, code=13, communication administratively prohibited)
*187.190.2.129 icmp_seq=3 ttl=253 time=69.061 ms (ICMP type=3, code=13, communication administratively prohibited)
*187.190.2.129 icmp_seq=4 ttl=253 time=6.448 ms (ICMP type=8, code=13, communication administratively prohibited)
*187.190.2.129 icmp_seq=5 ttl=253 time=6.399 ms (ICMP type=1, code=13, communication administratively prohibited)

PCB> ping 187.190.2.128
64 bytes from 187.190.2.128: icmp_seq=1 ttl=64 time=6.188 ms
64 bytes from 187.190.2.128: icmp_seq=2 ttl=64 time=6.121 ms
64 bytes from 187.190.2.128: icmp_seq=3 ttl=64 time=6.121 ms
64 bytes from 187.190.2.128: icmp_seq=4 ttl=64 time=6.484 ms
64 bytes from 187.190.2.128: icmp_seq=5 ttl=64 time=6.462 ms

```

Figura 21. Comprobación de VLAN's y ACL.

3.8 PAT (Port Address Translation)

Se planteó el escenario, donde dentro de alguna escuela de Zacatenco, se tiene un laboratorio, y varios usuarios van a estar conectados a esta red, por lo que, para no usar tantas direcciones, se implementó una PAT. Esto permite que los usuarios de esta subred, puedan salir al resto de la red, pero usando solo una dirección ip "pública". Las ip privadas son todas las ip de la red 192.168.16.0/24 y la ip pública es la ip 187.190.4.13/22. La figura 22 muestra la configuración en el router de esta PAT, además se configuró DHCP en el router para que los usuarios puedan obtener automáticamente una dirección IP privada.

```

ip nat pool NATPOOL 187.190.4.13 187.190.4.13 netmask 255.255.252.0
ip access-list standard LOCALACL
permit 192.168.16.0 0.0.0.255
exit
ip nat inside source list LOCALACL pool NATPOOL overload

int fa1/0
ip nat inside
ip address 192.168.16.1 255.255.255.0
no shutdown

int fa0/0
ip address 187.190.4.13 255.255.252.0
ip nat outside
no shutdown

```

Figura 22. Comandos de configuración de PAT.

En la figura 22 se pueden observar los comandos necesarios para configurar la PAT en el router 6, que es el del laboratorio de la escuela. Se puede ver que se debe de crear un pool de direcciones para especificar que direcciones ip se pueden usar para asignar a los usuarios que vayan a salir a la red, en este caso, solo se asigna una para que sea usada por todos estos usuarios para salir a la red. Además, se crea una ACL que indica que usuarios pueden salir a la red y usar esta PAT. Finalmente, se configuran las interfaces como inside y outside.

En la figura 23 se muestra como una pc pide mediante DHCP una dirección ip y se le asigna una dirección privada disponible, y se muestra como esta pc puede salir al resto de la red mediante la PAT.

```
PC4> ip dhcp
DORA IP 192.168.16.2/24 GW 192.168.16.1

PC4> ping 187.190.16.22

84 bytes from 187.190.16.22 icmp_seq=1 ttl=62 time=36.809 ms
84 bytes from 187.190.16.22 icmp_seq=2 ttl=62 time=39.172 ms
84 bytes from 187.190.16.22 icmp_seq=3 ttl=62 time=39.812 ms
84 bytes from 187.190.16.22 icmp_seq=4 ttl=62 time=38.449 ms
84 bytes from 187.190.16.22 icmp_seq=5 ttl=62 time=39.336 ms

PC4>
```

v

3.9 SSH (Secure Shell)

Se implementó este servicio en la máquina HTTP-1. Se utilizó el programa de Linux Openssh, para implementar este servicio. En la figura 24 se muestra como iniciar este servicio y crear un nuevo usuario. Además, es necesario crear un nuevo usuario para que se pueda acceder remotamente a este mediante ssh como se podrá ver en la siguiente sección.

```
root@HTTP-1:/# /etc/init.d/ssh start
* Starting OpenSSH Secure Shell server sshd
root@HTTP-1:/# adduser newuser
Adding user 'newuser' ...
Adding new group 'newuser' (1005) ...
Adding new user 'newuser' (1001) with group 'newuser' ...
Creating home directory '/home/newuser' ...
Copying files from '/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for newuser:
Enter the new value, or press ENTER for the default
Full Name []:
Room Number []:
Work Phone []:
Home Phone []:
Other []:
Is the information correct? [Y/n]
root@HTTP-1:/#
```

Figura 24. Ping desde dentro del laboratorio.

En la siguiente sección se comprobará el funcionamiento de este servicio.

3.10 ACL (Access Control List) Extendida

En la topología podemos encontrar 3 ACL : la primera esta dentro de la VLAN donde la VLAN 3 y 2 no pueden acceder entre ellas, pero si pueden acceder a los demás segmentos en la red. Podemos poner a prueba esta ACL con un ping entre cualquier pc de vlan 2 a 3 hacemos la prueba con un ping de PC7 a PC6. Esto se mostró en la sección de VLAN's.

Se implementó otra ACL en la subred del router 6 en la PAT se integró una acl para especificar qué redes pueden salir de la subred. Y por último, se tiene una tercera ACL en el router de Zacatenco, esta ACL permite la conexión ssh y telnet solamente de la máquina SNMP-1 hacia el servidor HTTP-1 y deniega cualquier otro intento de conexión ssh o telnet. Sin embargo, solo se restringe este servicio, pues cualquier computadora de la topología, puede seguir accediendo a la página mediante HTTP.

La figura 24 muestra los comandos para configurar la ACL en el router de Zacatenco, y como se asigna a la interfaz correspondiente.

```

ZACATEMCOV2(config)#tcp access-list extended ssh
ZACATEMCOV2(config-ext-nacl)#host 187.190.4.15 host 187.190.16.22 eq 22
ZACATEMCOV2(config-ext-nacl)#host 187.190.4.15 host 187.190.16.22 eq 23
ZACATEMCOV2(config-ext-nacl)#deny tcp any any eq 22
ZACATEMCOV2(config-ext-nacl)#deny tcp any any eq 23
ZACATEMCOV2(config-ext-nacl)#permit ip any any
ZACATEMCOV2(config-ext-nacl)#do sh access-lists
Extended IP access list ssh
10 permit tcp host 187.190.4.15 host 187.190.16.22 eq 22
20 permit tcp host 187.190.4.15 host 187.190.16.22 eq telnet
30 deny tcp any any eq 22
40 deny tcp any any eq telnet
50 permit ip any any
ZACATEMCOV2(config-ext-nacl)#exit
ZACATEMCOV2(config)#int fa0/0
ZACATEMCOV2(config-if)#ip access
ZACATEMCOV2(config-if)#ip access-group ssh out
ZACATEMCOV2(config-if)#exit
ZACATEMCOV2(config)#exit

```

Figura 25. Configuración para restringir el acceso mediante ssh o telnet al servidor HTTP.

Para verificar esta configuración, se intentará conectar con ssh desde SNMP-1 y después desde otro equipo, al igual, se probará que todas las máquinas tengan acceso a HTTP. La figura 25 muestra como desde la máquina SNMP-1 se puede acceder por ssh a HTTP-1, al igual que como se tiene acceso a HTTP desde la misma máquina. Mientras que la figura 26, muestra como desde la PC5, no se puede acceder a ssh en HTTP-1 pero si se tiene acceso a HTTP en HTTP-1.

```

root@ubuntu-18.04-arm:~# ssh root@197.196.16.22
The authenticity of host '197.196.16.22 (197.196.16.22)' can't be established.
ECDSA key fingerprint is SHA256:OCvCz1I8RwvCtDjPQWUqBmFmGZrC2HcM8dKbnkAas.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '197.196.16.22' (SHA256) to the list of known hosts.
root@197.196.16.22's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.11.8-smp-Ubuntu SMP Wed Aug 26)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/support

This system has been minimized by removing packages and content that are
not required in a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Could not chdir to home directory /home/ubuntu: No such file or directory
root@197.196.16.22 ~#
```

Figura 26. Conexión ssh y http desde SNMP1 hacia HTTP-1.

[illegible]

Figura 27. Conexión ssh denegada y conexión http exitosa desde PC5 hacia HTTP-1.

3.11 Servidor HTTP con Apache2

Para implementar el servidor HTTP, se hará uso del servidor web Apache2 en Ubuntu. Para esto, se creará un contenedor con las herramientas necesarias para poder correr este servicio. Para crear una imagen de Ubuntu con las herramientas necesarias, se hizo uso de un Dockerfile, el cual se muestra a continuación.


