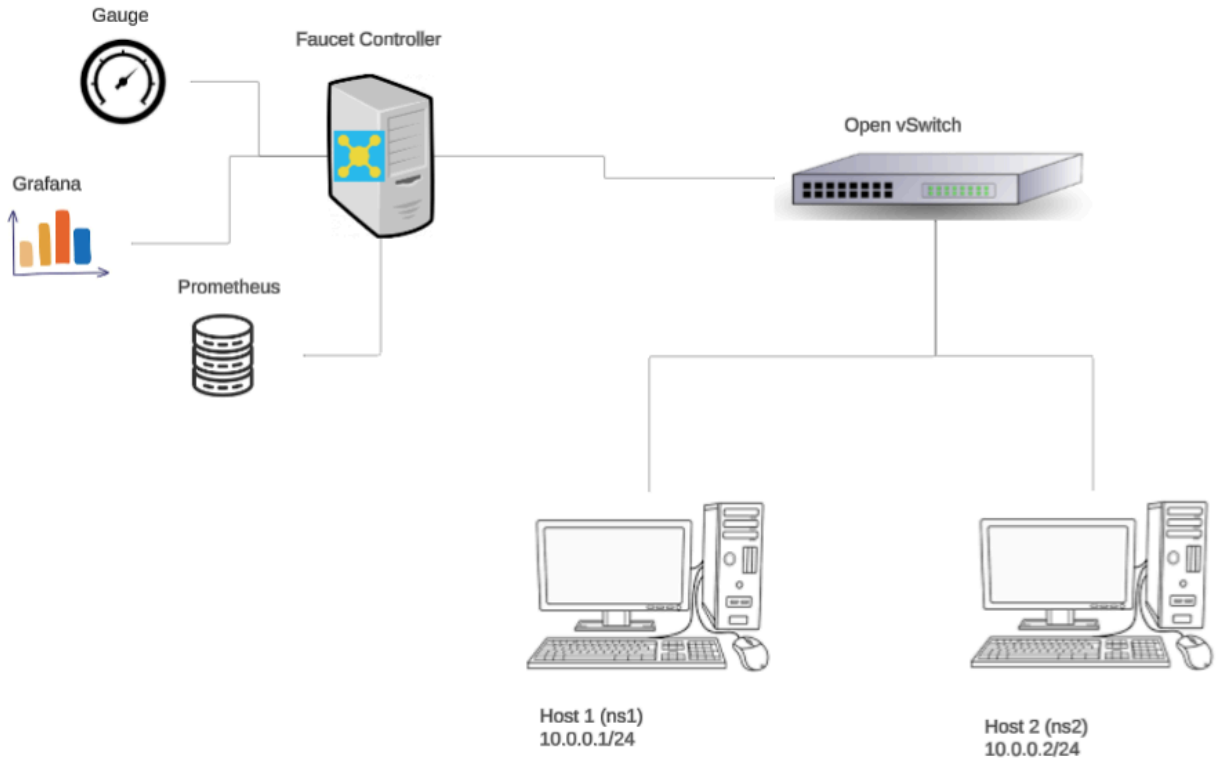
Eric Guzman
Professor Cannistra

4/21/2025
Network Virtualization

Lab 6

Description: In this lab we will go through several tutorials based around Faucet, learning how to configure different aspects of it.

Topology:



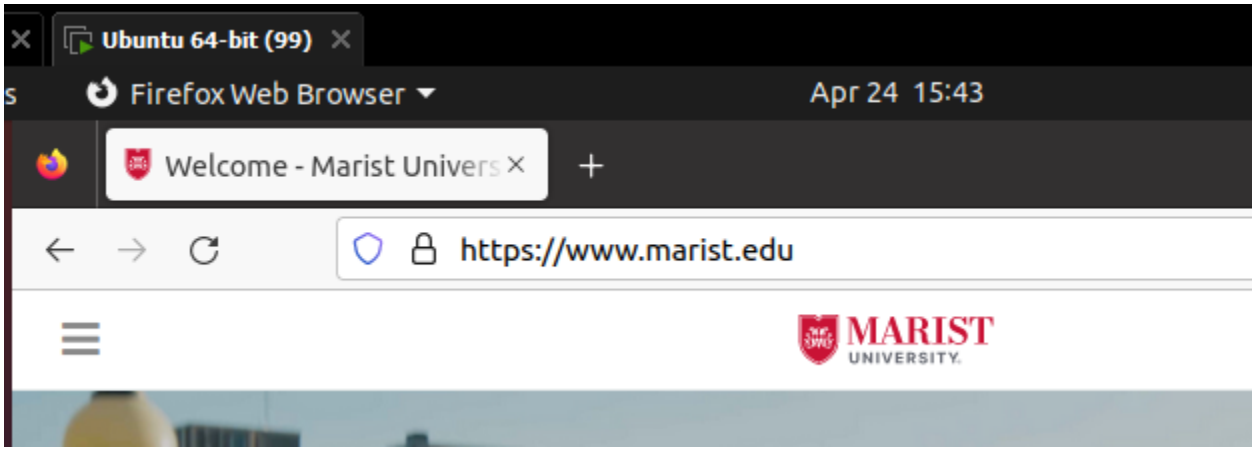
Syntax:

