José Cassinelli Andrés Muracciole Israel Vásquez

Documento de proyecto final – SDNV

El objetivo de esta práctica consiste en:

1) Crear y desplegar el escenario virtual VNX de la imagen 1
2) Realizar QoS en las fronteras de la red de acceso de forma que:
    a. H1: 10 Mbps de bajada y 5 Mbps de subida
    b. H2: 4 Mbps de bajada y 4 Mbps de subida
3) Configurar el router vyos con IPv4, NAT y DHCP para IPv4
4) Crear reglas de firewall para:
    a. Filtrar conexiones entrantes excepto http y HTTPS para H1 y SSH para H2
    b. Permitir todas las conexiones salientes
5) Proporcionar acceso a internet.
6) Soporte IPv6 (OPCIONAL)
7) DHCP para IPv6 (OPCIONAL)
8) Vyos WebGUI (OPCIONAL)
9) Crear una segunda red residencial (OPCIONAL)
10) Otros

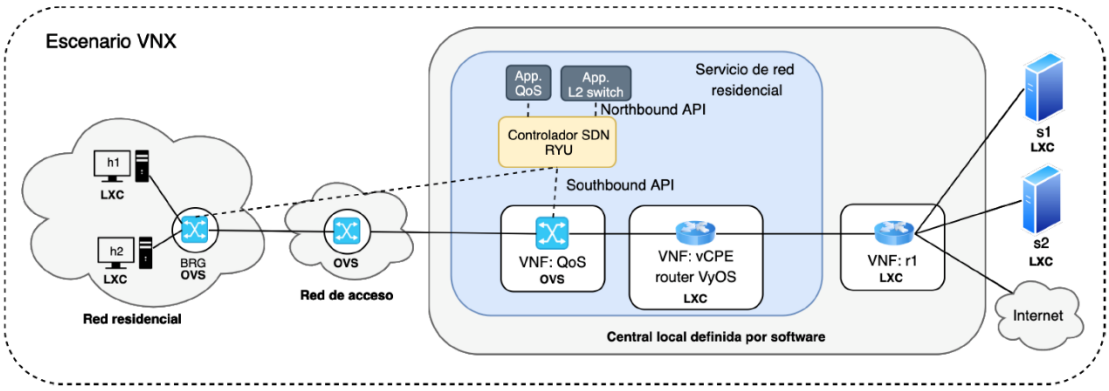

Figura 1 – Diagrama de red

A continuación, se procede a explicar los pasos necesarios para cumplir con lo pedido. A su vez se adjuntará a este documento el archivo .xml que despliega el escenario.

1 – Crear y desplegar el escenario virtual VNX

En esta instancia se desplegaron 2 host (H1 y H2), 2 switches OpenFlow, otro switch, un router vyos, un router Linux, 2 servidores (S1 y S2) y el controlador ryu. De todos los componentes del escenario, los únicos 2 que no tenían direcciones IP fijas eran los hosts ya que estos como se verá más adelante toman IP por DHCP. El resto levantan con IP estática según tabla 1. A su vez, se configuró para que al crear el escenario los hosts levanten un servidor apache2 en el puerto 80, el cual será útil para corroborar el correcto funcionamiento del firewall.

El controlador se encuentra corriendo localmente dentro de la máquina virtual por lo que los conmutadores OpenFlow se conectan a este mediante 127.0.0.1:6633.

La red de host es la 10.0.1.0/24, entre routers es la 10.1.1.0/24 y la red hacia los servidores es la 10.2.1.0/24.

NOMBRE	INTERFAZ	MAC	IPv4
H1	Eth1	00:00:00:00:01:01	10.0.1.7 (dhcp)
H2	Eth1	00:00:00:00:02:01	10.0.1.8 (dhcp)
NET9 (BRG)	-	00:00:00:00:09:00	N/C
NET8 (OVS)	-	00:00:00:00:08:00	N/C
NET7 (VNF)	-	00:00:00:00:07:00	N/C
r_vyos	Eth1	00:00:00:00:03:01	10.0.1.1
r_vyos	Eth2	00:00:00:00:03:02	10.1.1.1
r1	Eth1	00:00:00:00:04:01	10.2.1.1
r1	Eth2	00:00:00:00:04:02	10.1.1.2
r1	Eth9	00:00:00:00:04:03	dhcp
S1	Eth1	00:00:00:00:05:01	10.2.1.2
S2	Eth1	00:00:00:00:06:01	10.2.1.3

Tabla 1 – Direcciones capa 2 y 3 de los componentes de red

También se configuraron direcciones IPv6 en el xml pero la explicación de esto se dejará para el punto 6 y 7 del documento.
Por último, cabe destacar que al router “r1” se le conectó una red llamada “virbr0” que es la que conecta hacia fuera y le da conectividad a internet. Para que esto funcione fue necesario realizar un NAT entre las interfaces internas 1 y 2 y la 9 (externa).

```
<filetree seq="on_boot" root="/usr/bin/" perms="755">/usr/bin/vnx_config_nat</filetree>
<exec seq="on_boot" type="verbatim">
    /usr/bin/vnx_config_nat eth1 eth9
    /usr/bin/vnx_config_nat eth2 eth9
</exec>
```

Figura 2 – NAT para que los hosts, routers y servidores salgan hacia internet

2 – QoS

En este punto de la práctica se procedió a trabajar con el controlador ryu de forma de aplicar políticas de calidad de servicio para subida y bajada hacia los hosts en la frontera de la red de acceso. Esto se hizo utilizando la API de QoS de ryu.
Al no estar explicado en la letra si la finalidad de la misma es garantizar un ancho de banda mínimo o limitarlo, lo que se hizo fue definir que estos valores planteados sean los máximos (límite) ya que es la forma que se encontró de demostrar en la práctica que esté aplicando el QoS. Como el tráfico visto a partir del iperf antes de aplicar las políticas era superior al que se pedía en el objetivo, en caso de que estos valores sean los mínimos, no se vería reflejado esto en la práctica ya que al correr el comando iperf el resultado sería el mismo.