```

1 FROM ubuntu
2 RUN apt update
3 RUN apt update
4 ENV TZ=America/Mexico_City
5 RUN ln -sf /usr/share/zoneinfo/$TZ /etc/localtime && echo $TZ > /etc/timezone
6 RUN apt install vim git curl -y
7 RUN apt install iputils-ping -y
8 RUN apt install iproute2 -y
9 RUN apt install net-tools -y
10 RUN apt install systemctl -y
11 RUN apt install apache2 -y

```

Figura 28. Dockerfile para el servicio de HTTP.

Para poder construir una imagen a partir de la configuración mostrada en el Dockerfile, es necesario situarse en la carpeta donde se encuentre el Dockerfile y correr el siguiente comando.

```

escom@gns3vm:~/HTTP$ sudo docker build -t http:v1.0 .

```

Se necesitan permisos de superusuario, se hace uso del comando de docker *build*, la bandera *-t* indica el tag de la imagen que en este caso es *http:v1.0* y se agrega un *'.'* para indicarle que el dockerfile está en la carpeta actual.

Para comprobar que se haya creado la imagen, corremos el comando con permisos de superusuario *sudo docker images* que nos mostrará las imágenes guardadas en el repositorio de imágenes de dockers. En la siguiente imagen se muestra la salida del comando y se puede observar que la imagen fue creada satisfactoriamente.

```

escom@gns3vm:~/HTTP$ sudo docker images
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
http                 v1.0       f2c282434a29  10 minutes ago  338MB

```

Ahora, es necesario agregar un nuevo contenedor a GNS3 seleccionando la imagen creada. Para esto es necesario seguir los siguientes pasos:

Ir a la sección de Edit > Preferences y hacemos click.



En la ventana desplegada hacemos click en Docker containers y hacemos click en new.



Seleccionamos la imagen creada anteriormente, y hacemos click en next.



Se asigna un nombre al nuevo contenedor y se da click en next hasta finalizar la creación del recurso.



Una vez que observemos el recurso creado en los contenedores, damos click en apply y ok.

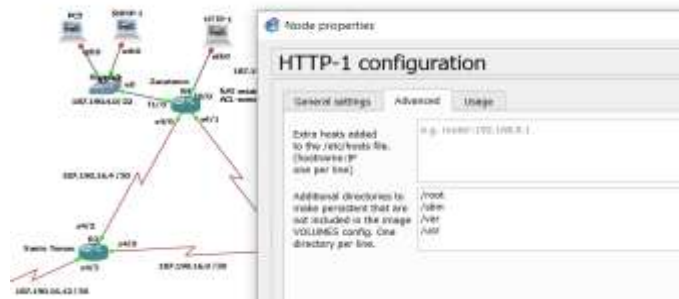


Una vez hecho esto, podremos agregar dispositivos con dicha imagen cargada.



Una vez tengamos este recurso, podremos seleccionarlo desde el menú izquierdo y arrastrarlo a la topología.

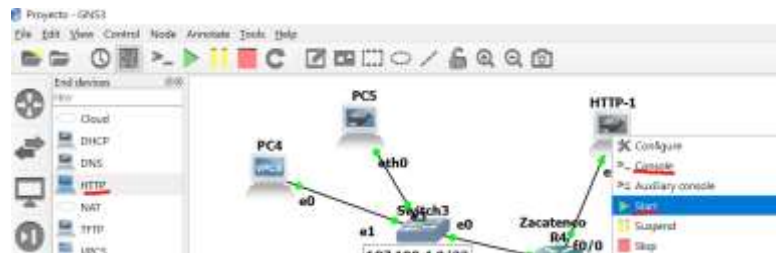
Adicionalmente sin encender damos doble clic y en configuración avanzada, agregamos los directorios que queramos que sean permanentes o persistentes en gns3 ya que se reinician cuando se dejan de usar al estado original de la imagen:



Se agregan los siguientes en este caso:

- /etc
- /sbin
- /var
- /usr

Hacemos click derecho en el dispositivo agregado y damos click en *Start* para iniciar el contenedor. Una vez iniciado, podemos iniciar una consola TTY para poder acceder al dispositivo.



Se agregó un nuevo dispositivo con la imagen de ubuntu para poder hacer pruebas hacia el servidor HTTP. Una vez agregados e iniciados ambos dispositivos, para PC5 se puede pedir una dirección mediante el servidor DHCP. Y el servidor es configurado temporalmente con una ip estática para fines de pruebas. Una vez configuradas las direcciones, es necesario iniciar el servidor apache2 en el servidor HTTP con el siguiente comando.

```

root@HTTP-1:/# systemctl start apache2
root@HTTP-1:/# systemctl status apache2
apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; disabled)
   Active: active (running)
root@HTTP-1:/#

```

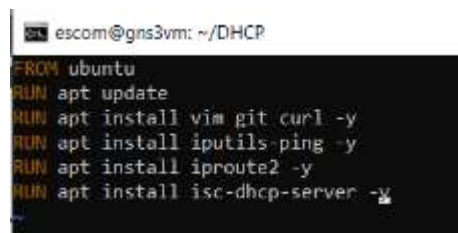
Para verificar que el servidor esté funcionando correctamente, hacemos una solicitud con el comando *curl* a la dirección del servidor HTTP. En la siguiente imagen se muestra que el servicio se encuentra funcionando correctamente y nos devuelve el archivo html por defecto del servidor apache.



```
root@PC3:~# cat Dockerfile
FROM ubuntu
RUN apt update
RUN apt install vim git curl -y
RUN apt install iputils-ping -y
RUN apt install iproute2 -y
RUN apt install isc-dhcp-server -y
CMD httpd -DFOREGROUND
HEALTHCHECK --interval=30s --timeout=30s --start-period=5s --retries=3 \
  curl -f http://localhost:80 || exit 1
```

3.11 Servidor DHCP con isc-dhcp.server

Con un Dockerfile creamos la imagen para el servicio de dhcp al igual que en el servidor HTTP:



```
escom@gns3vm: ~/DHCP
FROM ubuntu
RUN apt update
RUN apt install vim git curl -y
RUN apt install iputils-ping -y
RUN apt install iproute2 -y
RUN apt install isc-dhcp-server -y
```

Figura 29. Dockerfile para el servicio de DHCP.

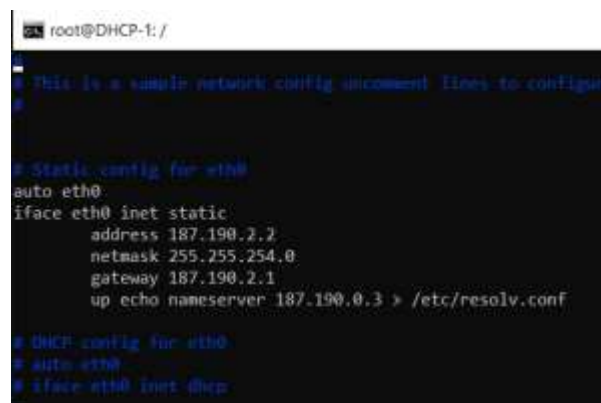
Creamos la imagen con docker build -t dhcp:v1 . y le damos enter



```
escom@gns3vm:~/DHCP$ docker build -t dhcp:v1 .
```

Una vez creada la imagen se agrega el contenedor a gns3 como lo vimos en la sección anterior.

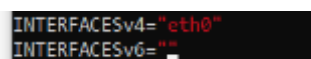
Para configurar el servicio de DCH, ese edita su interfaz para que sea estática. Para esto accedemos al archivo con un editor de texto, por ejemplo, "vi /etc/network/interfaces" y una vez dentro, se debe configurar como se muestra en la siguiente imagen.



```
root@DHCP-1: /
# This is a sample network config uncomment lines to configure
#
# Static config for eth0
auto eth0
iface eth0 inet static
    address 187.190.2.2
    netmask 255.255.254.0
    gateway 187.190.2.1
    up echo nameserver 187.190.0.3 > /etc/resolv.conf
# DHCP config for eth0
# auto-eth0
# iface eth0 inet dhcp
```

Figura 30. Configuración de interfaz de red de DHCP-1.

Se observa que en este caso se pone nameserver 187.190.0.3 ya que será nuestro servidor DNS. Luego editamos el archivo: "vi /etc/default/isc-dhcp-server" para que tenga el nombre de la interfaz.