Command	Description
sudo apt-get install faucet-all-in-one	Installs the whole Faucet package at once
sudo ovs-vsctl add-br br0	Creates a Switch bridge named br0
sudo ip link add veth-ns1 type veth	Creates a link to a network

peer name veth-br1	namespace (ns1)
sudo systemctl reload faucet	Reloads the Faucet configuration
as_ns host1 tcpdump -l -e -n -i veth0	Starts a tcpdump session that captures ethernet traffic
as_ns () {}	Run commands inside a network namespace
create_ns () {}	Creates a network namespace
Nano /etc/faucet/faucet.yaml	Opens the Faucet.yaml file in the Nano text editor

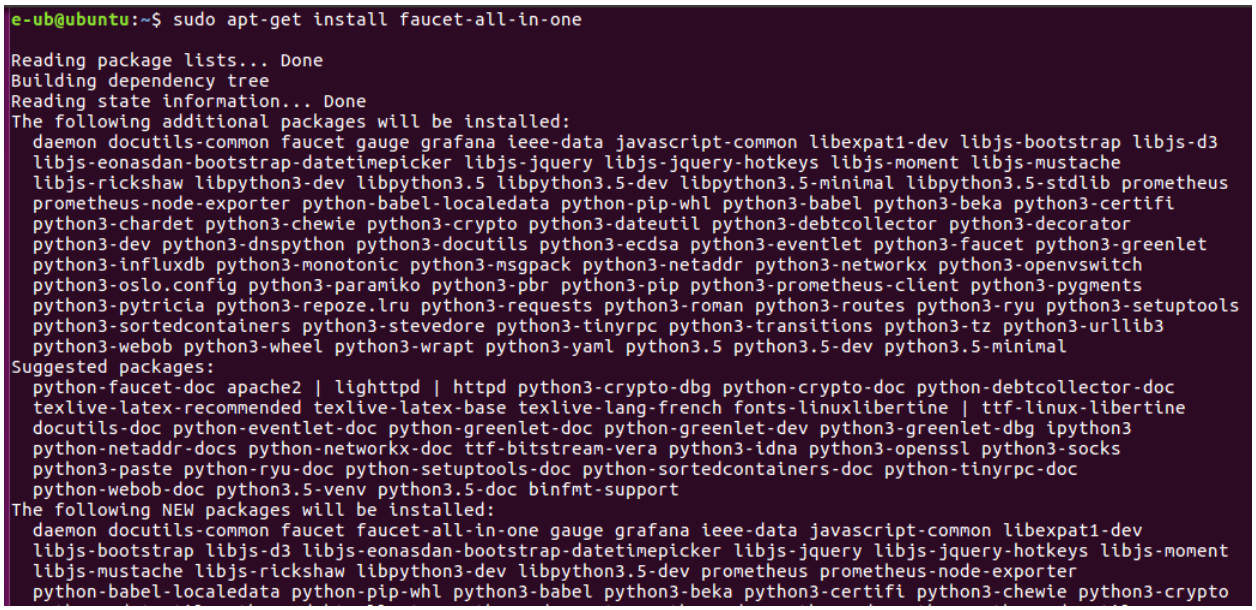
Verification:

Task One:

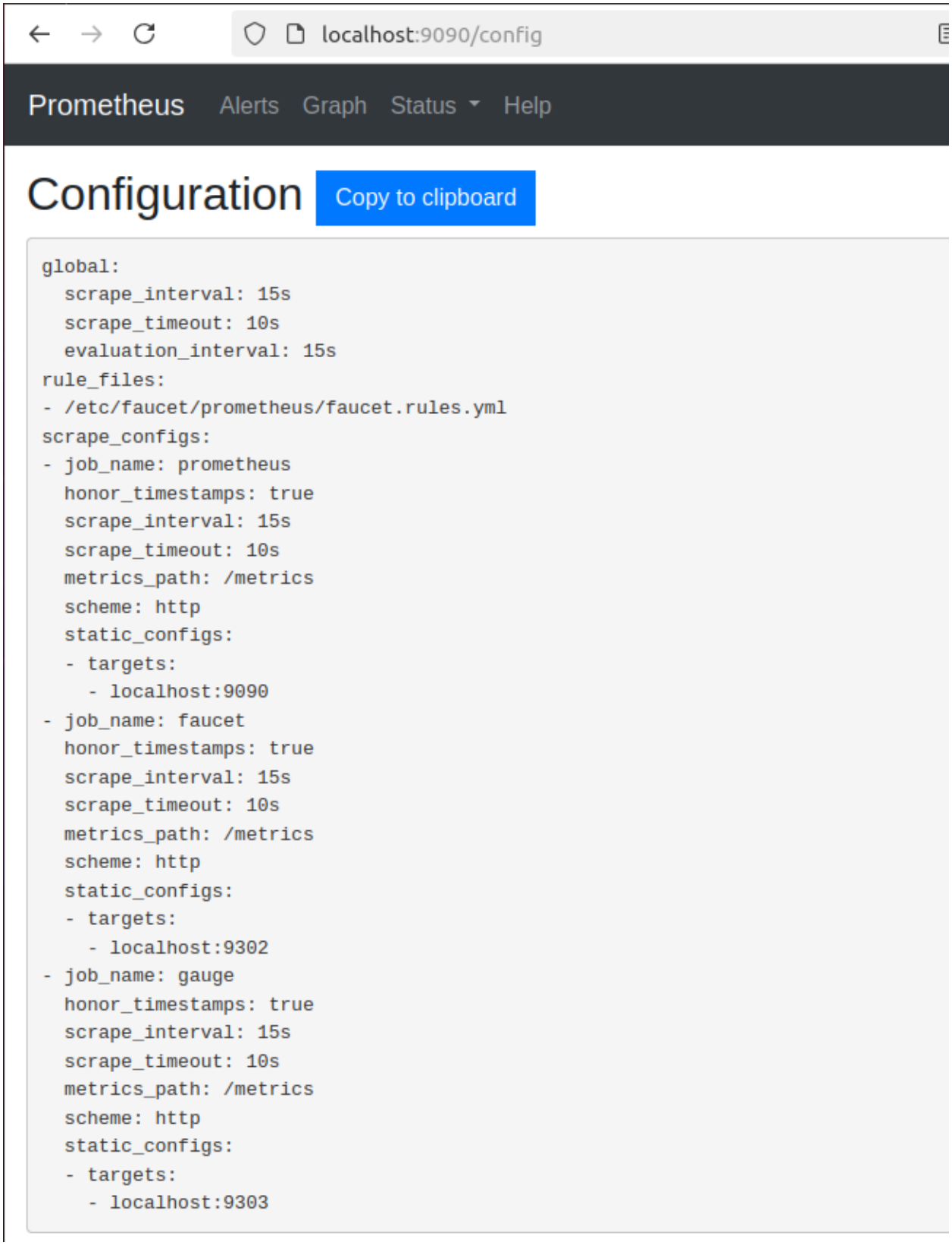


This screenshot shows that my new ubuntu VM has internet access!

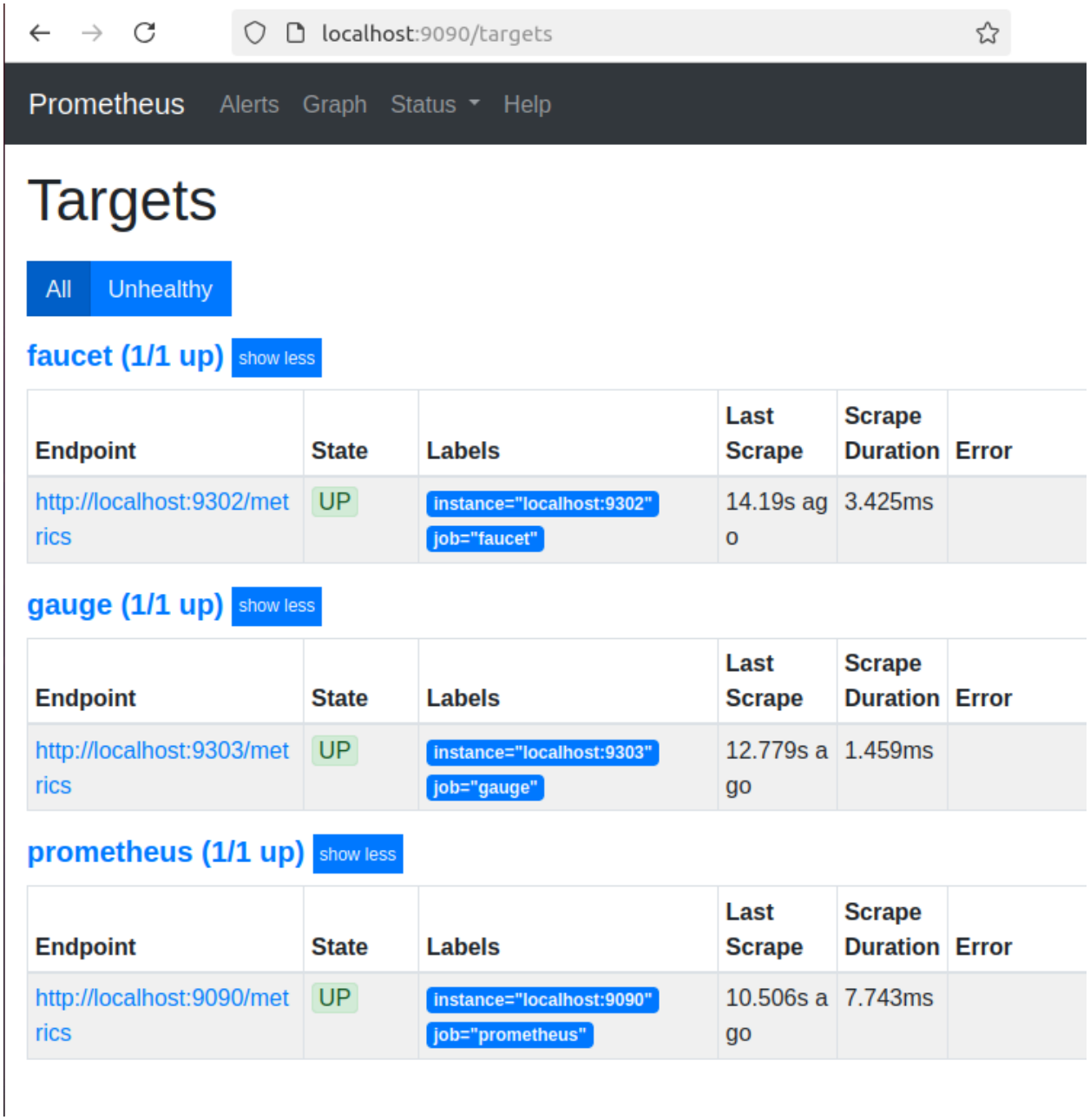
Task Two:



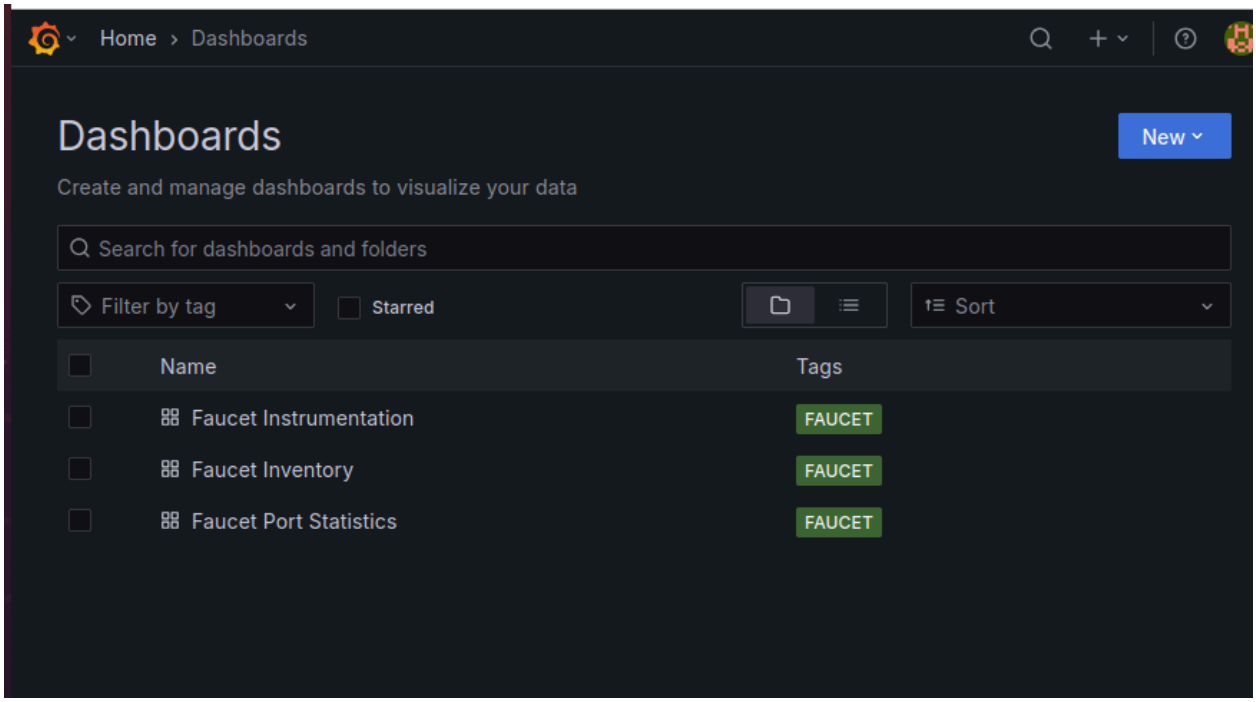
This screenshot shows that Faucet was installed successfully.



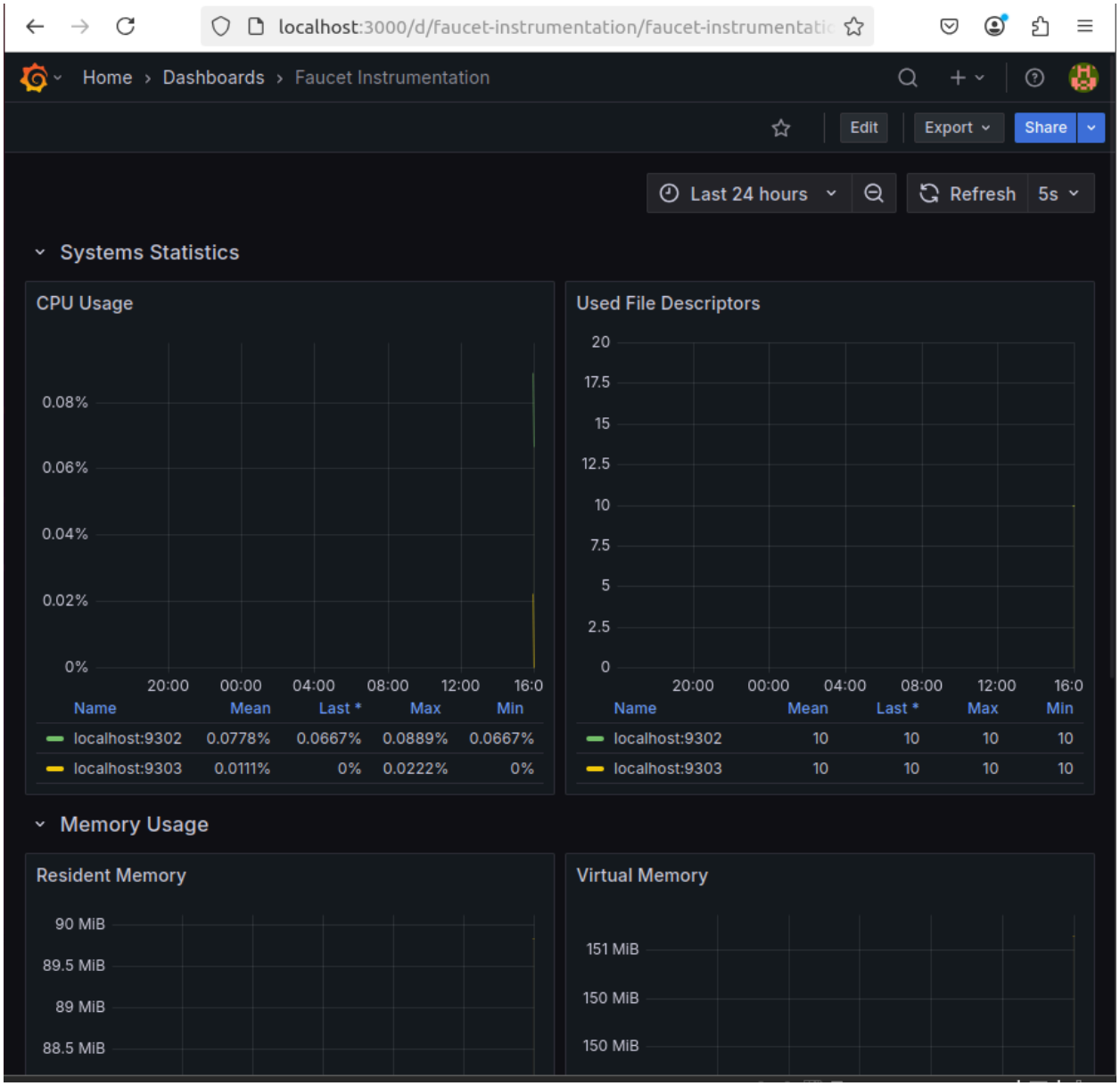
This screenshot shows my Prometheus /config page is up and running.



