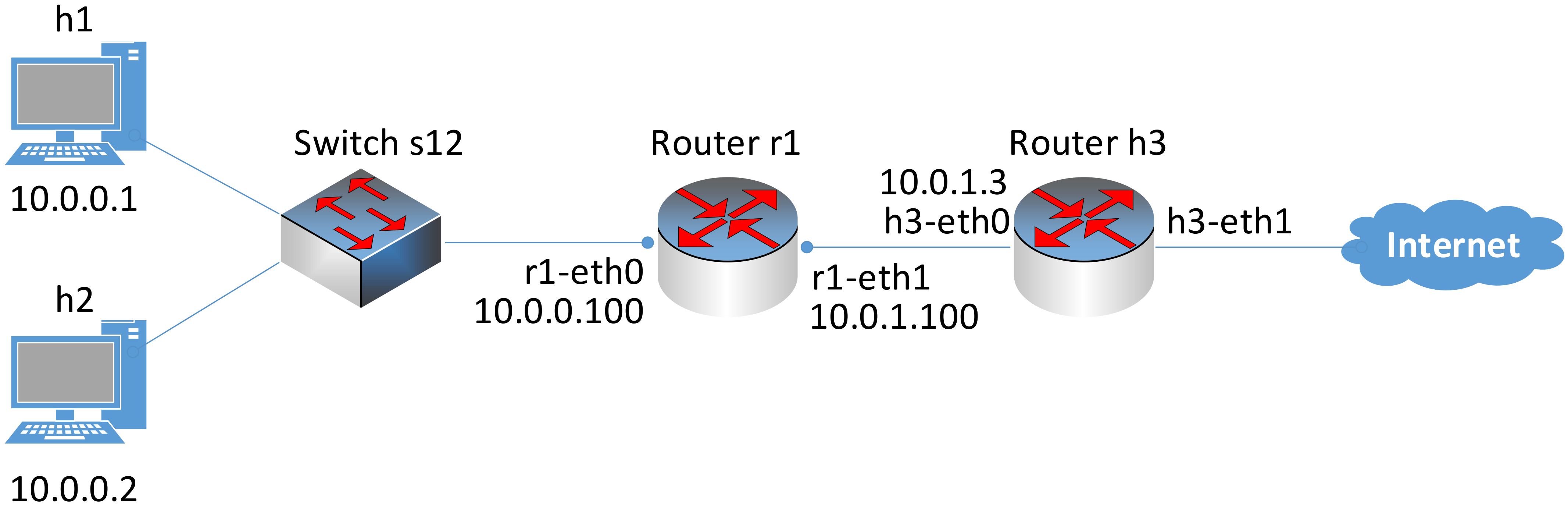
Connecting Virtual Environment to The Real World Using NAT

Connect a virtual network to the Internet

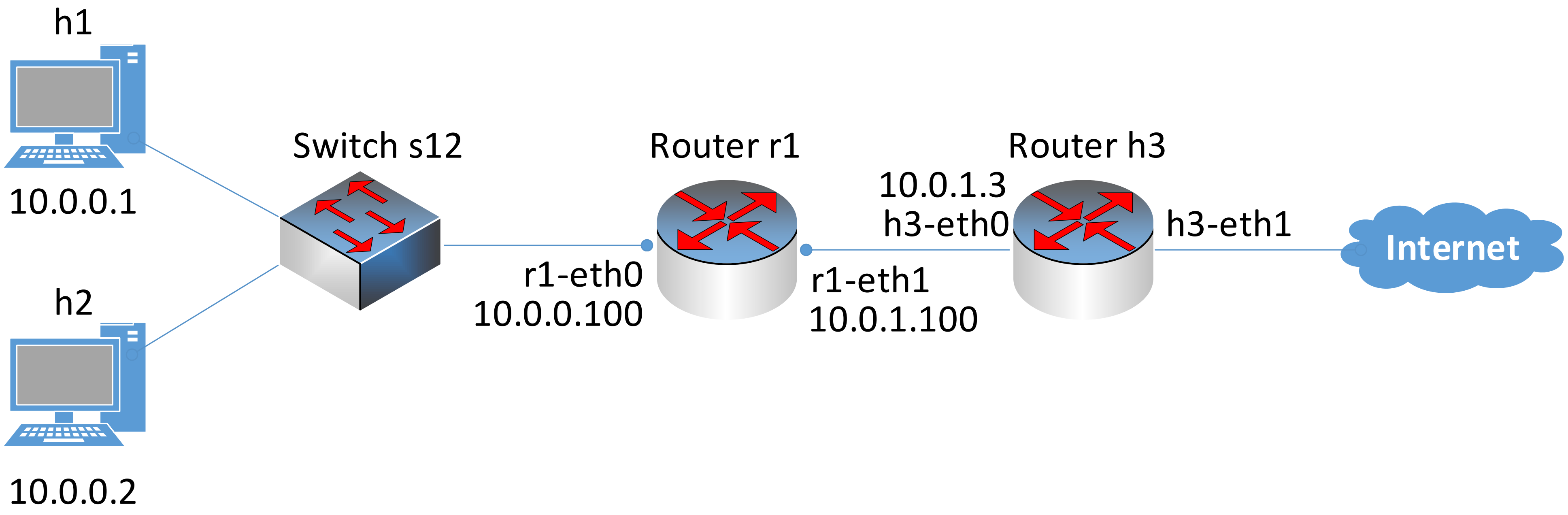
- Network configuration with a connection to the real world