```
INTERFACESv4="eth0"
INTERFACESv6=""
```

Y se deben de configurar el rango de redes que se le va a asignar a cada subred. En la imagen 31 se muestra la configuración de DHCP para la subred del CECyT 1.

```

subnet 187.190.2.0 netmask 255.255.254.0{
    range 187.190.2.10 187.190.2.20;
    option domain-name-servers 8.8.8.8, 8.8.4.4;
    option domain-name "redes3.net";
    option subnet-mask 255.255.254.0;
    option routers 187.190.2.1;
    option broadcast-address 187.190.3.255;
    default-lease-time 600;
    max-lease-time 7200;
}

```

Figura 31. Dockerfile para el servicio de DHCP.

También es posible asignar ip estáticas. Para agregar IP estáticas se agrega en el mismo apartado de la siguiente forma:

```

subnet 187.190.0.0 netmask 255.255.254.0{
    range 187.190.0.10 187.190.0.20;
    option domain-name-servers 187.190.0.3;
    option domain-name "redes3.net";
    option subnet-mask 255.255.254.0;
    option routers 187.190.0.1;
    option broadcast-address 187.190.1.255;
    default-lease-time 600;
    max-lease-time 7200;

    #IPs estáticas
    host tftp {
        hardware ethernet 5a:5d:03:31:32:f2;
        fixed-address 187.190.0.2;
        option routers 187.190.0.1;
        option broadcast-address 187.190.1.255;
        option domain-name "redes3.net";
        option domain-name-servers 187.190.0.3;
    }
}

```

Figura 32. Configuración de DHCP para asignar direcciones estáticas.

Una vez configuradas todas las subredes, se inicia el servicio DHCP:

```

root@DHCP-1:/# root@DHCP-1:/# service isc-dhcp-server start
Launching IPv4 server only.
 * Starting ISC DHCPv4 server dhcpcd
root@DHCP-1:/# service isc-dhcp-server status
Status of ISC DHCPv4 server: dhcpcd is running.
root@DHCP-1:/#

```

Figura 33. Dockerfile para el servicio de DHCP.

En cada uno de los routers, en las subredes, se agrega: *ip helper-address 187.190.2.2* que servirá para que los routers sepan dónde se encuentra el servidor DHCP.

Desde otra subred, se puede hacer la petición y se asignará la ip vía DHCP:

```

PC2> ip dhcp
DHCPA IP 187.190.0.11/23 GW 187.190.0.1

PC2> sh ip

NAME       : PC2[1]
IP/MASK    : 187.190.0.11/23
GATEWAY    : 187.190.0.1
DNS        : 187.190.0.3
DHCP SERVER: 187.190.2.2
DHCP LEASE : 595, 600/360/525
DOMAIN NAME: redes3.net
MAC        : 08:50:79:56:68:02
I/POR      : 20000
RHOST:PORT : 127.0.0.1:20001
MTU        : 1500

```

Figura 34. Configuración de ip mediante DHCP.

3.12 Servidor DNS con bind9

Al igual que en los anteriores configuramos un Dockerfile:

```

escom@gns3vm: ~/DNS
FROM ubuntu
RUN apt update -y
ENV TZ=Europe/Minsk
RUN ln -snf /usr/share/zoneinfo/$TZ /etc/localtime && echo $TZ > /etc/timezone
RUN apt install vim git curl -y
RUN apt install iputils-ping -y
RUN apt install iproute2 -y
RUN apt install bind9 -y

```

Figura 34. Dockerfile para el servicio de DNS.

Con el comando el siguiente comando creamos la imagen:

```
docker build -t dns:v1 .
```

Una vez creada la imagen la agregamos de la misma forma a gns3, ya en el contenedor, copiamos los siguientes archivos para configurarlos:

```

cp db.127 db.187.190.0.rev
cp db.local db.redes3.net.host

```

Configuramos el archivo `/etc/bind/named.conf.options` de la siguiente forma:

```

root@DNS-1: /etc/bind
options {
    directory "/var/cache/bind";
    forwarders {
        8.8.8.8;
    };
    dnssec-validation no;
    listen-on-v6 { any; };
};

```

Figura 34. Configuración de opciones.

Cabe mencionar que si nuestro servidor DNS no puede resolver el nombre acudira al indicado en este archivo, en este caso será google.com, pero podemos configurar otro servidor maestro el cual pueda resolver otros nombres y dividir la carga. Para esto, configuramos el archivo `/etc/bind/named.conf.local` de la siguiente forma:

```

root@DNS-1: /etc/bind
// Do any local configuration here
//
// Consider adding the 1918 zones here, if they are not used in your
// organization
//include "/etc/bind/zones.rfc1918";
//Zona directa
zone "redes3.net" {
    type master;
    file "/etc/bind/db.redes3.net.host";
};
//Zona Inversa
zone "0.190.187.in-addr.arpa" {
    type master;
    file "/etc/bind/db.187.190.0.rev";
    notify yes;
};

```

Figura 35. Configuración del archivo de configuración local de DNS.

Se configuran las zona directa e inversa, indicando dónde estará el archivo con las configuraciones indicando la ruta completa.

Con el siguiente comando verificamos que lo anterior este bien configurado, no devuelve nada si esta bien, en caso contrario, devolverá un error:

named-checkconf

```
root@DNS-1:/etc/bind# named-checkconf
root@DNS-1:/etc/bind#
```

Configuramos el archivo: `/etc/bind/db.redes3.net.host`, de la siguiente forma con las entradas que se deseen configurar, en este caso se está configurando una entrada para los diferentes servicios disponibles en la topología:

```
root@DNS-1:/etc/bind
$TTL 604800
@ IN SOA redes3.net. root.redes3.net. (
    2          ; Serial
    604800     ; Refresh
    86400      ; Retry
    2419200    ; Expire
    604800     ; Negative Cache TTL
)
;
;
redes3.net. IN NS  redes3.net.
redes3.net. IN A   187.190.0.1
dns.redes3.net. IN A 187.190.0.3
tftp.redes3.net. IN A 187.190.0.2
dhcp.redes3.net. IN A 187.190.2.2
http.redes3.net. IN A 187.190.2.3
snmp.redes3.net. IN A 187.190.4.2
```

Figura 36. Configuración de entradas DNS.

Configuramos el archivo: `/etc/bind/db.187.190.0.rev`, de la siguiente forma:

```
root@DNS-1:/etc/bind
$TTL 604800
@ IN SOA redes3.net. root.redes3.net. (
    1          ; Serial
    604800     ; Refresh
    86400      ; Retry
    2419200    ; Expire
    604800     ; Negative Cache TTL
)
;
;
@ IN NS  redes3.net.
3 IN PTR redes3.net.
3 IN PTR dns.redes3.net.
2 IN PTR tftp.redes3.net.
2 IN PTR dhcp.redes3.net.
3 IN PTR http.redes3.net.
2 IN PTR snmp.redes3.net.
```

Figura 37. Configuración del SOA y PTR's.

Con el siguiente comando revisamos que no tengamos un error en la configuración:

named-checkzone redes3.net /etc/bind/db.redes3.net.host

```
root@DNS-1:/etc/bind# named-checkzone redes3.net /etc/bind/db.redes3.net.host
zone redes3.net/IN: loaded serial 2
OK
root@DNS-1:/etc/bind#
```

Para que resuelva y todos los equipos que van a consultar DNS, esto en todos los equipos que resolverán nombres con este servidor. El archivo `/etc/resolv.conf` a menos que se configure como persistente.

```
root@DNS-1: /etc/bind
nameserver 187.190.0.1
```

Finalmente reiniciamos el servicio y en las siguientes imágenes, se puede observar como las máquinas ahora pueden resolver los nombres y se permite hacer ping.