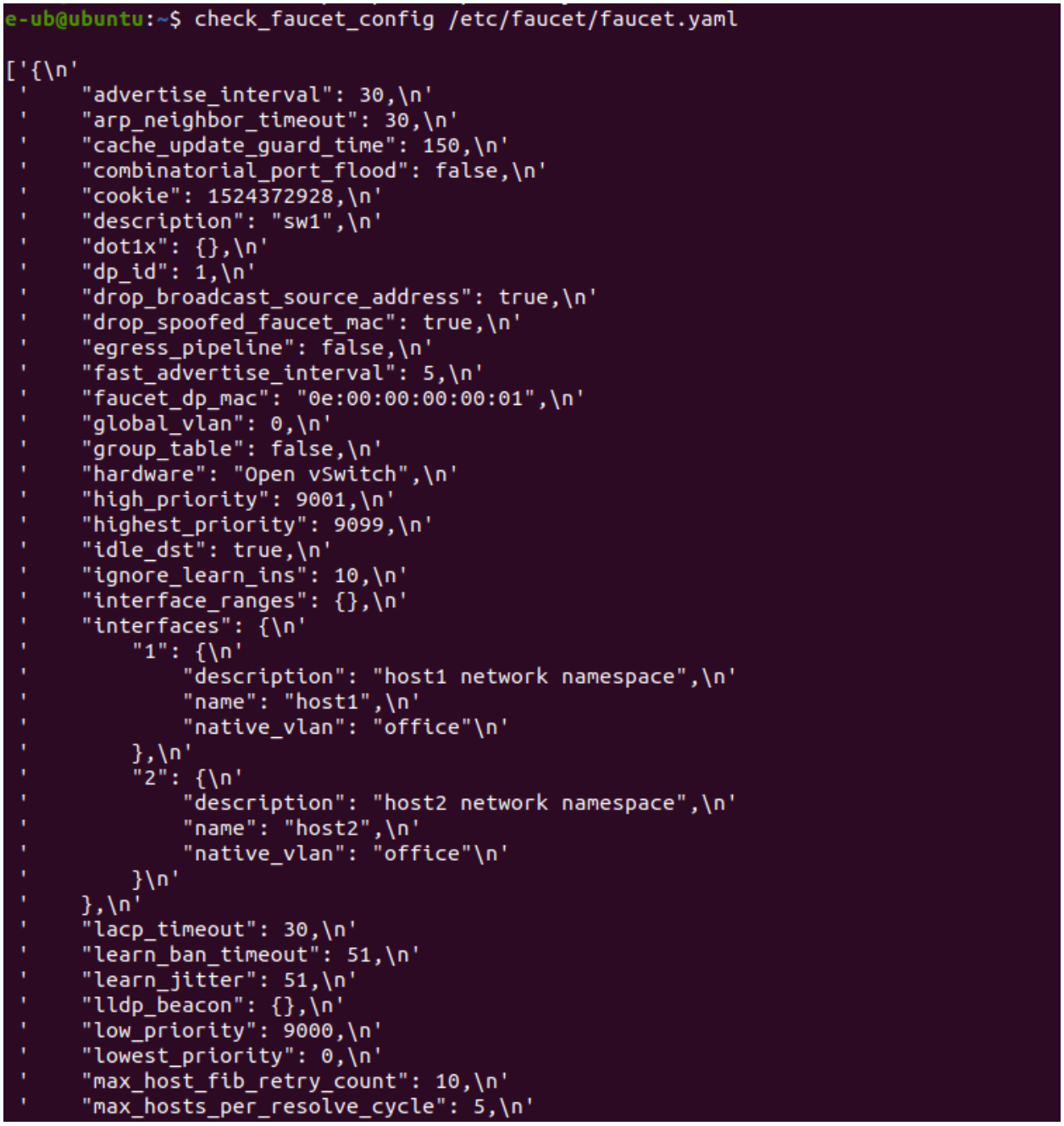
This screenshot shows my Prometheus /targets view page is up and running.



This screenshot shows that all of my Grafana dashboards were created properly.



This screenshot shows that my Faucet Instrumentation Grafana page is up.