Dicho esto, se procedió a configurar los perfiles y reglas de calidad de servicio en el controlador. Para ello primero se levantaron las API´s correspondientes y luego mediante los siguientes comandos se crearon las reglas:

```
curl -X PUT -d '"tcp:127.0.0.1:6632"'
http://localhost:8080/v1.0/conf/switches/0000000000000900/ovsdb_addr
```

```
curl -X PUT -d '"tcp:127.0.0.1:6632"'
http://localhost:8080/v1.0/conf/switches/0000000000000700/ovsdb_addr
```

```
curl -X PUT -d '"tcp:127.0.0.1:6632"'
http://localhost:8080/v1.0/conf/switches/0000000000000a00/ovsdb_addr

curl -X PUT -d '"tcp:127.0.0.1:6632"'
http://localhost:8080/v1.0/conf/switches/0000000000000b00/ovsdb_addr

curl -X POST -d '{"port_name": "net8-net9-1", "type": "linux-htb", "queues": [{"max_rate": "5000000"},
{"max_rate": "4000000"}]}' http://localhost:8080/qos/queue/0000000000000900

curl -X POST -d '{"port_name": "net8-net7-1", "type": "linux-htb", "queues": [{"max_rate": "10000000"},
{"max_rate": "4000000"}]}' http://localhost:8080/qos/queue/0000000000000700

curl -X POST -d '{"match": {"nw_src": "10.0.1.7", "nw_proto": "TCP"}, "actions":{"queue": "0"}}'
http://localhost:8080/qos/rules/0000000000000900

curl -X POST -d '{"match": {"nw_src": "10.0.1.8", "nw_proto": "TCP"}, "actions":{"queue": "1"}}'
http://localhost:8080/qos/rules/0000000000000900

curl -X POST -d '{"match": {"nw_dst": "10.0.1.7", "nw_proto": "TCP"}, "actions":{"queue": "0"}}'
http://localhost:8080/qos/rules/0000000000000700

curl -X POST -d '{"match": {"nw_dst": "10.0.1.8", "nw_proto": "TCP"}, "actions":{"queue": "1"}}'
http://localhost:8080/qos/rules/0000000000000700
```

Las primeras 2 sentencias definen el ovsdb_addr, las siguientes 2 estipulan los perfiles y las ultimas 4 se aplican a los conmutadores correspondientes en las interfaces deseadas y se le indica el tipo de tráfico y el origen o destino según corresponda.

A modo de ejemplo se adjuntan 4 capturas de pantalla donde en las primeras 2 (figuras 3 y 4) se ve que el ancho de banda utilizado antes de aplicar la calidad de servicio es en el entorno de los Gbps y en la siguientes (figuras 5 y 6) ya con QoS para el caso de un ping de H1 a S1.

```
root@h1:~# iperf -c 10.2.1.2 -i 1
------------------------------------------------------------
Client connecting to 10.2.1.2, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 10.0.1.7 port 57188 connected with 10.2.1.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  1.90 GBytes  16.4 Gbits/sec
[  3]  1.0- 2.0 sec  2.13 GBytes  18.3 Gbits/sec
[  3]  2.0- 3.0 sec  2.16 GBytes  18.5 Gbits/sec
[  3]  3.0- 4.0 sec  2.18 GBytes  18.7 Gbits/sec
[  3]  4.0- 5.0 sec  2.00 GBytes  17.2 Gbits/sec
[  3]  5.0- 6.0 sec  2.14 GBytes  18.4 Gbits/sec
[  3]  6.0- 7.0 sec  1.60 GBytes  13.8 Gbits/sec
[  3]  7.0- 8.0 sec  1.65 GBytes  14.2 Gbits/sec
[  3]  8.0- 9.0 sec  1.46 GBytes  12.6 Gbits/sec
[  3]  9.0-10.0 sec  1.07 GBytes  9.22 Gbits/sec
[  3]  0.0-10.0 sec  18.3 GBytes  15.7 Gbits/sec
```

Figura 3 – iperf en H1 con un ping de salida hacia S1 antes de aplicar QoS

```
root@S1:~# iperf -s -i 1
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 10.2.1.2 port 5001 connected with 10.1.1.1 port 57188
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec  1.90 GBytes  16.4 Gbits/sec
[  4]  1.0- 2.0 sec  2.13 GBytes  18.3 Gbits/sec
[  4]  2.0- 3.0 sec  2.16 GBytes  18.5 Gbits/sec
[  4]  3.0- 4.0 sec  2.18 GBytes  18.7 Gbits/sec
[  4]  4.0- 5.0 sec  2.00 GBytes  17.2 Gbits/sec
[  4]  5.0- 6.0 sec  2.14 GBytes  18.4 Gbits/sec
[  4]  6.0- 7.0 sec  1.60 GBytes  13.8 Gbits/sec
[  4]  7.0- 8.0 sec  1.65 GBytes  14.2 Gbits/sec
[  4]  8.0- 9.0 sec  1.46 GBytes  12.6 Gbits/sec
[  4]  9.0-10.0 sec  1.07 GBytes  9.22 Gbits/sec
[  4]  0.0-10.0 sec  18.3 GBytes  15.7 Gbits/sec
```

Figura 4 – iperf en S1 con un ping de llegada desde H1 antes de aplicar QoS

```
root@h1:~# iperf -c 10.2.1.2 -i 1
------------------------------------------------------------
Client connecting to 10.2.1.2, TCP port 5001
TCP window size: 85.0 KByte (default)
------------------------------------------------------------
[  3] local 10.0.1.7 port 33050 connected with 10.2.1.2 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0- 1.0 sec  1.37 MBytes  11.5 Mbits/sec
[  3]  1.0- 2.0 sec   636 KBytes  5.21 Mbits/sec
[  3]  2.0- 3.0 sec   764 KBytes  6.26 Mbits/sec
[  3]  3.0- 4.0 sec   382 KBytes  3.13 Mbits/sec
[  3]  4.0- 5.0 sec   636 KBytes  5.21 Mbits/sec
[  3]  5.0- 6.0 sec   764 KBytes  6.26 Mbits/sec
[  3]  6.0- 7.0 sec   382 KBytes  3.13 Mbits/sec
[  3]  7.0- 8.0 sec   764 KBytes  6.26 Mbits/sec
[  3]  8.0- 9.0 sec   954 KBytes  7.82 Mbits/sec
[  3]  9.0-10.0 sec  1.06 MBytes  8.86 Mbits/sec
[  3]  0.0-10.1 sec  7.59 MBytes  6.31 Mbits/sec
```

Figura 5 – iperf en H1 con un ping de salida hacia H1 después de aplicar QoS

```
root@S1:~# iperf -s -i 1
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[  4] local 10.2.1.2 port 5001 connected with 10.1.1.1 port 33050
[ ID] Interval       Transfer     Bandwidth
[  4]  0.0- 1.0 sec   542 KBytes  4.44 Mbits/sec
[  4]  1.0- 2.0 sec   477 KBytes  3.90 Mbits/sec
[  4]  2.0- 3.0 sec   553 KBytes  4.53 Mbits/sec
[  4]  3.0- 4.0 sec   263 KBytes  2.15 Mbits/sec
[  4]  4.0- 5.0 sec   501 KBytes  4.10 Mbits/sec
[  4]  5.0- 6.0 sec   584 KBytes  4.78 Mbits/sec
[  4]  6.0- 7.0 sec   581 KBytes  4.76 Mbits/sec
[  4]  7.0- 8.0 sec   578 KBytes  4.74 Mbits/sec
[  4]  8.0- 9.0 sec   580 KBytes  4.75 Mbits/sec
[  4]  9.0-10.0 sec   526 KBytes  4.31 Mbits/sec
[  4] 10.0-11.0 sec   576 KBytes  4.71 Mbits/sec
[  4] 11.0-12.0 sec   578 KBytes  4.74 Mbits/sec
[  4] 12.0-13.0 sec   578 KBytes  4.74 Mbits/sec
[  4] 13.0-14.0 sec   578 KBytes  4.74 Mbits/sec
[  4]  0.0-14.5 sec  7.59 MBytes  4.40 Mbits/sec
```

Figura 6 – iperf en S1 con un ping de llegada desde H1 después de aplicar QoS

La política de calidad de servicio hace referencia a que el máximo downlink posible para H1 es de 10 Mbps y 5 Mbps de downlink. En caso de hacerlo con H2 el rango será 4Mbps-4Mbps respectivamente. Se puede apreciar que nunca el tráfico supera los 10Mbps o los 5 Mbps luego de aplicar QoS por lo cual se corrobora su correcto funcionamiento.