```
escom@gns3vm:~/DNS docker exec -it 0b588c68482 /bin/bash
root@DNS-1:/# ping smp.redes3.net
PING smp.redes3.net (187.190.4.2) 56(84) bytes of data:
64 bytes from 187.190.4.2: icmp_seq=1 ttl=61 time=40.1 ms
64 bytes from 187.190.4.2: icmp_seq=2 ttl=61 time=36.1 ms
64 bytes from 187.190.4.2: icmp_seq=3 ttl=61 time=42.3 ms
^C
--- smp.redes3.net ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 36.148/39.494/42.233/2.526 ms
root@DNS-1:/# ping des.redes3.net
PING des.redes3.net (187.190.0.3) 56(84) bytes of data:
64 bytes from des.redes3.net (187.190.0.3): icmp_seq=1 ttl=60 time=50.2 ms
64 bytes from des.redes3.net (187.190.0.3): icmp_seq=2 ttl=60 time=41.3 ms
64 bytes from des.redes3.net (187.190.0.3): icmp_seq=3 ttl=60 time=41.3 ms
^C
--- des.redes3.net ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 41.333/44.266/46.600/1.633 ms
root@DNS-1:/# ping smp.redes3.net
PING smp.redes3.net (187.190.4.2) 56(84) bytes of data:
64 bytes from 187.190.4.2: icmp_seq=1 ttl=61 time=47.2 ms
64 bytes from 187.190.4.2: icmp_seq=2 ttl=61 time=47.1 ms
64 bytes from 187.190.4.2: icmp_seq=3 ttl=61 time=34.2 ms
64 bytes from 187.190.4.2: icmp_seq=4 ttl=61 time=41.6 ms
64 bytes from 187.190.4.2: icmp_seq=5 ttl=61 time=48.1 ms
^C
--- smp.redes3.net ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 1305ms
rtt min/avg/max/mdev = 34.239/43.608/48.149/5.267 ms
root@DNS-1:/# ping dhcp.redes3.net
PING dhcp.redes3.net (187.190.2.2) 56(84) bytes of data:
64 bytes from 187.190.2.2: icmp_seq=1 ttl=60 time=46.7 ms
64 bytes from 187.190.2.2: icmp_seq=2 ttl=60 time=52.8 ms
64 bytes from 187.190.2.2: icmp_seq=3 ttl=60 time=49.6 ms
64 bytes from 187.190.2.2: icmp_seq=4 ttl=60 time=56.6 ms
^C
```

Figura 38. Comprobación de resolución de nombres.

3.13 Servidor TFTP con tftpd-hpa

Al igual que en los anteriores servicios con un Dockerfile creamos la imagen y la insertamos en gns3 de la misma forma:

```
escom@gns3vm: ~/TFTP
FROM ubuntu
RUN apt update
RUN apt install vim git curl -y
RUN apt install iputils-ping -y
RUN apt install iproute2 -y
RUN apt install tftpd-hpa -y
RUN apt install python3 -y
RUN apt install netscript-2.4 -y
RUN apt install net-tools -y
RUN apt install systemctl -y
```

Figura 38. Dockerfile para el servicio de TFTP.

De igual forma se crea la imagen

```
buld -t tftp:v1 .
```

Se exporta a gns3.

Las configuraciones para el servidor son las siguientes:

Se crea carpeta y se le da permisos:


```
mkdir /tftp
sudo chmod -R 777 /tftp
sudo chown -R nobody /tftp
```

Se tiene que modificar el archivo `vim /etc/default/tftpd-hpa`.

```
//Se modifica para que quede asi:
TFTP_USERNAME="tftp"
TFTP_DIRECTORY="/tftp"
TFTP_ADDRESS=":69"
TFTP_OPTIONS="--secure --create"
```

Figura 38. Configuración para el servicio tftp.

Para revisar que el servicio esté corriendo, se corren los siguientes comandos: `service tftpd-hpa start`, `service tftpd-hpa status`. En las siguientes imágenes se muestra como se ejecuta el script implementado en Python para hacer los respaldos de las configuraciones de los routers.

[illegible]

Figura 39. Comprobación del servicio TFTP.

3.14 SNMP (Simple Network Management Protocol)

El servicio de SNMP se implementó en dos partes en conjunto. La primera consiste en un programa de Linux llamado SnmpTrapd, este nos permite configurar un contenedor como servidor para recibir las traps provenientes de los diferentes routers de la topología. La segunda parte es un script de Python, el cual tiene las diferentes opciones de SNMP: Get, Get Bulk y Set, y adicionalmente, se añadió una opción para visualizar las traps recientes recibidas mediante el servicio de snmptrapd.

En la figura 40 se puede observar el Dockerfile usado para la configuración de el contenedor SNMP-1 en el cual se implementará este servicio, se puede observar que se configura la zona horaria, se instala *snmptrapd* y se instala la librería de Python *pysnmp*, la cual servirá para implementar el script

```
root@ubuntu:~# apt update
root@ubuntu:~# TZ=America/Mexico_City
root@ubuntu:~# ln -sfn /usr/share/zoneinfo/$TZ /etc/localtime && echo $TZ > /etc/timezone
root@ubuntu:~# apt install vim git curl -y
root@ubuntu:~# apt install iproute2 -y
root@ubuntu:~# apt install net-tools -y
root@ubuntu:~# apt install systemctl -y
root@ubuntu:~# apt install snmp -y
root@ubuntu:~# apt install snmpd -y
root@ubuntu:~# apt install snmptrapd -y
root@ubuntu:~# apt install python3 -y
root@ubuntu:~# apt install python3-pip -y
root@ubuntu:~# pip install pysnmp
```

Figura 40. Dockerfile para el servicio de SNMP.

Una vez tengamos la imagen, agregamos un contenedor con esta y podemos iniciar el servicio de traps. Antes de poder iniciar el servicio de traps, se deben de realizar configuraciones en archivos de

configuración para que se puede acceder a las comunidades y para que se guarden las traps en un archivo de logs. En la figura 41 se muestra la configuración del primer archivo `/etc/default/snmptrapd`, en el cual se tendrá que agregar la línea `TRAPDRUN=yes` y se editará la línea de `TRAPDOTS='...'` a `TRAPDOPTS='-Lf /var/log/traps.log -LSwd -p /run/snmptrapd.pid'`, esto es para que se manden los traps a el archivo de logs `traps.log`.

```
root@SNMP-1: /  
# This file controls the behaviour of /etc/init.d/snmptrapd  
# but not of the corresponding systemd service file.  
# If needed, create an override file in  
# /etc/systemd/system/snmptrapd.service.d/main.conf  
# the man 5 systemd.unit and man 5 systemd.service  
TRAPDRUN=yes  
# snmptrapd options (see Syslog).  
TRAPDOPTS="-Lf /var/log/traps.log -LSwd -p /run/snmptrapd.pid"  
/etc/default/snmptrapd" 8L, 367C written
```

Figura 41. Configuración del archivo `/etc/default/snmptrapd`.

Después de esto, se debe verificar que en el archivo `/etc/snmp/snmptrapd.conf` se encuentre la línea habilitada `authCommunity log,execute,net escom`, donde en lugar de `'escom'` se debe poner el nombre de la comunidad según sea el caso. En la figura 42 se muestra como se debe de ver este archivo.

```
root@SNMP-1: /  
# This file is intended to only be an example.  
# When the trapd agent starts up, this is where it will look for it.  
# All lines beginning with a '#' are ignored, and are commented out.  
# The trapd agent will only look for configuration commands for the agent.  
# Example: trapd agent will only look for the agent.  
authCommunity log,execute,net escom  
# trapd agent will only look for the agent.  
# trapd agent will only look for the agent.  
# trapd agent will only look for the agent.  
# trapd agent will only look for the agent.  
/etc/snmp/snmptrapd.conf" 24L, 103C written
```

Figura 42. Configuración del archivo `/etc/snmp/snmptrapd.conf`.

Ahora, se puede reiniciar el servicio como se muestra en la imagen 43.

```
root@SNMP-1: /# /etc/init.d/snmptrapd restart  
* Restarting SNMP Trap Services snmptrapd  
root@SNMP-1: /# /etc/init.d/snmptrapd status  
* snmptrapd is running  
root@SNMP-1: /#
```

Figura 43. Reinicio del servicio de traps de `SnmpTrapD`.

Ahora, se deben configurar los routers para que manden los traps a este contenedor dentro de la topología, esto se hace con el comando `snmp-server host`, donde se especifica la ip del servidor de traps, la versión de SNMP (en este caso, se usará la versión 2). Además, para poder acceder a la información de los routers y tener las funciones de Get, Get Bulk y Set se debe configurar la comunidad en el router con el comando `snmp-server community nombre`. En la figura 44 se muestran dichas configuraciones para el router Zacatenco.

```
ZACATENCOV2(config)#snmp-server community escom rw  
ZACATENCOV2(config)#snmp-server host 187.190.4.15 version 2c escom  
ZACATENCOV2(config)#snmp-server enable traps
```

Figura 44. Configuración de los routers para habilitar traps y funciones SNMP.

Los comandos de la figura 44 se deben repetir para todos los routers. En este caso, se están habilitando todas las traps en todos los routers, sin embargo, se pueden configurar solamente específicas traps dependiendo la necesidad. Ahora, se procederán a probar las funcionalidades de SNMP implementadas en el script de Python. El script de Python estará referenciado en la sección de referencias.