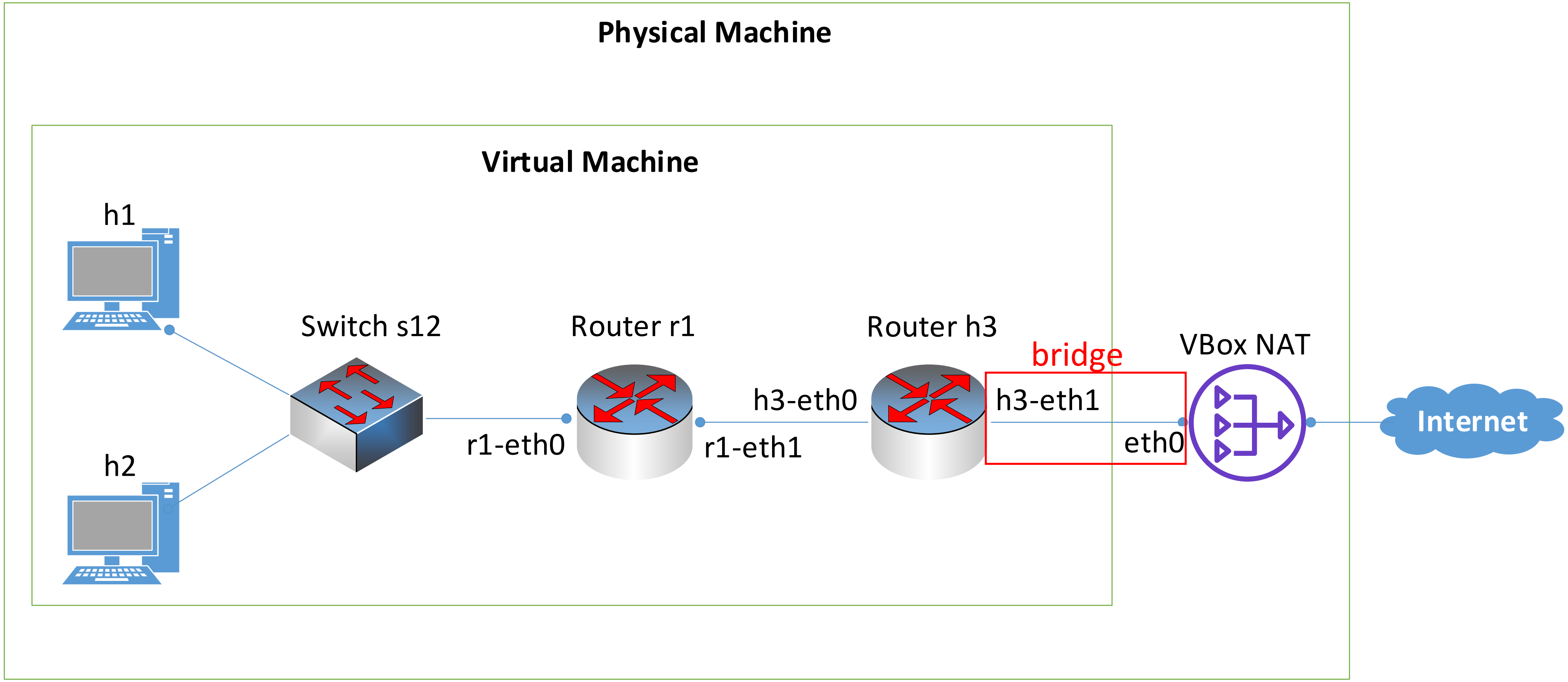
- NAT (Network Address Translation)

Connect a virtual network to the Internet

- Main steps:
  1. We require a real IP address on h3-eth1 interface of h3.
  2. We need to masquerade the traffic coming from h1 and h2.