3 – IPv4, NAT y DHCPv4

Como se mencionó anteriormente, todas las interfaces de red y los servidores poseen direcciones IP estáticas ya que en la práctica es también lo que se suele hacer, dejando los hosts de la red residencial tomar IP mediante DHCP. Para ello fue necesario configurar la etiqueta <ipv4> en el xml para que aprenda direcciones por dhcp y a su vez en el router vyos configurar el pool de IP´s a entregar. Para ello se propuso utilizar la red 10.0.1.0/24 donde la .1 la tenía el router vyos y el pool comenzara a entregar desde la .10 a la .255, de forma de dejar un rango reservado de direcciones para equipos que a futuro puedan precisar IP estática (buenas prácticas).
Si bien con esta configuración ya bastara, se decidió que los hosts tomen direcciones fijas por DHCP ya que luego será necesario para configurar las reglas del firewall. Por tal motivo se configuró el router vyos para que a partir de la dirección MAC del host, se le asigne una dirección IP específica. Hay que mencionar que todas las interfaces de los equipos que conforman el escenario poseen una dirección MAC configurada de antemano en el xml. Con esto se asignó la dirección 10.0.1.7 a H1 y 10.0.1.8 a H2.

En lo que refiere al ruteo, se eligió realizar ruteo estático ya que era lo más simple, aunque se podría haber configurado algún protocolo de ruteo sin problema.

Por último, en relación al NAT, este se configuró como PAT, de forma que todos los mensajes “hacia fuera” salen con la dirección de origen la de la interfaz de salida del router vyos. A continuación, se muestra mediante el uso de tcpdump captura del ping de H1 a S1 antes y después de aplicar el NAT. Como se puede ver sin configurar el NAT la dirección de origen es la de H1 (10.0.1.7) y luego de aplicar el NAT, esta es 10.1.1.1.

```
root@S1:~# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
22:25:00.179806 ARP, Request who-has S1 tell _gateway, length 28
22:25:00.179829 ARP, Reply S1 is-at 00:00:00:00:05:01 (oui Ethernet), length 28
22:25:00.179834 IP 10.0.1.7 > S1: ICMP echo request, id 561, seq 1, length 64
22:25:00.179844 IP S1 > 10.0.1.7: ICMP echo reply, id 561, seq 1, length 64
22:25:01.181765 IP 10.0.1.7 > S1: ICMP echo request, id 561, seq 2, length 64
22:25:01.181786 IP S1 > 10.0.1.7: ICMP echo reply, id 561, seq 2, length 64
22:25:02.190344 IP 10.0.1.7 > S1: ICMP echo request, id 561, seq 3, length 64
```

Figura 7 – tcpdump en S1 con ping desde H1 sin NAT en router vyos

```
root@S1:~# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
22:26:11.400553 IP 10.1.1.1 > S1: ICMP echo request, id 563, seq 1, length 64
22:26:11.400570 IP S1 > 10.1.1.1: ICMP echo reply, id 563, seq 1, length 64
22:26:12.430288 IP 10.1.1.1 > S1: ICMP echo request, id 563, seq 2, length 64
22:26:12.430324 IP S1 > 10.1.1.1: ICMP echo reply, id 563, seq 2, length 64
22:26:13.455966 IP 10.1.1.1 > S1: ICMP echo request, id 563, seq 3, length 64
```

Figura 8 – tcpdump en S1 con ping desde H1 con NAT en router vyos

4 – Reglas de firewall

En esta sección se configuraron reglas de firewall en el router vyos para permitir únicamente el tráfico HTTP y HTTPS entrante a H1 y SSH entrante a H2. Todo el resto del tráfico IPv4 entrante es negado por defecto.
Por otro lado, el trafico saliente tanto hacia internet como hacia la red de servidores es permitido. Los comandos necesarios para realizar esto se pueden ver en el anexo 2 de este documento.

Como se mencionó anteriormente, fue necesario configurar el servidor dhcp para que los hosts tomen determinada IP ya que son a estas a las que luego se les aplica la regla de filtrado. Otras soluciones para atacar este problema podrían haber sido mediante la creación de subredes o vlans, pero se optó por esta ya que creemos que es la más escalable y no hay desperdicio o mal uso de direcciones IP.

La forma de corroborar que se están aplicando las reglas es intentar hacer un telnet a los hosts con el puerto 80 o 433 y ver que para H1 acepta conexión, pero no para H2, o intentar conectarse vía SSH a ambos hosts y comprobar que para H2 es posible acceder, pero no así para H1. Por otra parte, el tráfico saliente haciendo un ping a los servidores o hacia internet debería salir sin problemas.

5 – Proporcionar acceso a internet

El router de salida es el r1, por lo que a este es donde se conecta la interfaz eth9 a la red “virbr0” la cual es la salida a internet. Esta toma dirección por dhcp de la red 192.168.122.0/24 y mediante un NAT permite que los equipos puedan acceder hacia fuera del escenario. El NAT para este caso se muestra en la figura 2.

6 – Soporte Ipv6 (OPCIONAL)

Por soporte IPv6 se entiende que los equipos del escenario puedan comunicarse entre ellos mediante protocolo de red IPv6. Para ello se procedió a realizar algo parecido a lo hecho para IPv4, de forma que todos los equipos posean una dirección IPv6 fija excepto los hosts como se indica en la siguiente tabla.

NOMBRE	INTERFAZ	IPv6
H1	Eth1	2001:db8::200:ff:fe00:101(dhcp)
H2	Eth1	2001:db8::200:ff:fe00:201(dhcp)
NET9 (BRG)	-	N/C
NET8 (OVS)	-	N/C
NET7 (VNF)	-	N/C
r_vyos	Eth1	2001:db8::200:ff:fe00:301
r_vyos	Eth2	2001:db8:0:1:200:ff:fe00:302
r1	Eth1	2001:db8:0:2:200:ff:fe00:401
r1	Eth2	2001:db8:0:1:200:ff:fe00:402
r1	Eth9	N/C
S1	Eth1	2001:db8:0:2:200:ff:fe00:501
S2	Eth1	2001:db8:0:2:200:ff:fe00:601

Tabla 2 – Direcciones IPv6 de los componentes de red del escenario principal

Una vez realizada la configuración pertinente en el xml tanto para la asignación de direcciones como para el ruteo (estático), se comprueba la conectividad extremo a extremo por ping.