```

root@SNMP-1:/etc/nms# python3 nms-script.py
BIENVENIDO
Ingresa el nombre de la comunidad MIB: escom
1.- Get
2.- Get Bulk
3.- Editar opción
4.- Ver traps
5.-
Ingresa la ip del agente: 187.190.16.17
Ingresa los OID's deseados separados por comas: 1.3.6.1.2.1.1.5.0
['1.3.6.1.2.1.1.5.0': 'UPIICSA']

```

Figura 45. Ejecución y Get en el script de Python SNMP.

En la figura 45 se muestra la ejecución del script de Python que nos permitirá hacer uso de las funcionalidades de SNMP. El script se puede ejecutar con el comando `python3 nms-script.py` dependiendo de la versión de Python usada. El programa nos solicitará el nombre de la comunidad para leer y escribir, en este caso es `escom`. En la captura 45 se muestra la funcionalidad de GET, en la cual ingresamos la ip del agente a revisar, el oid del parámetro a consultar (se pueden consultar varios parámetros al mismo tiempo, solo hay que ingresarlos oid's separados por comas), en este caso, se ingresa la ip del router de UPIICSA y el oid correspondiente al hostname, y se puede observar que el nombre es el correcto.

```

Ingresa la ip del agente: 187.190.16.17
Ingresa el OID deseado: 1.3.6.1.2.1.1.5.0
Ingresa el valor deseado: ROUTERUPIICSA
Valor editado [['1.3.6.1.2.1.1.5.0': 'UPIICSA']] -> ['ROUTERUPIICSA']
1.- Get
2.- Get Bulk
3.- Editar opción
4.- Ver traps
5.-
Ingresa la ip del agente: 187.190.16.17
Ingresa los OID's deseados separados por comas: 1.3.6.1.2.1.1.5.0
['1.3.6.1.2.1.1.5.0': 'ROUTERUPIICSA']
1.- Get
2.- Get Bulk
3.- Editar opción
4.- Ver traps

```

Password:
ROUTERUPIICSA#

Figura 46. Función SET y GET mediante SNMP.

En la figura 46 se muestra la funcionalidad de SET, en la cual cambiamos el nombre del router `UPIICSA` a `ROUTERUPIICSA` y después de esto, se verifica mediante un GET y al revisar el router mismo.

```

Ingresa la ip del agente: 187.190.16.17
Ingresa los OID's deseados separados por comas: 1.3.6.1.2.1.1.5.0
['1.3.6.1.2.1.1.5.0': 'SANTOTOMAS']
1.- Get
2.- Get Bulk
3.- Editar opción
4.- Ver traps
5.-
Ingresa la ip del agente: 187.190.16.17
Ingresa los OID's deseados separados por comas: 1.3.6.1.2.1.2.2.1.2.1.3.6.1.2.1.2.2.1.3
Ingresa el OID para contar las entradas: 1.3.6.1.2.1.2.1.3
1.3.6.1.2.1.2.2.1.2.3+FastEthernet0/0
1.3.6.1.2.1.2.2.1.3+2
1.3.6.1.2.1.2.2.1.2.3+FastEthernet1/0
1.3.6.1.2.1.2.2.1.3+2
1.3.6.1.2.1.2.2.1.2.3+FastEthernet2/0
1.3.6.1.2.1.2.2.1.3+2
1.3.6.1.2.1.2.2.1.2.3+FastEthernet3/0
1.3.6.1.2.1.2.2.1.3+2
1.3.6.1.2.1.2.2.1.2.3+Serial0/0
1.3.6.1.2.1.2.2.1.3+1
1.3.6.1.2.1.2.2.1.2.3+Serial0/1
1.3.6.1.2.1.2.2.1.3+2
1.3.6.1.2.1.2.2.1.2.3+Serial0/2
1.3.6.1.2.1.2.2.1.3+1
1.3.6.1.2.1.2.2.1.2.3+Serial0/3
1.3.6.1.2.1.2.2.1.3+1

```



Figura 47. Función GET BULK mediante SNMP.

En la figura 47 se muestra la consulta mediante `GET BULK` del agente `SANTOTOMAS`. En este caso, se están consultando dos parámetros: el nombre de las interfaces y el estado de ellas. Y se puede observar que se obtiene el nombre de las interfaces junto con el estado. El número 1, significa que están activas, y el número 2 significa que están no activas. Se puede observar que coincide con las interfaces activas en la topología en la interfaz de GNS3, que son las interfaces: `s4/0`, `s4/2` y `s4/3`, y el resto están inactivas.

```
CECyT1#config
Enter configuration commands, one per line. End with CTF/Z.
CECyT1(config)#int fa0/0
CECyT1(config-if)#shutdown
CECyT1(config-if)#
*Dec 21 03:00:00.501: L1M S-0/0/0/0: Interface FastEthernet0/0, changed state to administratively down
*Dec 21 03:00:00.501: L1M P0/0/0-0-0/0/0: Line protocol on Interface FastEthernet0/0, changed state to down
CECyT1(config-if)#exit
CECyT1(config)#int fa0/0
CECyT1(config-if)#no shutdown
CECyT1(config-if)#