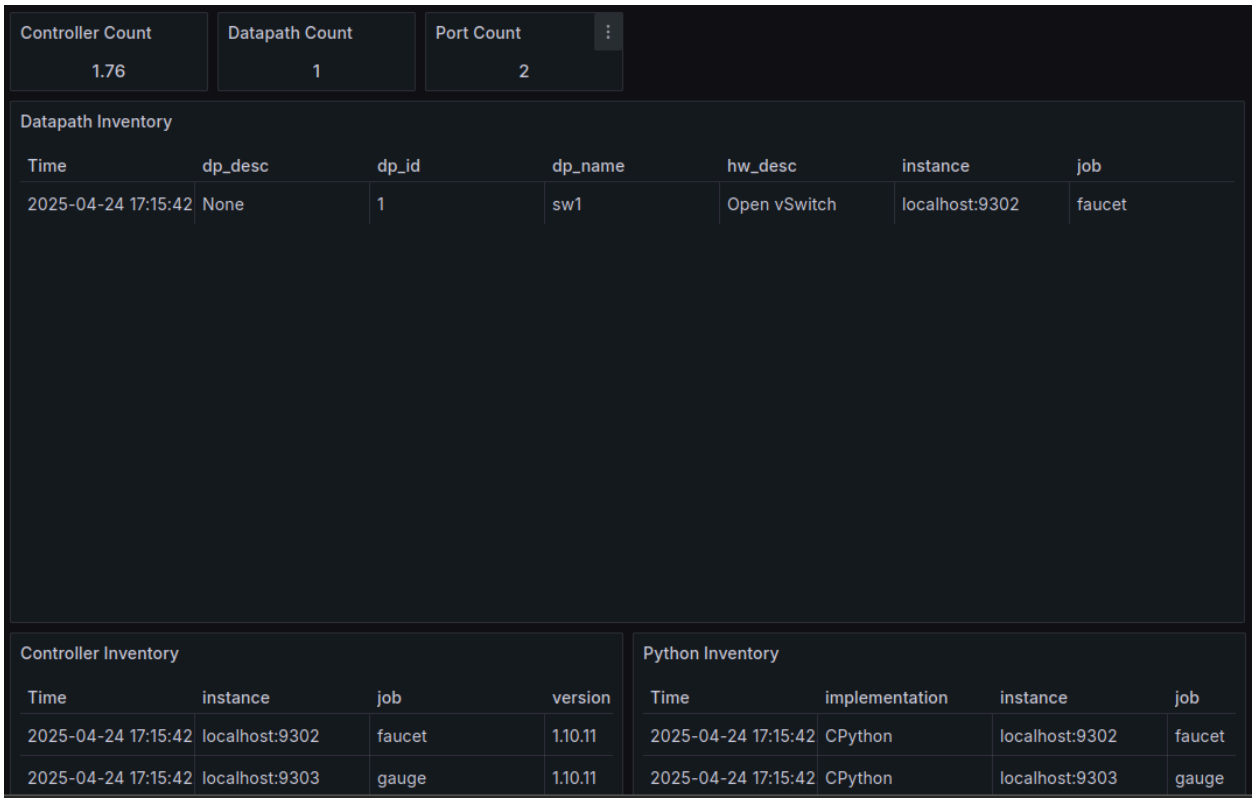
This screenshot shows the “check_faucet_config” command and that my configuration was loaded properly.

```
e-ub@ubuntu:~$ cat /var/log/faucet/faucet.log
Apr 24 15:52:34 faucet INFO     version 1.10.11
Apr 24 15:52:34 faucet INFO     Reloading configuration
Apr 24 15:52:34 faucet INFO     configuration /etc/faucet/faucet.yaml changed, analyzing differe
nces
Apr 24 15:52:34 faucet.config INFO     including file: /etc/faucet/acls.yaml
Apr 24 15:52:34 faucet INFO     Add new datapath DPID 1 (0x1)
Apr 24 15:52:34 faucet.valve INFO     DPID 1 (0x1) sw1 IPv4 routing is active on VLAN office vid
:100 tagged: Port 5 untagged: Port 1,Port 2,Port 4 with VIPs ['10.0.100.254/24']
Apr 24 15:52:34 faucet.valve INFO     DPID 1 (0x1) sw1 IPv4 routing is active on VLAN guest vid:
200 untagged: Port 3 with VIPs ['10.0.200.254/24']
Apr 24 15:52:34 faucet.valve INFO     DPID 1 (0x1) sw1 IPv6 routing is active on VLAN office vid
:100 tagged: Port 5 untagged: Port 1,Port 2,Port 4 with VIPs ['2001:100::1/64', 'fe80::c00:ff:fe
00:1001/64']
Apr 24 15:52:34 faucet.valve INFO     DPID 1 (0x1) sw1 IPv6 routing is active on VLAN guest vid:
200 untagged: Port 3 with VIPs ['2001:200::1/64', 'fe80::c00:ff:fe00:2001/64']
Apr 24 15:52:34 faucet INFO     Add new datapath DPID 2 (0x2)
Apr 24 15:52:34 faucet.valve INFO     DPID 2 (0x2) sw2 IPv4 routing is active on VLAN office vid
:100 tagged: Port 24 untagged: Port 1 with VIPs ['10.0.100.254/24']
Apr 24 15:52:34 faucet.valve INFO     DPID 2 (0x2) sw2 IPv4 routing is active on VLAN guest vid:
200 tagged: Port 24 untagged: Port 2,Port 4 with VIPs ['10.0.200.254/24']
Apr 24 15:52:34 faucet.valve INFO     DPID 2 (0x2) sw2 IPv6 routing is active on VLAN office vid
:100 tagged: Port 24 untagged: Port 1 with VIPs ['2001:100::1/64', 'fe80::c00:ff:fe00:1001/64']
Apr 24 15:52:34 faucet.valve INFO     DPID 2 (0x2) sw2 IPv6 routing is active on VLAN guest vid:
200 tagged: Port 24 untagged: Port 2,Port 4 with VIPs ['2001:200::1/64', 'fe80::c00:ff:fe00:2001
/64']
Apr 24 16:08:11 faucet INFO     Reloading configuration
Apr 24 16:08:11 faucet INFO     configuration /etc/faucet/faucet.yaml changed, analyzing differe
nces
Apr 24 16:08:11 faucet INFO     Reconfiguring existing datapath DPID 1 (0x1)
Apr 24 16:08:11 faucet.valve INFO     DPID 1 (0x1) sw1 DP routers config changed - requires cold
 start
Apr 24 16:08:11 faucet.valve INFO     DPID 1 (0x1) sw1 table IDs changed, old {0, 1, 2, 3, 4, 5,
 6, 7, 8} new {0, 1, 2, 3}
Apr 24 16:08:11 faucet.valve INFO     DPID 1 (0x1) sw1 cold starting
Apr 24 16:08:11 faucet.valve INFO     DPID 1 (0x1) sw1 forcing DP reconnection to ensure ports a
re synchronized
Apr 24 16:08:11 faucet.valve ERROR    DPID 1 (0x1) sw1 send_flow_msgs: DP not up
Apr 24 16:08:11 faucet INFO     Deleting de-configured DPID 2 (0x2)
e-ub@ubuntu:~$
```