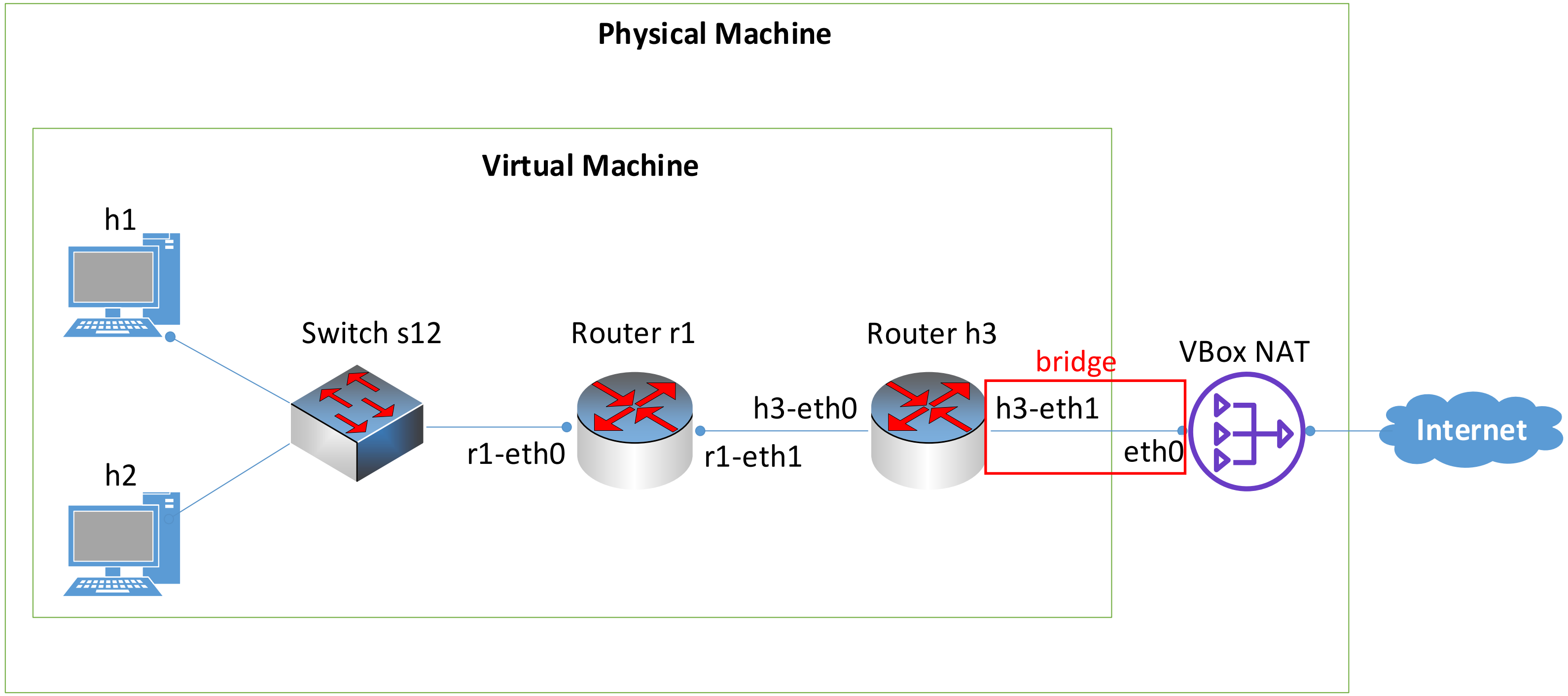


Bridging the network adapter

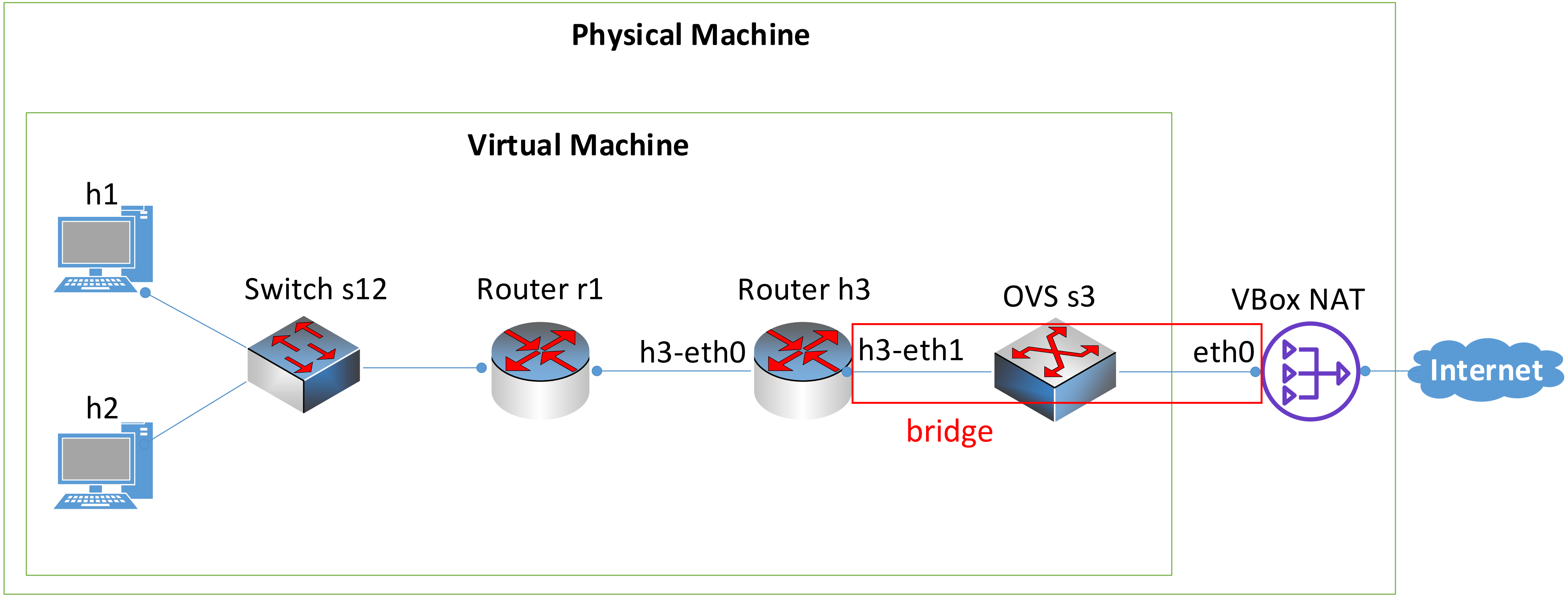


Borrowing an IP Address

1. Create a bridge between a real interface in your host machine, and h3's eth1.
2. Finding and setting up a suitable IP address for h3-eth1.