Message received at (2021-12-21 03:18:00) from 187.100.16.14: Serial0/0 administratively down
Message received at (2021-12-21 03:18:00) from 187.100.16.14: Serial module 0
Message received at (2021-12-21 03:17:30) from 187.100.16.18: FastEthernet0/0 administratively down
Message received at (2021-12-21 03:17:05) from 187.100.16.18: L1M S-0/0/0/0: Interface FastEthernet0/0, changed state to up
Message received at (2021-12-21 03:17:06) from 187.100.16.18: FastEthernet0/0 up
1:- int
2:- int fa0/0
3:- int fa0/0
4:- int fa0/0
5:- int fa0/0
6:- int fa0/0
7:- int fa0/0
8:- int fa0/0
9:- int fa0/0
10:- int fa0/0
11:- int fa0/0
12:- int fa0/0
13:- int fa0/0
14:- int fa0/0
15:- int fa0/0
16:- int fa0/0
17:- int fa0/0
18:- int fa0/0
19:- int fa0/0
20:- int fa0/0
21:- int fa0/0
22:- int fa0/0
23:- int fa0/0
24:- int fa0/0
25:- int fa0/0
26:- int fa0/0
27:- int fa0/0
28:- int fa0/0
29:- int fa0/0
30:- int fa0/0
31:- int fa0/0
32:- int fa0/0
33:- int fa0/0
34:- int fa0/0
35:- int fa0/0
36:- int fa0/0
37:- int fa0/0
38:- int fa0/0
39:- int fa0/0
40:- int fa0/0
41:- int fa0/0
42:- int fa0/0
43:- int fa0/0
44:- int fa0/0
45:- int fa0/0
46:- int fa0/0
47:- int fa0/0
48:- int fa0/0
49:- int fa0/0
50:- int fa0/0
51:- int fa0/0
52:- int fa0/0
53:- int fa0/0
54:- int fa0/0
55:- int fa0/0
56:- int fa0/0
57:- int fa0/0
58:- int fa0/0
59:- int fa0/0
60:- int fa0/0
61:- int fa0/0
62:- int fa0/0
63:- int fa0/0
64:- int fa0/0
65:- int fa0/0
66:- int fa0/0
67:- int fa0/0
68:- int fa0/0
69:- int fa0/0
70:- int fa0/0
71:- int fa0/0
72:- int fa0/0
73:- int fa0/0
74:- int fa0/0
75:- int fa0/0
76:- int fa0/0
77:- int fa0/0
78:- int fa0/0
79:- int fa0/0
80:- int fa0/0
81:- int fa0/0
82:- int fa0/0
83:- int fa0/0
84:- int fa0/0
85:- int fa0/0
86:- int fa0/0
87:- int fa0/0
88:- int fa0/0
89:- int fa0/0
90:- int fa0/0
91:- int fa0/0
92:- int fa0/0
93:- int fa0/0
94:- int fa0/0
95:- int fa0/0
96:- int fa0/0
97:- int fa0/0
98:- int fa0/0
99:- int fa0/0
100:- int fa0/0
101:- int fa0/0
102:- int fa0/0
103:- int fa0/0
104:- int fa0/0
105:- int fa0/0
106:- int fa0/0
107:- int fa0/0
108:- int fa0/0
109:- int fa0/0
110:- int fa0/0
111:- int fa0/0
112:- int fa0/0
113:- int fa0/0
114:- int fa0/0
115:- int fa0/0
116:- int fa0/0
117:- int fa0/0
118:- int fa0/0
119:- int fa0/0
120:- int fa0/0
121:- int fa0/0
122:- int fa0/0
123:- int fa0/0
124:- int fa0/0
125:- int fa0/0
126:- int fa0/0
127:- int fa0/0
128:- int fa0/0
129:- int fa0/0
130:- int fa0/0
131:- int fa0/0
132:- int fa0/0
133:- int fa0/0
134:- int fa0/0
135:- int fa0/0
136:- int fa0/0
137:- int fa0/0
138:- int fa0/0
139:- int fa0/0
140:- int fa0/0
141:- int fa0/0
142:- int fa0/0
143:- int fa0/0
144:- int fa0/0
145:- int fa0/0
146:- int fa0/0
147:- int fa0/0
148:- int fa0/0
149:- int fa0/0
150:- int fa0/0
151:- int fa0/0
152:- int fa0/0
153:- int fa0/0
154:- int fa0/0
155:- int fa0/0
156:- int fa0/0
157:- int fa0/0
158:- int fa0/0
159:- int fa0/0
160:- int fa0/0
161:- int fa0/0
162:- int fa0/0
163:- int fa0/0
164:- int fa0/0
165:- int fa0/0
166:- int fa0/0
167:- int fa0/0
168:- int fa0/0
169:- int fa0/0
170:- int fa0/0
171:- int fa0/0
172:- int fa0/0
173:- int fa0/0
174:- int fa0/0
175:- int fa0/0
176:- int fa0/0
177:- int fa0/0
178:- int fa0/0
179:- int fa0/0
180:- int fa0/0
181:- int fa0/0
182:- int fa0/0
183:- int fa0/0
184:- int fa0/0
185:- int fa0/0
186:- int fa0/0
187:- int fa0/0
188:- int fa0/0
189:- int fa0/0
190:- int fa0/0
191:- int fa0/0
192:- int fa0/0
193:- int fa0/0
194:- int fa0/0
195:- int fa0/0
196:- int fa0/0
197:- int fa0/0
198:- int fa0/0
199:- int fa0/0
200:- int fa0/0
201:- int fa0/0
202:- int fa0/0
203:- int fa0/0
204:- int fa0/0
205:- int fa0/0
206:- int fa0/0
207:- int fa0/0
208:- int fa0/0
209:- int fa0/0
210:- int fa0/0
211:- int fa0/0
212:- int fa0/0
213:- int fa0/0
214:- int fa0/0
215:- int fa0/0
216:- int fa0/0
217:- int fa0/0
218:- int fa0/0
219:- int fa0/0
220:- int fa0/0
221:- int fa0/0
222:- int fa0/0
223:- int fa0/0
224:- int fa0/0
225:- int fa0/0
226:- int fa0/0
227:- int fa0/0
228:- int fa0/0
229:- int fa0/0
230:- int fa0/0
231:- int fa0/0
232:- int fa0/0
233:- int fa0/0
234:- int fa0/0
235:- int fa0/0
236:- int fa0/0
237:- int fa0/0
238:- int fa0/0
239:- int fa0/0
240:- int fa0/0
241:- int fa0/0
242:- int fa0/0
243:- int fa0/0
244:- int fa0/0
245:- int fa0/0
246:- int fa0/0
247:- int fa0/0
248:- int fa0/0
249:- int fa0/0
250:- int fa0/0
251:- int fa0/0
252:- int fa0/0
253:- int fa0/0
254:- int fa0/0
255:- int fa0/0
256:- int fa0/0
257:- int fa0/0
258:- int fa0/0
259:- int fa0/0
260:- int fa0/0
261:- int fa0/0
262:- int fa0/0
263:- int fa0/0
264:- int fa0/0
265:- int fa0/0
266:- int fa0/0
267:- int fa0/0
268:- int fa0/0
269:- int fa0/0
270:- int fa0/0
271:- int fa0/0
272:- int fa0/0
273:- int fa0/0
274:- int fa0/0
275:- int fa0/0
276:- int fa0/0
277:- int fa0/0
278:- int fa0/0
279:- int fa0/0
280:- int fa0/0
281:- int fa0/0
282:- int fa0/0
283:- int fa0/0
284:- int fa0/0
285:- int fa0/0
286:- int fa0/0
287:- int fa0/0
288:- int fa0/0
289:- int fa0/0
290:- int fa0/0
291:- int fa0/0
292:- int fa0/0
293:- int fa0/0
294:- int fa0/0
295:- int fa0/0
296:- int fa0/0
297:- int fa0/0
298:- int fa0/0
299:- int fa0/0
300:- int fa0/0
301:- int fa0/0
302:- int fa0/0
303:- int fa0/0
304:- int fa0/0
305:- int fa0/0
306:- int fa0/0
307:- int fa0/0
308:- int fa0/0
309:- int fa0/0
310:- int fa0/0
311:- int fa0/0
312:- int fa0/0
313:- int fa0/0
314:- int fa0/0
315:- int fa0/0
316:- int fa0/0
317:- int fa0/0
318:- int fa0/0
319:- int fa0/0
320:- int fa0/0
321:- int fa0/0
322:- int fa0/0
323:- int fa0/0
324:- int fa0/0
325:- int fa0/0
326:- int fa0/0
327:- int fa0/0
328:- int fa0/0
329:- int fa0/0
330:- int fa0/0
331:- int fa0/0
332:- int fa0/0
333:- int fa0/0
334:- int fa0/0
335:- int fa0/0
336:- int fa0/0
337:- int fa0/0
338:- int fa0/0
339:- int fa0/0
340:- int fa0/0
341:- int fa0/0
342:- int fa0/0
343:- int fa0/0
344:- int fa0/0
345:- int fa0/0
346:- int fa0/0
347:- int fa0/0
348:- int fa0/0
349:- int fa0/0
350:- int fa0/0
351:- int fa0/0
352:- int fa0/0
353:- int fa0/0
354:- int fa0/0
355:- int fa0/0
356:- int fa0/0
357:- int fa0/0
358:- int fa0/0
359:- int fa0/0
360:- int fa0/0
361:- int fa0/0
362:- int fa0/0
363:- int fa0/0
364:- int fa0/0
365:- int fa0/0
366:- int fa0/0
367:- int fa0/0
368:- int fa0/0
369:- int fa0/0
370:- int fa0/0
371:- int fa0/0
372:- int fa0/0
373:- int fa0/0
374:- int fa0/0
375:- int fa0/0
376:- int fa0/0
377:- int fa0/0
378:- int fa0/0
379:- int fa0/0
380:- int fa0/0
381:- int fa0/0
382:- int fa0/0
383:- int fa0/0
384:- int fa0/0
385:- int fa0/0
386:- int fa0/0
387:- int fa0/0
388:- int fa0/0
389:- int fa0/0
390:- int fa0/0