7 – DHCP Ipv6 (OPCIONAL)

Al igual que para el caso de IPv4, se vio que era necesario que los hosts si bien tomen dirección IPv6 mediante DHCP, esta debería ser fija. En una primera instancia se intentó realizarlo mediante la función de static-mapping que proporciona vyos, pero creemos que esta sección de configuración tiene problemas ya que al poner el identificador (últimos 4 bloques de la dirección MAC según lo que indica la wiki de vyos), este tiraba abajo el servidor DHCPv6. Por lo tanto, la forma de asignación de IP que se decidió llevar a cabo fue a partir de IPv6 stateless, también conocido como autoconfiguración con SLAAC, donde los primeros 64 bits de la dirección surgen del prefijo y los siguientes 64 a partir de la dirección MAC-EUI-64.

8 – Vyos WebGUI (OPCIONAL)

Para levantar la interfaz web y acceder a ella es necesario tener acceso desde la máquina virtual a r_vyos, de tal modo que se pueda ingresar a la interfaz web desde el explorador ingresando la IP de la interfaz ethernet del router en la que se encuentra la máquina virtual.

Una vez logrado esto los procedimientos para levantar la interfaz web en r_vyos fue:

1- Ingresar a la línea de comandos de VyOS.
2- configure
3- set service https (Con esto levantamos el servicio SSL y obligamos también a levantar el servidor web apache de r_vyos)
4- commit
5- save

```
vyos@vyos:~$ configure
[edit]
vyos@vyos# set service https
[edit]
vyos@vyos# commit
[ service https ]
Generating a 2048 bit RSA private key
.....................................................................................
..................+++
.........+++
writing new private key to '/etc/lighttpd/server.pem'
-----

[ service https ]
Stopping web server: lighttpd.
Starting web server: lighttpd.
Stopping PAGER server
save
Starting PAGER server

[edit]
vyos@vyos# save
Warning: you have uncommitted changes that will not be saved.

Saving configuration to '/config/config.boot'...
```

Figura 9 – Pasos para ingresar a la interfaz GUI

Con esto hemos logrado levantar el servicio apache. Ahora ingresamos por web y tenemos lo siguiente:

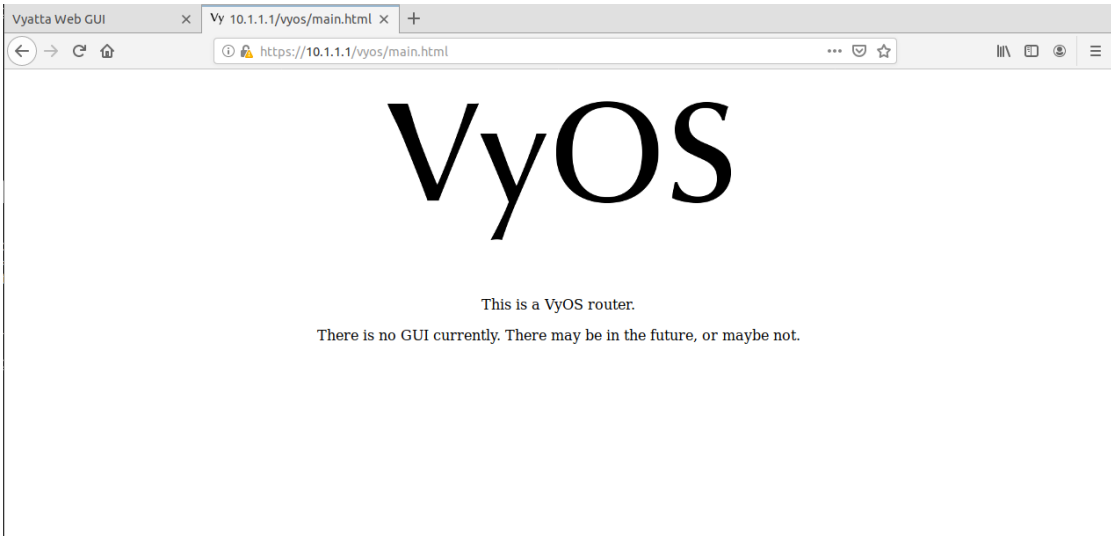

Figura 10 – Interfaz Web VyOS

Lo que nos indica que el servicio ha levantado correctamente pero no tenemos acceso a la configuración, se procedió a descargar los ficheros de internet de la interfaz gráfica de VyOS pero bajo el nombre de vyatta web gui, ya que al parecer no hay para VyOS y se procede a copiar los archivos dentro de la carpeta /var/www/ del r_vyos.
Volvemos a ingresar a la interfaz web y obtenemos:

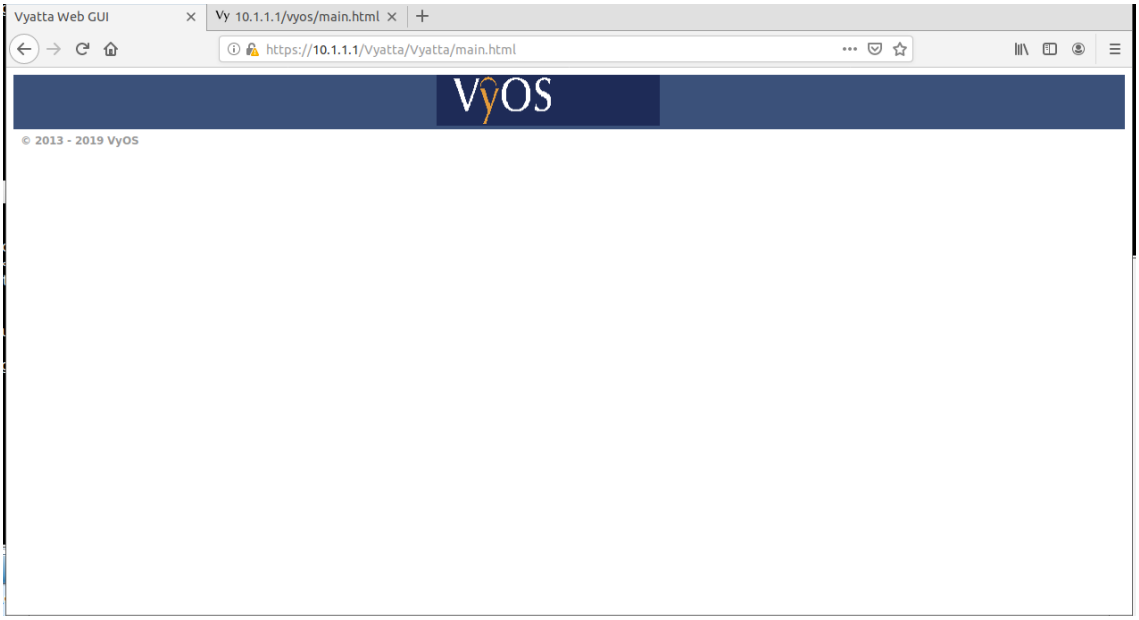

Figura 11 – Interfaz Web VyOS (vyatta)

Lo que nos quiere decir que se cargaron los archivos con éxito, pero de alguna manera la aplicación web de vyos no conecta con los descargados de vyatta y no se puede configurar el router a través de la interfaz web.
Para esto sería recomendable buscar los archivos de VyOS

9 – Crear una segunda red residencial (OPCIONAL)

Se examinó la práctica de OSM, en donde creamos un escenario parecido. Siguiendo los conceptos vistos en clase y con ayuda de consultas a los profesores de la asignatura, se desplegó un escenario con dos redes residenciales (h1 y h2 una red y h3, h4 otra red).