This screenshot shows the faucet.log and that the configuration load was successful.

```
e-ub@ubuntu:~$ sudo ovs-vsctl show
4c1563ea-406a-44c8-9b10-03f6f6c98f5c
    Bridge br0
        Controller "tcp:127.0.0.1:6653"
            is_connected: true
        Controller "tcp:127.0.0.1:6654"
        Port br0
            Interface br0
                type: internal
        Port veth-host1
            Interface veth-host1
        Port veth-host2
            Interface veth-host2
    ovs_version: "2.13.8"
e-ub@ubuntu:~$
```

This screenshot shows the ovs-vsctl show command and that my controllers were connected successfully.



Controller Count	Datapath Count	Port Count
1.76	1	2

Datapath Inventory

Time	dp_desc	dp_id	dp_name	hw_desc	instance	job
2025-04-24 17:15:42	None	1	sw1	Open vSwitch	localhost:9302	faucet

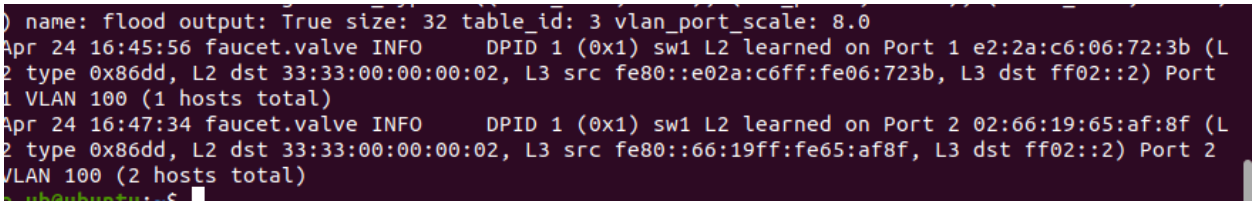
Controller Inventory

Time	instance	job	version
2025-04-24 17:15:42	localhost:9302	faucet	1.10.11
2025-04-24 17:15:42	localhost:9303	gauge	1.10.11