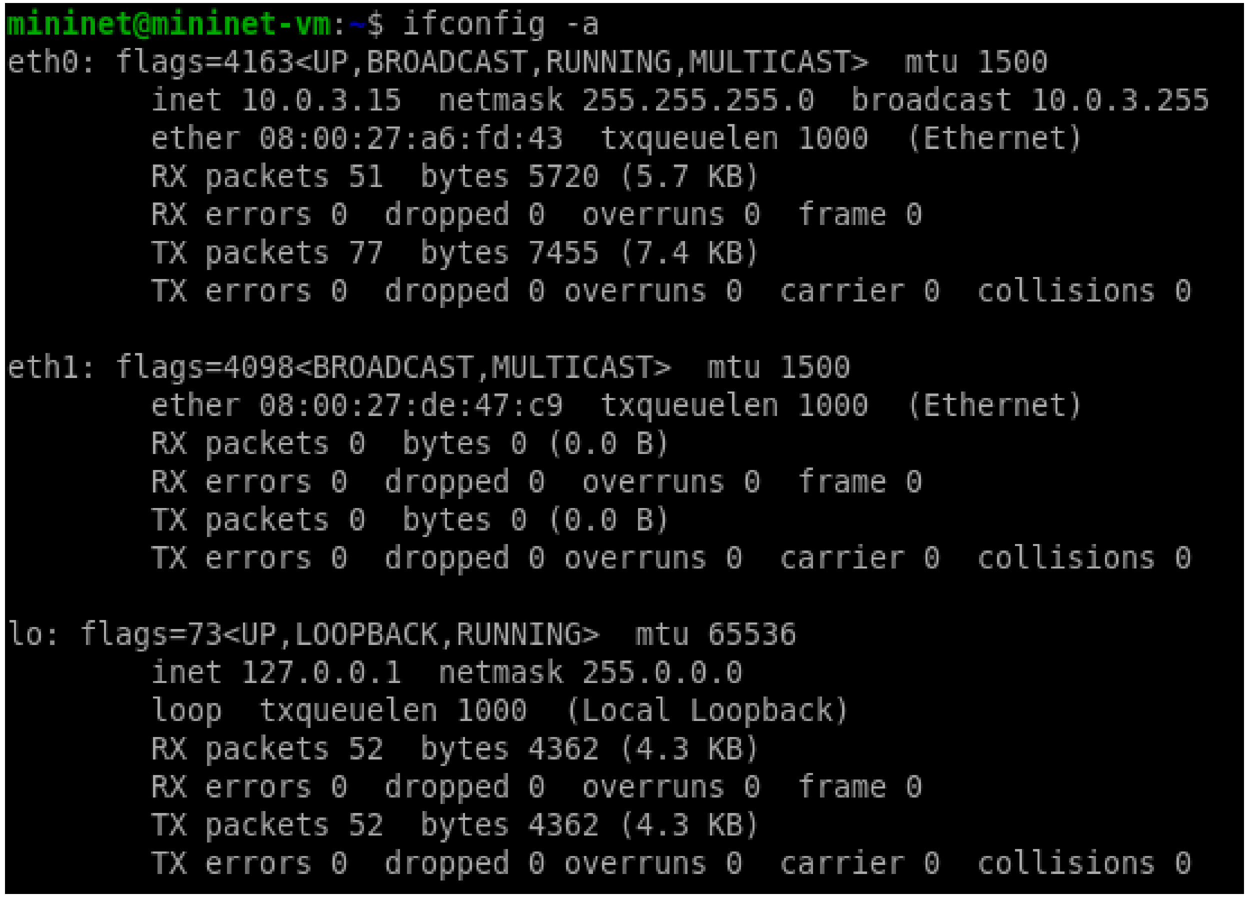


Using Open vSwitch (OVS)



How many interfaces

- we first need to find out which interface our VM is using to access the internet