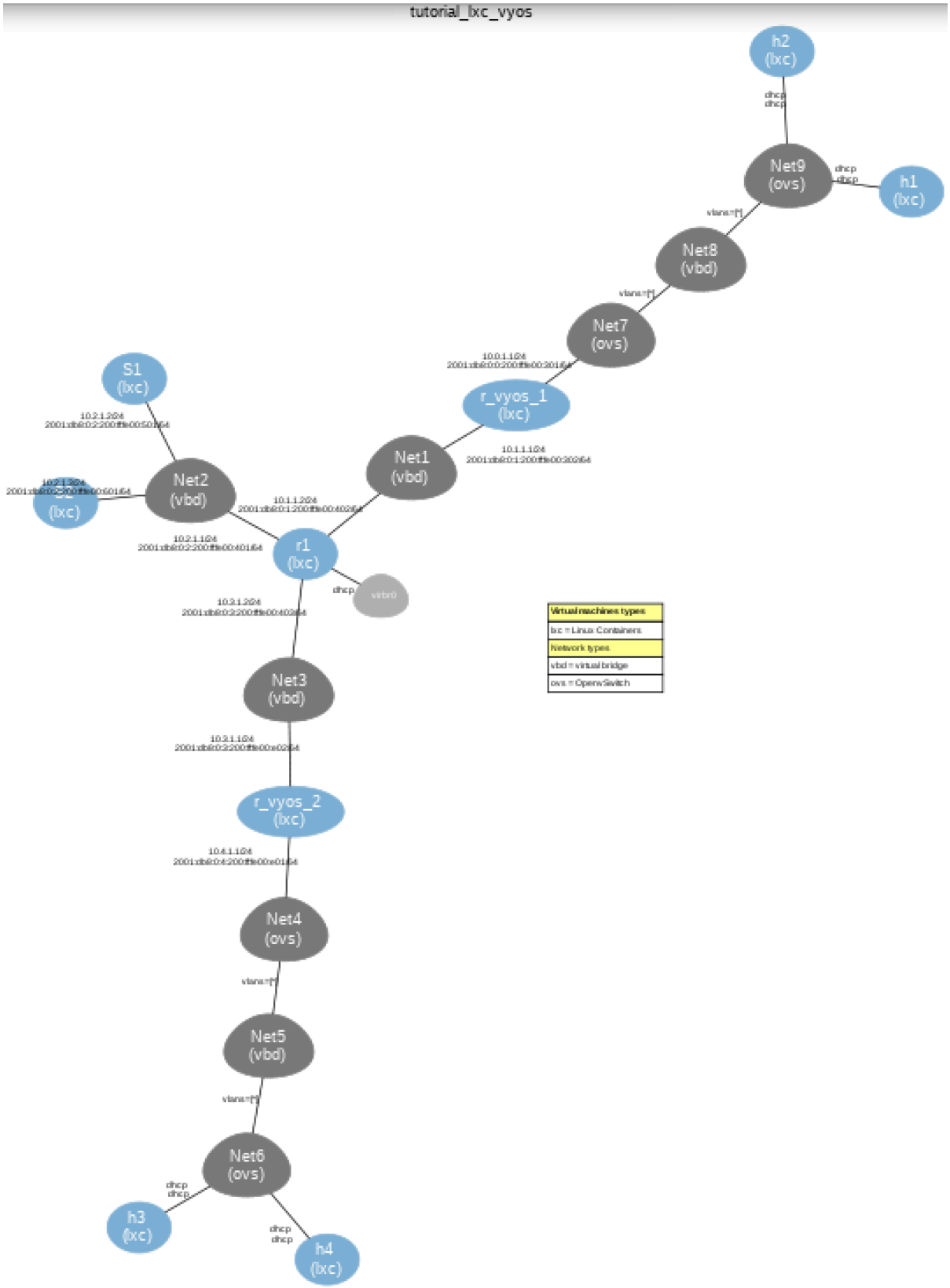

Figura 12 – Escenario con 2 redes residenciales

La segunda red residencial empieza desde la subnet del router de salida (en este caso nuestra Net 3). Se probó conectividad entre los hosts y los Servers (s1 y s2) y todo fue bien. Como se trata de dos redes residenciales diferentes, no deben estar encaminadas entre sí, lo cual se puede ver haciendo un traceroute entre redes. Por otra parte como son redes residenciales perfectamente podrían tener el mismo rango de IP´s privadas y no generar conflicto entre ellas.

Si se detiene o se cae el servicio del controlador, ambas redes residenciales no tendrán salida a internet, que por la teoría es correcto. Solo tendrán salida a internet los servidores porque no están dentro de la central local. Para que haya una buena práctica y, como ya fue explicado anteriormente, hay que configurar los routers vyos, uno por cada red residencial.
Para ello se recomienda una buena práctica que es la automatización de ejecución de comandos en el r_VyOs. Para ello se deben seguir los siguientes pasos:

a. Crear un fichero en el directorio conf/set-hostname (puede ser cualquier nombre).
b. Dentro del fichero se debe incluir una cabecera:
    i. #!/bin/vbash
    ii. source /opt/vyatta/etc/functions/script-template
c. Después de añadir la cabecera, se incluye todo el contenido de configuración que se deseé automatizar (incluir toda la configuración después de b-ii).
d. Para que se copie la configuración del fichero al router vyos, se debe incluir el siguiente código.
    i. <filetree seq="set-hostname" root="/root/" perms="755">conf/set-hostname</filetree>
    ii. <exec seq="set-hostname" type="verbatim" ostype="system"> /root/set-hostname
    iii. </exec>
e. Por último, se ejecuta el siguiente código en el directorio donde se encuentra el xml.
    i. vnx -f <nombre_del_fichero_xml> -x set-hostname

En la siguiente imagen se podrá ver el resultado de esta ejecución.