391:- int fa0/0
392:- int fa0/0
393:- int fa0/0
394:- int fa0/0
395:- int fa0/0
396:- int fa0/0
397:- int fa0/0
398:- int fa0/0
399:- int fa0/0
400:- int fa0/0
401:- int fa0/0
402:- int fa0/0
403:- int fa0/0
404:- int fa0/0
405:- int fa0/0
406:- int fa0/0
407:- int fa0/0
408:- int fa0/0
409:- int fa0/0
410:- int fa0/0
411:- int fa0/0
412:- int fa0/0
413:- int fa0/0
414:- int fa0/0
415:- int fa0/0
416:- int fa0/0
417:- int fa0/0
418:- int fa0/0
419:- int fa0/0
420:- int fa0/0
421:- int fa0/0
422:- int fa0/0
423:- int fa0/0
424:- int fa0/0
425:- int fa0/0
426:- int fa0/0
427:- int fa0/0
428:- int fa0/0
429:- int fa0/0
430:- int fa0/0
431:- int fa0/0
432:- int fa0/0
433:- int fa0/0
434:- int fa0/0
435:- int fa0/0
436:- int fa0/0
437:- int fa0/0
438:- int fa0/0
439:- int fa0/0
440:- int fa0/0
441:- int fa0/0
442:- int fa0/0
443:- int fa0/0
444:- int fa0/0
445:- int fa0/0
446:- int fa0/0
447:- int fa0/0
448:- int fa0/0
449:- int fa0/0
450:- int fa0/0
451:- int fa0/0
452:- int fa0/0
453:- int fa0/0
454:- int fa0/0
455:- int fa0/0
456:- int fa0/0
457:- int fa0/0
458:- int fa0/0
459:- int fa0/0
460:- int fa0/0
461:- int fa0/0
462:- int fa0/0
463:- int fa0/0
464:- int fa0/0
465:- int fa0/0
466:- int fa0/0
467:- int fa0/0
468:- int fa0/0
469:- int fa0/0
470:- int fa0/0
471:- int fa0/0
472:- int fa0/0
473:- int fa0/0
474:- int fa0/0
475:- int fa0/0
476:- int fa0/0
477:- int fa0/0
478:- int fa0/0
479:- int fa0/0
480:- int fa0/0
481:- int fa0/0
482:- int fa0/0
483:- int fa0/0
484:- int fa0/0
485:- int fa0/0
486:- int fa0/0
487:- int fa0/0
488:- int fa0/0
489:- int fa0/0
490:- int fa0/0
491:- int fa0/0
492:- int fa0/0
493:- int fa0/0
494:- int fa0/0
495:- int fa0/0
496:- int fa0/0
497:- int fa0/0
498:- int fa0/0
499:- int fa0/0
500:- int fa0/0
501:- int fa0/0
502:- int fa0/0
503:- int fa0/0
504:- int fa0/0
505:- int fa0/0
506:- int fa0/0
507:- int fa0/0
508:- int fa0/0
509:- int fa0/0
510:- int fa0/0
511:- int fa0/0
512:- int fa0/0
513:- int fa0/0
514:- int fa0/0
515:- int fa0/0
516:- int fa0/0
517:- int fa0/0
518:- int fa0/0
519:- int fa0/0
520:- int fa0/0
521:- int fa0/0
522:- int fa0/0
523:- int fa0/0
524:- int fa0/0
525:- int fa0/0
526:- int fa0/0
527:- int fa0/0
528:- int fa0/0
529:- int fa0/0
530:- int fa0/0
531:- int fa0/0
532:- int fa0/0
533:- int fa0/0
534:- int fa0/0
535:- int fa0/0
536:- int fa0/0
537:- int fa0/0
538:- int fa0/0
539:- int fa0/0
540:- int fa0/0
541:- int fa0/0
542:- int fa0/0
543:- int fa0/0
544:- int fa0/0
545:- int fa0/0
546:- int fa0/0
547:- int fa0/0
548:- int fa0/0
549:- int fa0/0
550:- int fa0/0
551:- int fa0/0
552:- int fa0/0
553:- int fa0/0
554:- int fa0/0
555:- int fa0/0
556:- int fa0/0
557:- int fa0/0
558:- int fa0/0
559:- int fa0/0
560:- int fa0/0
561:- int fa0/0
562:- int fa0/0
563:- int fa0/0
564:- int fa0/0
565:- int fa0/0
566:- int fa0/0
567:- int fa0/0
568:- int fa0/0
569:- int fa0/0
570:- int fa0/0
571:- int fa0/0
572:- int fa0/0
573:- int fa0/0
574:- int fa0/0
575:- int fa0/0
576:- int fa0/0
577:- int fa0/0
578:- int fa0/0
579:- int fa0/0
580:- int fa0/0
581:- int fa0/0
582:- int fa0/0
583:- int fa0/0
584:- int fa0/0
585:- int fa0/0
586:- int fa0/0
587:- int fa0/0
588:- int fa0/0
589:- int fa0/0
590:- int fa0/0
591:- int fa0/0
592:- int fa0/0
593:- int fa0/0
594:- int fa0/0
595:- int fa0/0
596:- int fa0/0
597:- int fa0/0
598:- int fa0/0
599:- int fa0/0
600:- int fa0/0
601:- int fa0/0
602:- int fa0/0
603:- int fa0/0
604:- int fa0/0
605:- int fa0/0
606:- int fa0/0
607:- int fa0/0
608:- int fa0/0
609:- int fa0/0
610:- int fa0/0
611:- int fa0/0
612:- int fa0/0
613:- int fa0/0
614:- int fa0/0
615:- int fa0/0
616:- int fa0/0
617:- int fa0/0
618:- int fa0/0
619:- int fa0/0
620:- int fa0/0
621:- int fa0/0
622:- int fa0/0
623:- int fa0/0
624:- int fa0/0
625:- int fa0/0
626:- int fa0/0
627:- int fa0/0
628:- int fa0/0
629:- int fa0/0
630:- int fa0/0
631:- int fa0/0
632:- int fa0/0
633:- int fa0/0
634:- int fa0/0
635:- int fa0/0
636:- int fa0/0
637:- int fa0/0
638:- int fa0/0
639:- int fa0/0
640:- int fa0/0
641:- int fa0/0
642:- int fa0/0
643:- int fa0/0
644:- int fa0/0
645:- int fa0/0
646:- int fa0/0
647:- int fa0/0
648:- int fa0/0
649:- int fa0/0
650:- int fa0/0
651:- int fa0/0
652:- int fa0/0
653:- int fa0/0
654:- int fa0/0
655:- int fa0/0
656:- int fa0/0
657:- int fa0/0
658:- int fa0/0
659:- int fa0/0
660:- int fa0/0
661:- int fa0/0
662:- int fa0/0
663:- int fa0/0
664:- int fa0/0
665:- int fa0/0
666:- int fa0/0
667:- int fa0/0
668:- int fa0/0
669:- int fa0/0
670:- int fa0/0
671:- int fa0/0
672:- int fa0/0
673:- int fa0/0
674:- int fa0/0
675:- int fa0/0
676:- int fa0/0
677:- int fa0/0
678:- int fa0/0
679:- int fa0/0
680:- int fa0/0
681:- int fa0/0
682:- int fa0/0
683:- int fa0/0
684:- int fa0/0
685:- int fa0/0
686:- int fa0/0
687:- int fa0/0
688:- int fa0/0
689:- int fa0/0
690:- int fa0/0
691:- int fa0/0
692:- int fa0/0
693:- int fa0/0
694:- int fa0/0
695:- int fa0/0
696:- int fa0/0
697:- int fa0/0
698:- int fa0/0
699:- int fa0/0
700:- int fa0/0
701:- int fa0/0
702:- int fa0/0
703:- int fa0/0
704:- int fa0/0
705:- int fa0/0
706:- int fa0/0
707:- int fa0/0
708:- int fa0/0
709:- int fa0/0
710:- int fa0/0
711:- int fa0/0
712:- int fa0/0
713:- int fa0/0
714:- int fa0/0
715:- int fa0/0
716:- int fa0/0
717:- int fa0/0
718:- int fa0/0
719:- int fa0/0
720:- int fa0/0
721:- int fa0/0
722:- int fa0/0
723:- int fa0/0
724:- int fa0/0
725:- int fa0/0
726:- int fa0/0
727:- int fa0/0
728:- int fa0/0
729:- int fa0/0
730:- int fa0/0
731:- int fa0/0
732:- int fa0/0
733:- int fa0/0
734:- int fa0/0
735:- int fa0/0
736:- int fa0/0
737:- int fa0/0
738:- int fa0/0
739:- int fa0/0
740:- int fa0/0
741:- int fa0/0
742:- int fa0/0
743:- int fa0/0
744:- int fa0/0
745:- int fa0/0
746:- int fa0/0
747:- int fa0/0
748:- int fa0/0
749:- int fa0/0
750:- int fa0/0
751:- int fa0/0
752:- int fa0/0
753:- int fa0/0
754:- int fa0/0
755:- int fa0/0
756:- int fa0/0
757:- int fa0/0
758:- int fa0/0
759:- int fa0/0
760:- int fa0/0
761:- int fa0/0
762:- int fa0/0
763:- int fa0/0
764:- int fa0/0
765:- int fa0/0
766:- int fa0/0
767:- int fa0/0
768:- int fa0/0
769:- int fa0/0
770:- int fa0/0
771:- int fa0/0
772:- int fa0/0
773:- int fa0/0
774:- int fa0/0
775:- int fa0/0
776:- int fa0/0
777:- int fa0/0
778:- int fa0/0
779:- int fa0/0
780:- int fa0/0
781:- int fa0/0
782:- int fa0/0
783:- int fa0/0
784:- int fa0/0
785:- int fa0/0
786:- int fa0/0
787:- int fa0/0
788:- int fa0/0
789:- int fa0/0
790:- int fa0/0
791:- int fa0/0
792:- int fa0/0
793:- int fa0/0
794:- int fa0/0
795:- int fa0/0
796:- int fa0/0
797:- int fa0/0
798:- int fa0/0
799:- int fa0/0
800:- int fa0/0
801:- int fa0/0
802:- int fa0/0