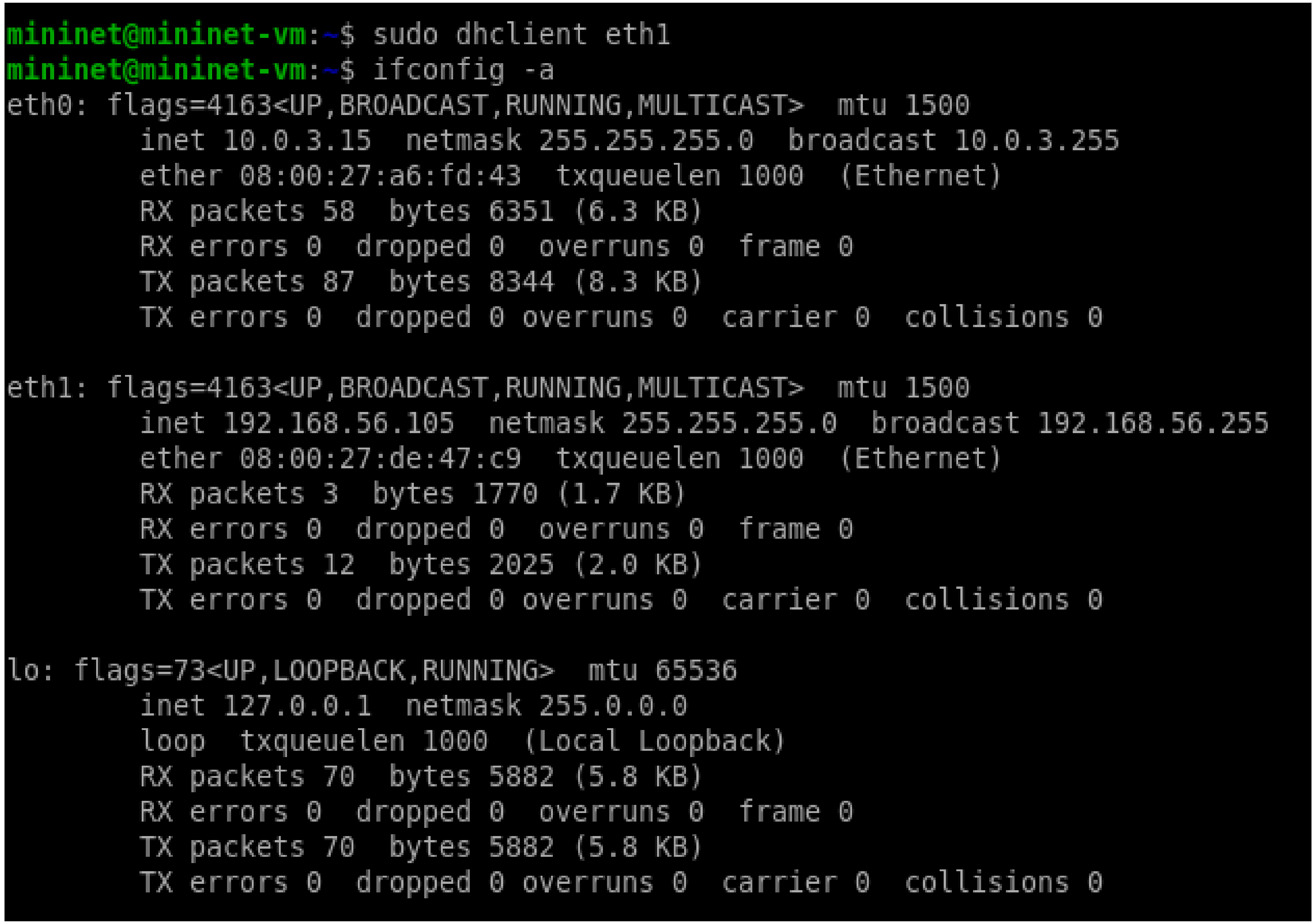
```
mininet@mininet-vm:~$ ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.3.15  netmask 255.255.255.0  broadcast 10.0.3.255
        ether 08:00:27:a6:fd:43  txqueuelen 1000  (Ethernet)
        RX packets 51  bytes 5720 (5.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 77  bytes 7455 (7.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 08:00:27:de:47:c9  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 52  bytes 4362 (4.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 52  bytes 4362 (4.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Provide an IP address

- $ sudo dhclient eth1



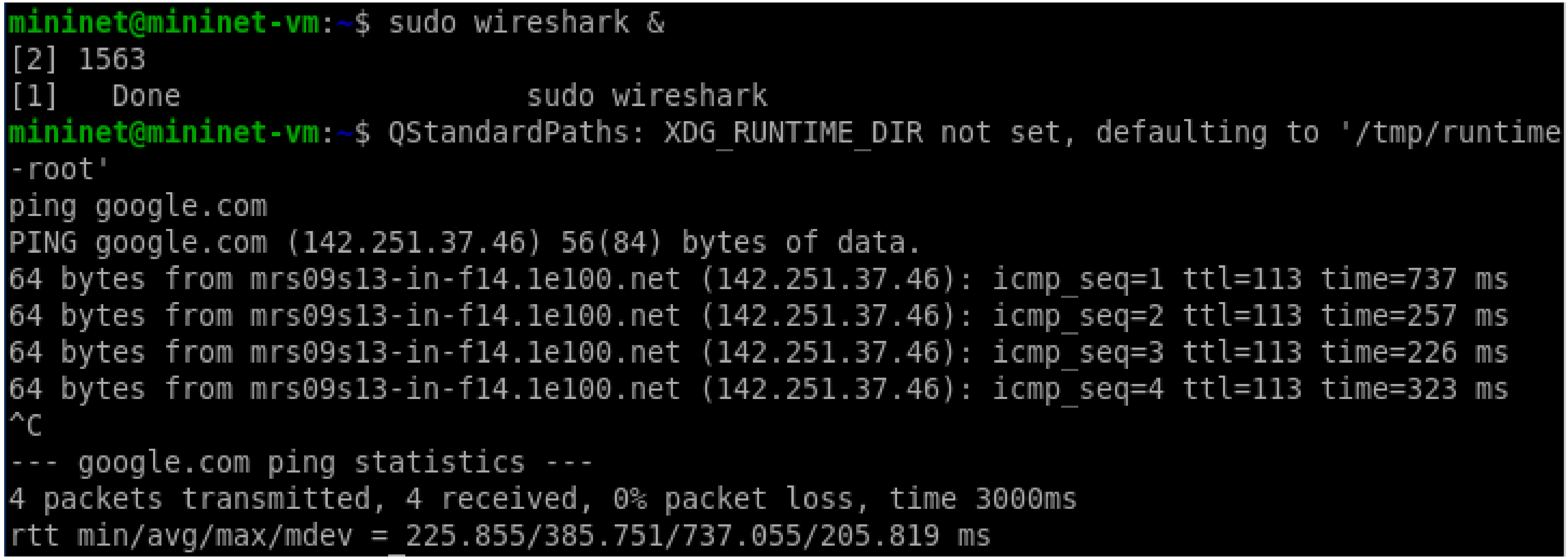
```
mininet@mininet-vm:~$ sudo dhclient eth1
mininet@mininet-vm:~$ ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.3.15  netmask 255.255.255.0  broadcast 10.0.3.255
        ether 08:00:27:a6:fd:43  txqueuelen 1000  (Ethernet)
        RX packets 58  bytes 6351 (6.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 87  bytes 8344 (8.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.56.105  netmask 255.255.255.0  broadcast 192.168.56.255
        ether 08:00:27:de:47:c9  txqueuelen 1000  (Ethernet)
        RX packets 3  bytes 1770 (1.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 2025 (2.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 70  bytes 5882 (5.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 70  bytes 5882 (5.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Which interface