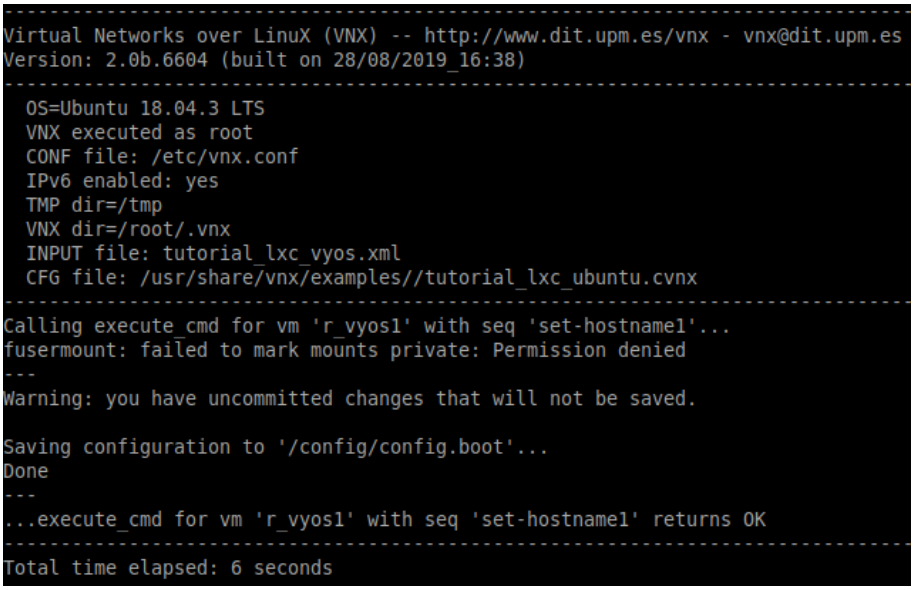

Figura 13 – Resultado de automatización de configuración de los VyOS

Con este nuevo escenario más extenso, se pasa a detallar la tabla de IPv4, IPv6 y MAC para cada caso:

NOMBRE	INTERFAZ	MAC	IPv4	IPv6
H1	Eth1	00:00:00:00:01:01	10.0.1.7 (dhcp)	2001:db8::200:ff:fe00:101
H2	Eth1	00:00:00:00:02:01	10.0.1.8 (dhcp)	2001:db8::200:ff:fe00:201
H3	Eth1	00:00:00:00:0C:01	10.4.1.7 (dhcp)	2001:db8:0:4:200:ff:fe00:c01
H4	Eth1	00:00:00:00:0D:01	10.4.1.8 (dhcp)	2001:db8:0:4:200:ff:fe00:d01
NET9 (BRG)	-	00:00:00:00:09:00	N/C	N/C
NET8 (OVS)	-	00:00:00:00:08:00	N/C	N/C
NET7 (VNF)	-	00:00:00:00:07:00	N/C	N/C
r_vyos_1	Eth1	00:00:00:00:03:01	10.0.1.1	2001:db8::200:ff:fe00:301
r_vyos_1	Eth2	00:00:00:00:03:02	10.1.1.1	2001:db8:0:1:200:ff:fe00:302
r_vyos_2	Eth1	00:00:00:00:0E:01	10.4.1.1	2001:db8:0:4:200:ff:fe00:e01
r_vyos_2	Eth2	00:00:00:00:0E:02	10.3.1.1	2001:db8:0:3:200:ff:fe00:e02
r1	Eth1	00:00:00:00:04:01	10.2.1.1	2001:db8:0:2:200:ff:fe00:401
r1	Eth2	00:00:00:00:04:02	10.1.1.2	2001:db8:0:1:200:ff:fe00:402
r1	Eth3	00:00:00:00:04:03	10.3.1.2	2001:db8:0:3:200:ff:fe00:403
r1	Eth9	00:00:00:00:04:04	dhcp	-
S1	Eth1	00:00:00:00:05:01	10.2.1.2	2001:db8:0:2:200:ff:fe00:501
S2	Eth1	00:00:00:00:06:01	10.2.1.3	2001:db8:0:2:200:ff:fe00:601

Tabla 3 – Direcciones IPv4, IPv6 y MAC de los componentes de red del escenario extendido

10 – Otros (OPCIONAL)

Para seguir profundizando con la práctica se procedió a realizar las reglas de filtrado de firewall de tráfico IPv4 mencionadas en el punto 4, pero ahora para IPv6. La configuración de la misma se ve reflejad en el anexo 2 y 3. De esta forma se protege la red doméstica tanto para IPv4, como para IPv6, reduciendo así las vulnerabilidades.

Por otro lado, se buscó realizar también la configuración de NAT para direcciones IPv6, pero esto no se pudo hacer ya que la versión de vyos que se utilizó en este trabajo fue la 1.1.8 y recién en la 3.13 permite realizar NPTv6.

Por último, se buscó la forma de levantar una interfaz web del controlador ryu donde se muestra los conmutadores conectados al controlador como se ve en la siguiente imagen. Esta interfaz se levanta en el puerto 8080 del host y la forma de desplegarla se indica en el anexo 1.