803:- int fa0/0
804:- int fa0/0
805:- int fa0/0
806:- int fa0/0
807:- int fa0/0
808:- int fa0/0
809:- int fa0/0
810:- int fa0/0
811:- int fa0/0
812:- int fa0/0
813:- int fa0/0
814:- int fa0/0
815:- int fa0/0
816:- int fa0/0
817:- int fa0/0
818:- int fa0/0
819:- int fa0/0
820:- int fa0/0
821:- int fa0/0
822:- int fa0/0
823:- int fa0/0
824:- int fa0/0
825:- int fa0/0
826:- int fa0/0
827:- int fa0/0
828:- int fa0/0
829:- int fa0/0
830:- int fa0/0
831:- int fa0/0
832:- int fa0/0
833:- int fa0/0
834:- int fa0/0
835:- int fa0/0
836:- int fa0/0
837:- int fa0/0
838:- int fa0/0
839:- int fa0/0
840:- int fa0/0
841:- int fa0/0
842:- int fa0/0
843:- int fa0/0
844:- int fa0/0
845:- int fa0/0
846:- int fa0/0
847:- int fa0/0
848:- int fa0/0
849:- int fa0/0
850:- int fa0/0
851:- int fa0/0
852:- int fa0/0
853:- int fa0/0
854:- int fa0/0
855:- int fa0/0
856:- int fa0/0
857:- int fa0/0
858:- int fa0/0
859:- int fa0/0
860:- int fa0/0
861:- int fa0/0
862:- int fa0/0
863:- int fa0/0
864:- int fa0/0
865:- int fa0/0
866:- int fa0/0
867:- int fa0/0
868:- int fa0/0
869:- int fa0/0
870:- int fa0/0
871:- int fa0/0
872:- int fa0/0
873:- int fa0/0
874:- int fa0/0
875:- int fa0/0
876:- int fa0/0
877:- int fa0/0
878:- int fa0/0
879:- int fa0/0
880:- int fa0/0
881:- int fa0/0
882:- int fa0/0
883:- int fa0/0
884:- int fa0/0
885:- int fa0/0
886:- int fa0/0
887:- int fa0/0
888:- int fa0/0
889:- int fa0/0
890:- int fa0/0
891:- int fa0/0
892:- int fa0/0
893:- int fa0/0
894:- int fa0/0
895:- int fa0/0
896:- int fa0/0
897:- int fa0/0
898:- int fa0/0
899:- int fa0/0
900:- int fa0/0
901:- int fa0/0
902:- int fa0/0
903:- int fa0/0
904:- int fa0/0
905:- int fa0/0
906:- int fa0/0
907:- int fa0/0
908:- int fa0/0
909:- int fa0/0
910:- int fa0/0
911:- int fa0/0
912:- int fa0/0
913:- int fa0/0
914:- int fa0/0
915:- int fa0/0
916:- int fa0/0
917:- int fa0/0
918:- int fa0/0
919:- int fa0/0
920:- int fa0/0
921:- int fa0/0
922:- int fa0/0
923:- int fa0/0
924:- int fa0/0
925:- int fa0/0
926:- int fa0/0
927:- int fa0/0
928:- int fa0/0
929:- int fa0/0
930:- int fa0/0
931:- int fa0/0
932:- int fa0/0
933:- int fa0/0
934:- int fa0/0
935:- int fa0/0
936:- int fa0/0
937:- int fa0/0
938:- int fa0/0
939:- int fa0/0
940:- int fa0/0
941:- int fa0/0
942:- int fa0/0
943:- int fa0/0
944:- int fa0/0
945:- int fa0/0
946:- int fa0/0
947:- int fa0/0
948:- int fa0/0
949:- int fa0/0
950:- int fa0/0
951:- int fa0/0
952:- int fa0/0
953:- int fa0/0
954:- int fa0/0
955:- int fa0/0
956:- int fa0/0
957:- int fa0/0
958:- int fa0/0
959:- int fa0/0
960:- int fa0/0
961:- int fa0/0
962:- int fa0/0
963:- int fa0/0
964:- int fa0/0
965:- int fa0/0
966:- int fa0/0
967:- int fa0/0
968:- int fa0/0
969:- int fa0/0
970:- int fa0/0
971:- int fa0/0
972:- int fa0/0
973:- int fa0/0
974:- int fa0/0
975:- int fa0/0
976:- int fa0/0
977:- int fa0/0
978:- int fa0/0
979:- int fa0/0
980:- int fa0/0
981:- int fa0/0
982:- int fa0/0
983:- int fa0/0
984:- int fa0/0
985:- int fa0/0
986:- int fa0/0
987:- int fa0/0
988:- int fa0/0
989:- int fa0/0
990:- int fa0/0
991:- int fa0/0
992:- int fa0/0
993:- int fa0/0
994:- int fa0/0
995:- int fa0/0
996:- int fa0/0
997:- int fa0/0
998:- int fa0/0
999:- int fa0/0
1000:- int fa0/0
```

Figura 48. Visualización de traps.

En la figura 48 se puede observar la funcionalidad 4, que es la de ver traps, en este caso, se emite una trap al desactivar la interfaz Fa0/0 del router CECyT1, y se vuelve a activar casi enseguida. Al seleccionar la opción 4, se podrán visualizar las traps recibidas, desde donde están llegando y cual es el mensaje recibido.

4 Conclusiones

Para la realización del presente proyecto se tuvieron que investigar bastantes temas nuevos, ya que no todos los integrantes del equipo estaban familiarizados con la tecnología de Azure, Docker o GNS3, además de que los temas que teníamos que implementar, aunque ya habían sido vistos en clase, se tuvieron que reforzar consultando los apuntes de clase, lo cual fue bueno para que reforzar y aclarar los temas.

Por otro lado, el crear una máquina virtual en gns3 representa un reto cuando no se está familiarizado con esta plataforma, sin embargo, se considera que es algo bastante útil tanto para este proyecto, como para la vida laboral, ya que te permite familiarizarte con servicios en la nube

Con respecto a Docker, nos topamos con pared completamente, pues aunque servicios similares habían sido usados, como por ejemplo, virtual box, en Docker cambia completamente este paradigma, pues la imagen que se descarga de sus repositorios viene solamente con lo básico, es decir, por defecto está completamente limpia. Lo anterior es una ventaja ya que las imágenes y contenedores son muy ligeros en cuanto a los recursos que utiliza, sin embargo, se tuvieron algunas dificultades para adaptarse a estas imágenes.

Aprender Docker fue un tema bastante beneficioso, ya que es muy útil cuando se sabe utilizar y te facilita bastante las cosas cuando se domina el tema. Además, Docker es una tecnología bastante socorrida en la industria en la actualidad.

Con respecto a gns3, también se tuvieron algunos inconvenientes, en especial con docker y su uso en gns3, pero una vez dominado, resultó bastante bien y fue una buena práctica, aprendimos bastantes cosas, se resolvieron algunas dudas respecto a los temas, y nos ayudó a familiarizarnos con tecnologías muy útiles en un ambiente muy cercano al real.

En general, se considera que fue una buena experiencia, ya que se pusieron en práctica todos los temas vistos en la materia de Administración de Servicios en Red, esto permitió reforzar los temas a la vez de que se aprendía como aplicarlos en situaciones reales, pues GNS3 es un emulador.

Se pudo llegar a la conclusión de que es importante saber como implementar servicios en la vida real, ya sea para la vida laboral, o para proyectos personales. Y se pudo concluir que la administración de estos servicios es fundamental, y el saber como hacer esta administración es indispensable.

5 Referencias

- Jones S. (2020) *¿Qué es Apache web server y como funciona?* Recuperado el 20 de diciembre del 2021. [Online] Disponible en: <https://kinsta.com/es/base-de-conocimiento/que-es-apache/>
- Amazon (2021) *¿Qué es Dokcer?* Recuperado el 20 de diciembre del 2021. [Online] Disponible en: <https://aws.amazon.com/es/docker/>
- Microsoft Azure. (2021) *¿Qué es Azure?* Recuperado el 20 de diciembre del 2021. [Online] Disponible en: <https://azure.microsoft.com/es-mx/overview/what-is-azure/>
- Docker (2021) *Documentación*. [Online] Disponible en: <https://docs.docker.com/>
- Scripts para SNMP en Python: <https://github.com/DonaldoAyala/snmp-nms>