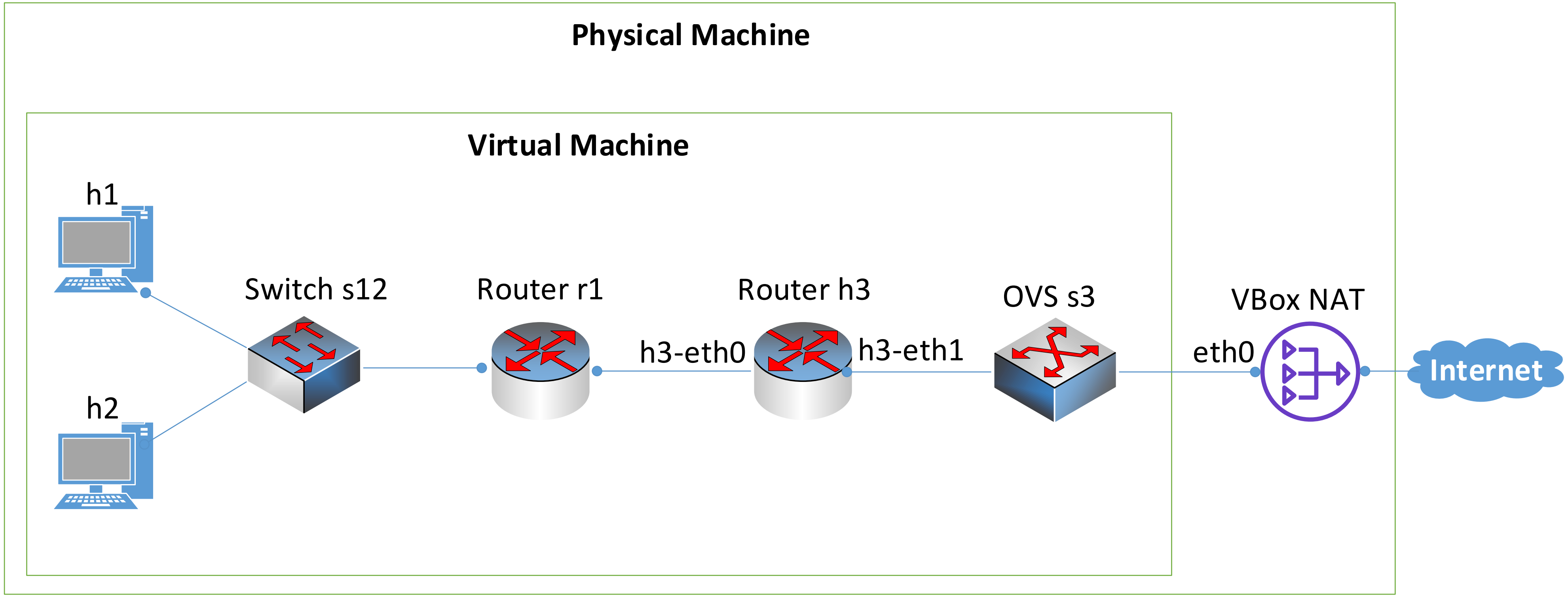
- $ sudo wireshark &
- ping google.com



```
mininet@mininet-vm:~$ sudo wireshark &
[2] 1563
[1]   Done                    sudo wireshark
mininet@mininet-vm:~$ QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime
-root'
ping google.com
PING google.com (142.251.37.46) 56(84) bytes of data.
64 bytes from mrs09s13-in-f14.1e100.net (142.251.37.46): icmp_seq=1 ttl=113 time=737 ms
64 bytes from mrs09s13-in-f14.1e100.net (142.251.37.46): icmp_seq=2 ttl=113 time=257 ms
64 bytes from mrs09s13-in-f14.1e100.net (142.251.37.46): icmp_seq=3 ttl=113 time=226 ms
64 bytes from mrs09s13-in-f14.1e100.net (142.251.37.46): icmp_seq=4 ttl=113 time=323 ms
^C
--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 225.855/385.751/737.055/205.819 ms
```