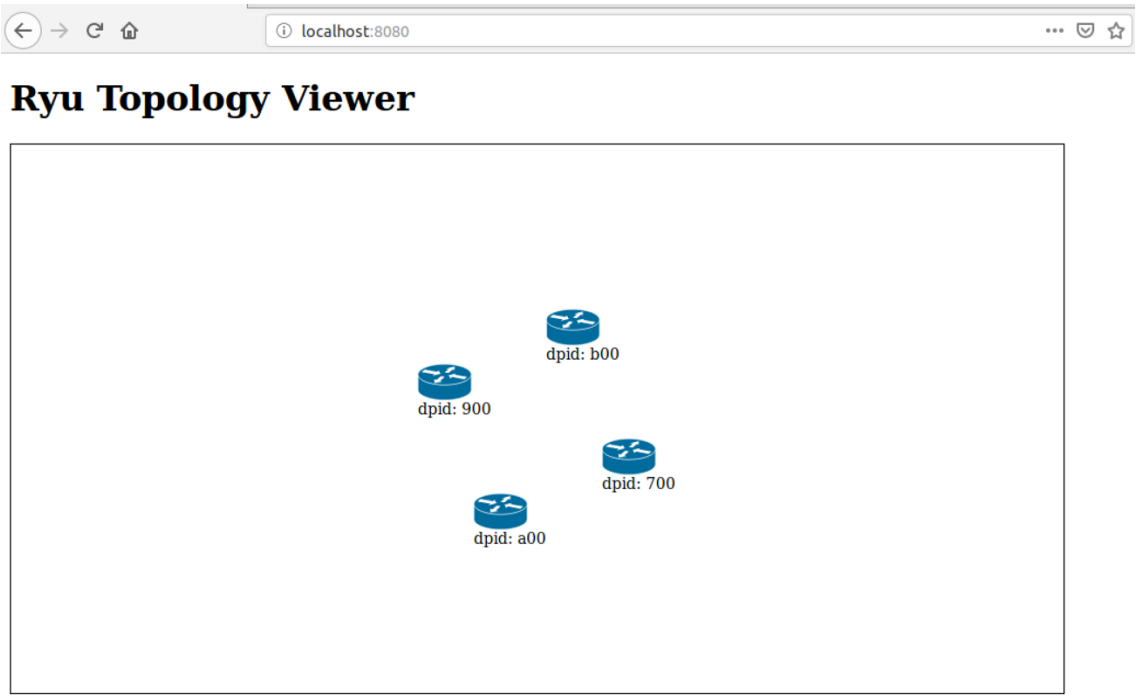

Figura 14 – Interfaz web del controlador Ryu

ANEXO 1 – Comandos para ejecutar el controlador con las API´s correspondientes

sudo ovs-vsctl set Bridge Net9 protocols=OpenFlow13

sudo ovs-vsctl set Bridge Net7 protocols=OpenFlow13

sudo ovs-vsctl set-manager ptcp:6632

sed '/OFPFlowMod(/,/)/s/)/, table_id=1)/' ryu/ryu/app/simple_switch_13.py > ryu/ryu/app/qos_simple_switch_13.py

cd ryu/; python ./setup.py install

ryu-manager --observe-links ryu.app.rest_topology ryu.app.ws_topology ryu.app.gui_topology.gui_topology ryu.app.ofctl_rest ryu.app.rest_qos ryu.app.qos_simple_switch_13 ryu.app.rest_conf_switch

ANEXO 2 – Configuración para router vyos_1 (set-hostname)

```
configure
set service dhcp-server shared-network-name LAN authoritative enable
set service dhcp-server shared-network-name LAN subnet 10.0.1.0/24 default-router '10.0.1.1'
set service dhcp-server shared-network-name LAN subnet 10.0.1.0/24 dns-server '10.0.1.1'
set service dhcp-server shared-network-name LAN subnet 10.0.1.0/24 domain-name 'host-network'
set service dhcp-server shared-network-name LAN subnet 10.0.1.0/24 lease '86400'
set service dhcp-server disabled 'false'
set service dhcp-server shared-network-name LAN subnet 10.0.1.0/24 start 10.0.1.10 stop '10.0.1.254'
set service dhcp-server shared-network-name LAN subnet 10.0.1.0/24 static-mapping DHCP-H1 ip-address 10.0.1.7
set service dhcp-server shared-network-name LAN subnet 10.0.1.0/24 static-mapping DHCP-H1 mac-address 00:00:00:00:01:01
set service dhcp-server shared-network-name LAN subnet 10.0.1.0/24 static-mapping DHCP-H2 ip-address 10.0.1.8
set service dhcp-server shared-network-name LAN subnet 10.0.1.0/24 static-mapping DHCP-H2 mac-address 00:00:00:00:02:01
set interfaces ethernet eth1 ipv6 router-advert managed-flag 'true'
set interfaces ethernet eth1 ipv6 router-advert send-advert true
set interfaces ethernet eth1 ipv6 router-advert prefix 2001:db8::/64
set service dhcpv6-server shared-network-name LAN6 subnet 2001:db8::/64 address-range prefix 2001:db8::/64
set service dhcpv6-server shared-network-name LAN6 subnet 2001:db8::/64 address-range start 2001:db8::10 stop 2001:db8::200
set firewall group network-group NETWORK-HOSTS network 10.0.1.0/24
set firewall group network-group NETWORK-H1 network 10.0.1.7/32
set firewall group network-group NETWORK-H2 network 10.0.1.8/32
set firewall name HACIA-HOSTS default-action drop
set firewall name HACIA-HOSTS rule 10 action accept
set firewall name HACIA-HOSTS rule 10 state established enable
set firewall name HACIA-HOSTS rule 10 state related enable
set firewall name HACIA-HOSTS rule 20 action accept
set firewall name HACIA-HOSTS rule 20 destination port 80
set firewall name HACIA-HOSTS rule 20 protocol tcp
set firewall name HACIA-HOSTS rule 20 destination group network-group NETWORK-H1
set firewall name HACIA-HOSTS rule 20 description Acceso_HTTP_a_H1
set firewall name HACIA-HOSTS rule 30 action accept
set firewall name HACIA-HOSTS rule 30 destination port 433
set firewall name HACIA-HOSTS rule 30 protocol tcp
set firewall name HACIA-HOSTS rule 30 description Acceso_HTTPS_a_H1
set firewall name HACIA-HOSTS rule 30 destination group network-group NETWORK-H1
set firewall name HACIA-HOSTS rule 40 action accept
set firewall name HACIA-HOSTS rule 40 destination port 22
set firewall name HACIA-HOSTS rule 40 protocol tcp
set firewall name HACIA-HOSTS rule 40 destination group network-group NETWORK-H2
set firewall name HACIA-HOSTS rule 40 description Acceso_SSH_a_H2
set firewall ipv6-name HACIA-HOSTS-v6 default-action drop
```

```
set firewall ipv6-name HACIA-HOSTS-v6 rule 10 action accept
set firewall ipv6-name HACIA-HOSTS-v6 rule 10 state established enable
set firewall ipv6-name HACIA-HOSTS-v6 rule 10 state related enable
set firewall ipv6-name HACIA-HOSTS-v6 rule 20 action accept
set firewall ipv6-name HACIA-HOSTS-v6 rule 20 destination port 80
set firewall ipv6-name HACIA-HOSTS-v6 rule 20 protocol tcp
set firewall ipv6-name HACIA-HOSTS-v6 rule 20 destination address 2001:db8:0:0:200:ff:fe00:101/128
set firewall ipv6-name HACIA-HOSTS-v6 rule 20 description Acceso_HTTP_a_H1-v6
set firewall ipv6-name HACIA-HOSTS-v6 rule 30 action accept
set firewall ipv6-name HACIA-HOSTS-v6 rule 30 destination port 433
set firewall ipv6-name HACIA-HOSTS-v6 rule 30 protocol tcp
set firewall ipv6-name HACIA-HOSTS-v6 rule 30 description Acceso_HTTPS_a_H1-v6
set firewall ipv6-name HACIA-HOSTS-v6 rule 30 destination address 2001:db8:0:0:200:ff:fe00:101/128
set firewall ipv6-name HACIA-HOSTS-v6 rule 40 action accept
set firewall ipv6-name HACIA-HOSTS-v6 rule 40 destination port 22
set firewall ipv6-name HACIA-HOSTS-v6 rule 40 protocol tcp
set firewall ipv6-name HACIA-HOSTS-v6 rule 40 destination address 2001:db8:0:0:200:ff:fe00:201/128
set firewall ipv6-name HACIA-HOSTS-v6 rule 40 description Acceso_SSH_a_H2-v6
set firewall name HACIA-FUERA default-action accept
set interfaces ethernet eth2 firewall in name HACIA-HOSTS
set interfaces ethernet eth2 firewall in ipv6-name HACIA-HOSTS-v6
set interfaces ethernet eth1 firewall local name HACIA-FUERA
set nat source rule 100 outbound-interface eth2
set nat source rule 100 source address 10.0.1.0/24
set nat source rule 100 translation address masquerade
commit
save
```

ANEXO 3 – Configuración para router vyos_2 (set-hostname2)

```
configure
set service dhcp-server shared-network-name LAN authoritative enable
set service dhcp-server shared-network-name LAN subnet 10.4.1.0/24 default-router 10.4.1.1
set service dhcp-server shared-network-name LAN subnet 10.4.1.0/24 dns-server 10.4.1.1
set service dhcp-server shared-network-name LAN subnet 10.4.1.0/24 domain-name host-network
set service dhcp-server shared-network-name LAN subnet 10.4.1.0/24 lease '86400'
set service dhcp-server disabled 'false'
set service dhcp-server shared-network-name LAN subnet 10.4.1.0/24 start 10.4.1.10 stop 10.4.1.254
set service dhcp-server shared-network-name LAN subnet 10.4.1.0/24 static-mapping DHCP-H3 ip-
address 10.4.1.7
set service dhcp-server shared-network-name LAN subnet 10.4.1.0/24 static-mapping DHCP-H3 mac-
address 00:00:00:00:0C:01
set service dhcp-server shared-network-name LAN subnet 10.4.1.0/24 static-mapping DHCP-H4 ip-
address 10.4.1.8
set service dhcp-server shared-network-name LAN subnet 10.4.1.0/24 static-mapping DHCP-H4 mac-
address 00:00:00:00:0D:01
set interfaces ethernet eth1 ipv6 router-advert managed-flag 'true'
set interfaces ethernet eth1 ipv6 router-advert send-advert true
set interfaces ethernet eth1 ipv6 router-advert prefix 2001:db8:0:4::/64
set service dhcpv6-server shared-network-name LAN6 subnet 2001:db8:0:4::/64 address-range prefix
2001:db8:0:4::/64
set service dhcpv6-server shared-network-name LAN6 subnet 2001:db8:0:4::/64 address-range start
2001:db8:0:4::10 stop 2001:db8:0:4::200
set firewall group network-group NETWORK-HOSTS network 10.4.1.0/24
set firewall group network-group NETWORK-H3 network 10.4.1.7/32
set firewall group network-group NETWORK-H4 network 10.4.1.8/32
set firewall name HACIA-HOSTS default-action drop
set firewall name HACIA-HOSTS rule 10 action accept
set firewall name HACIA-HOSTS rule 10 state established enable
set firewall name HACIA-HOSTS rule 10 state related enable
set firewall name HACIA-HOSTS rule 20 action accept
set firewall name HACIA-HOSTS rule 20 destination port 80
set firewall name HACIA-HOSTS rule 20 protocol tcp
set firewall name HACIA-HOSTS rule 20 destination group network-group NETWORK-H3
set firewall name HACIA-HOSTS rule 20 description Acceso_HTTP_a_H3
```

```
set firewall name HACIA-HOSTS rule 30 action accept
set firewall name HACIA-HOSTS rule 30 destination port 433
set firewall name HACIA-HOSTS rule 30 protocol tcp
set firewall name HACIA-HOSTS rule 30 description Acceso_HTTPS_a_H3
set firewall name HACIA-HOSTS rule 30 destination group network-group NETWORK-H3
set firewall name HACIA-HOSTS rule 40 action accept
set firewall name HACIA-HOSTS rule 40 destination port 22
set firewall name HACIA-HOSTS rule 40 protocol tcp
set firewall name HACIA-HOSTS rule 40 destination group network-group NETWORK-H4
set firewall name HACIA-HOSTS rule 40 description Acceso_SSH_a_H4
set firewall ipv6-name HACIA-HOSTS-v6 default-action drop
set firewall ipv6-name HACIA-HOSTS-v6 rule 10 action accept
set firewall ipv6-name HACIA-HOSTS-v6 rule 10 state established enable
set firewall ipv6-name HACIA-HOSTS-v6 rule 10 state related enable
set firewall ipv6-name HACIA-HOSTS-v6 rule 20 action accept
set firewall ipv6-name HACIA-HOSTS-v6 rule 20 destination port 80
set firewall ipv6-name HACIA-HOSTS-v6 rule 20 protocol tcp
set firewall ipv6-name HACIA-HOSTS-v6 rule 20 destination address 2001:db8:0:4:200:ff:fe00:c01/128
set firewall ipv6-name HACIA-HOSTS-v6 rule 20 description Acceso_HTTP_a_H3-v6
set firewall ipv6-name HACIA-HOSTS-v6 rule 30 action accept
set firewall ipv6-name HACIA-HOSTS-v6 rule 30 destination port 433
set firewall ipv6-name HACIA-HOSTS-v6 rule 30 protocol tcp
set firewall ipv6-name HACIA-HOSTS-v6 rule 30 description Acceso_HTTPS_a_H3-v6
set firewall ipv6-name HACIA-HOSTS-v6 rule 30 destination address 2001:db8:0:4:200:ff:fe00:c01/128
set firewall ipv6-name HACIA-HOSTS-v6 rule 40 action accept
set firewall ipv6-name HACIA-HOSTS-v6 rule 40 destination port 22
set firewall ipv6-name HACIA-HOSTS-v6 rule 40 protocol tcp
set firewall ipv6-name HACIA-HOSTS-v6 rule 40 destination address 2001:db8:0:4:200:ff:fe00:d01/128
set firewall ipv6-name HACIA-HOSTS-v6 rule 40 description Acceso_SSH_a_H4-v6
set firewall name HACIA-FUERA default-action accept
set interfaces ethernet eth2 firewall in name HACIA-HOSTS
set interfaces ethernet eth2 firewall in ipv6-name HACIA-HOSTS-v6
set interfaces ethernet eth1 firewall local name HACIA-FUERA
set nat source rule 100 outbound-interface eth2
set nat source rule 100 source address 10.4.1.0/24
set nat source rule 100 translation address masquerade
commit
save
```

ANEXO 4 – Comandos útiles

LEVANTAR APACHE MANUALMENTE:

```
sudo vnx -f /usr/share/vnx/examples/practica_final.xml -v -x start-www -M h1
sudo vnx -f /usr/share/vnx/examples/practica_final.xml -v -x start-www -M h2
```

REALIZAR PETICIONES E ASIGNACIÓN DE IPv4 O IPv6

```
Dhclient
Dhclient -6
```

COMPROBAR FUNCIONAMIENTO DE NAT:

```
tcpdump -i eth1 (en el servidor)
```

COMPROBAR PUERTOS HABILITADOS:

```
nmap – n localhost
```

COMPROBAR QoS:

```
(En servidor) iperf -s -i 1
(En host) iperf -c ip_server -i 1
```