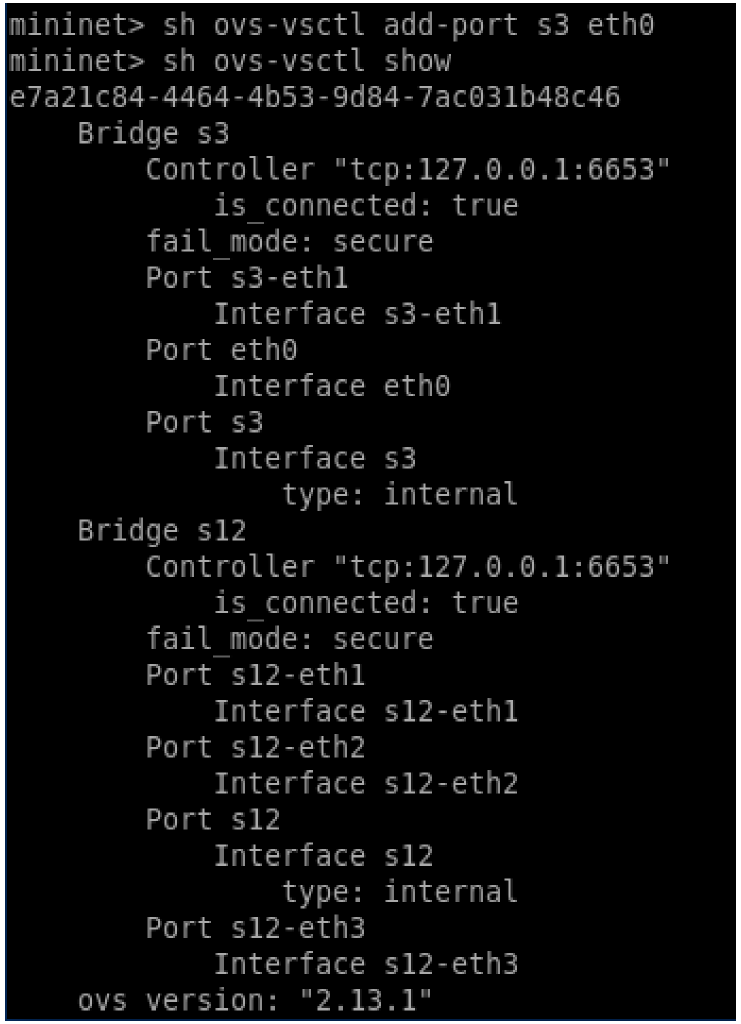
Bridging the network adapter

- $ sudo python topo2_int.py
- mininet> sh ovs-vsctl add-port s3 eth0



Show the interface

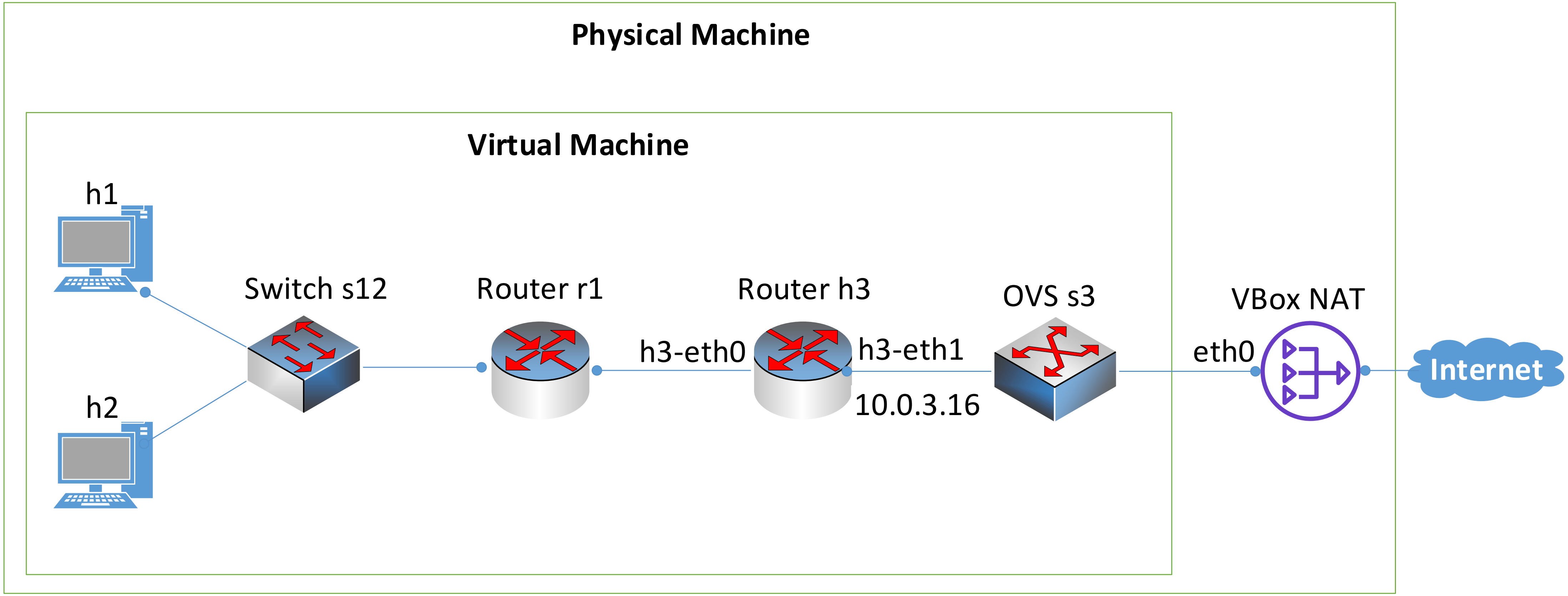
- mininet> sh ovs-vsctl show



```
mininet> sh ovs-vsctl add-port s3 eth0
mininet> sh ovs-vsctl show
e7a21c84-4464-4b53-9d84-7ac031b48c46
    Bridge s3
        Controller "tcp:127.0.0.1:6653"
            is_connected: true
        fail_mode: secure
        Port s3-eth1
            Interface s3-eth1
        Port eth0
            Interface eth0
        Port s3
            Interface s3
                type: internal
    Bridge s12
        Controller "tcp:127.0.0.1:6653"
            is_connected: true
        fail_mode: secure
        Port s12-eth1
            Interface s12-eth1
        Port s12-eth2
            Interface s12-eth2
        Port s12
            Interface s12
                type: internal
        Port s12-eth3
            Interface s12-eth3
    ovs_version: "2.13.1"
```

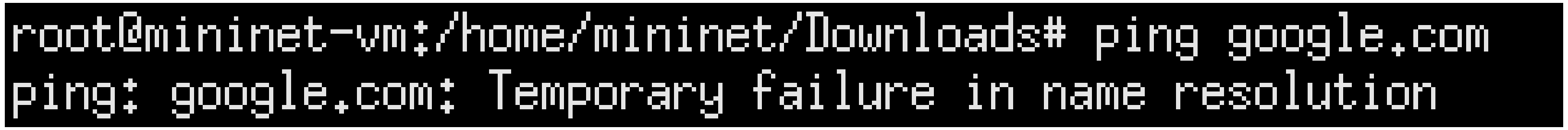
Set an IP address to h3-eth1

- # dhclient h3-eth1



Check connectivity

- # ping google.com

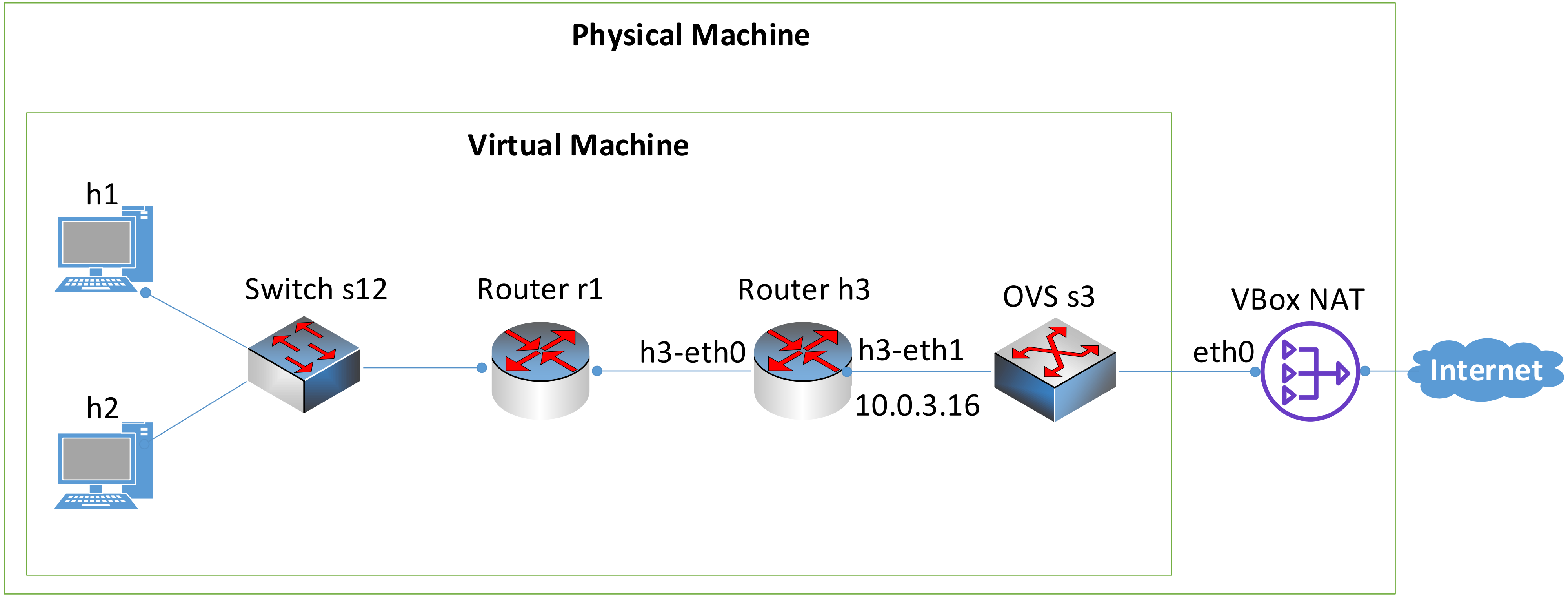


```
root@mininet-vm:/home/mininet/Downloads# ping google.com
ping: google.com: Temporary failure in name resolution
```

- This error occurs when the system cannot translate a website name into an IP address. The system cannot communicate with the DNS server and returns the error.

- # ping 8.8.8.8

Masquerade



- # iptables -t nat -A […] -o […] -j […]

Specify the address of DNS server

- Ping google with its IP address
  - `# ping 8.8.8.8`

- Configure the DNS server:
  - `# sudo echo nameserver 8.8.8.8 > /etc/resolv.conf`

- Ping google with its domain name:
  - `# ping google.com`