Python Inventory

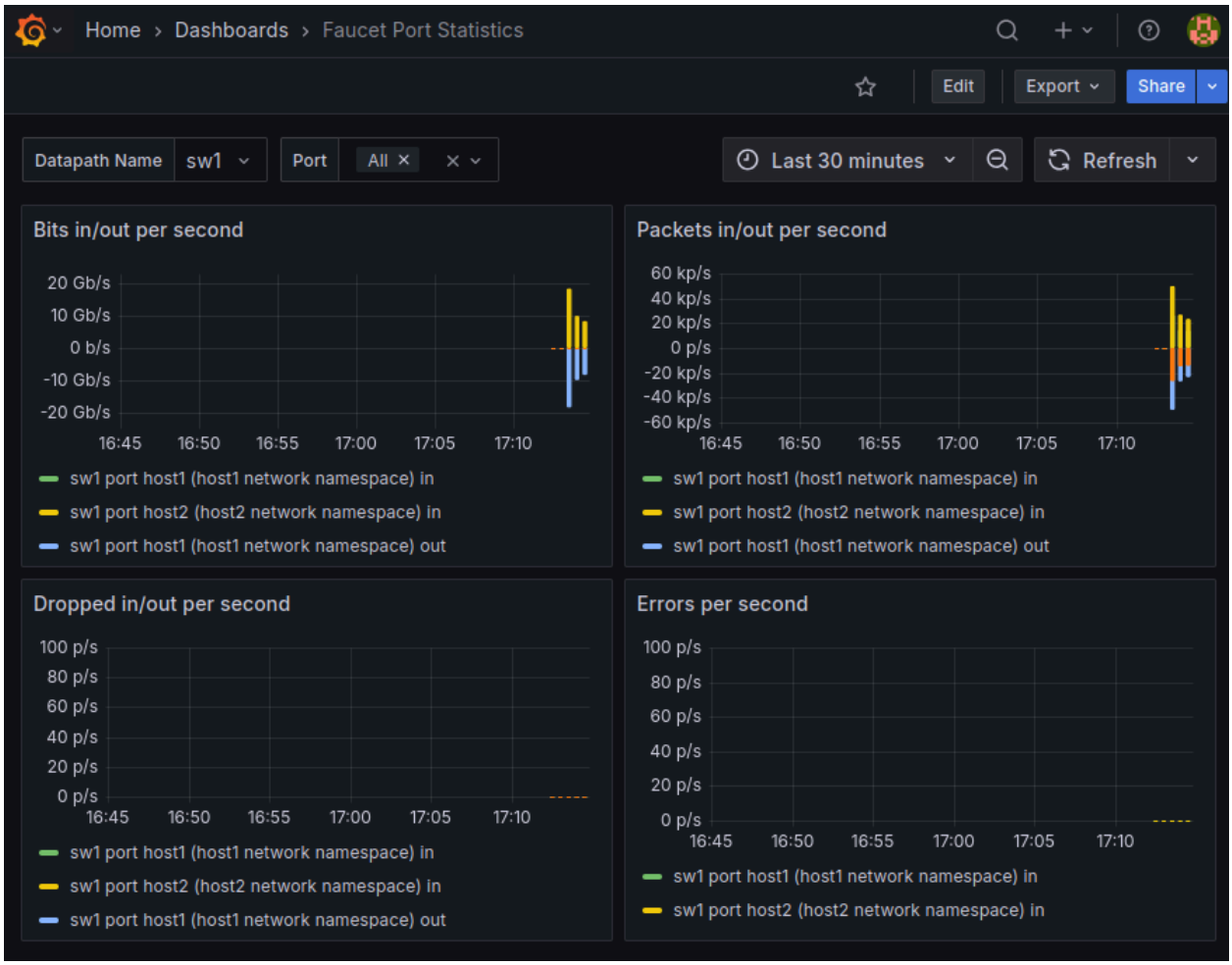
Time	implementation	instance	job
2025-04-24 17:15:42	CPython	localhost:9302	faucet
2025-04-24 17:15:42	CPython	localhost:9303	gauge

This screenshot shows my datapaths showing up on the Faucet Inventory Grafana page proving that they were connected correctly.



```
) name: flood output: True size: 32 table_id: 3 vlan_port_scale: 8.0
Apr 24 16:45:56 faucet.valve INFO     DPID 1 (0x1) sw1 L2 learned on Port 1 e2:2a:c6:06:72:3b (L
2 type 0x86dd, L2 dst 33:33:00:00:00:02, L3 src fe80::e02a:c6ff:fe06:723b, L3 dst ff02::2) Port
1 VLAN 100 (1 hosts total)
Apr 24 16:47:34 faucet.valve INFO     DPID 1 (0x1) sw1 L2 learned on Port 2 02:66:19:65:af:8f (L
2 type 0x86dd, L2 dst 33:33:00:00:00:02, L3 src fe80::66:19ff:fe65:af8f, L3 dst ff02::2) Port 2
VLAN 100 (2 hosts total)
```

This screenshot shows that Faucet learned the MAC addresses of host1 and host2 via the faucet.log file.

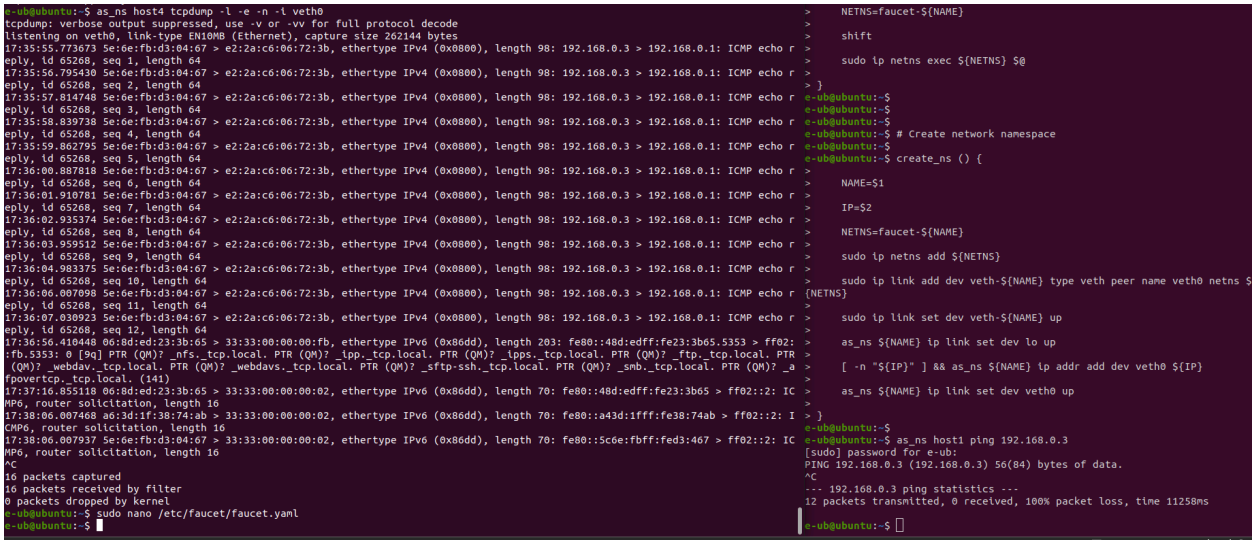


This screenshot shows the traffic generated from iperf3 and how it populated the Faucet Port Statistics page.

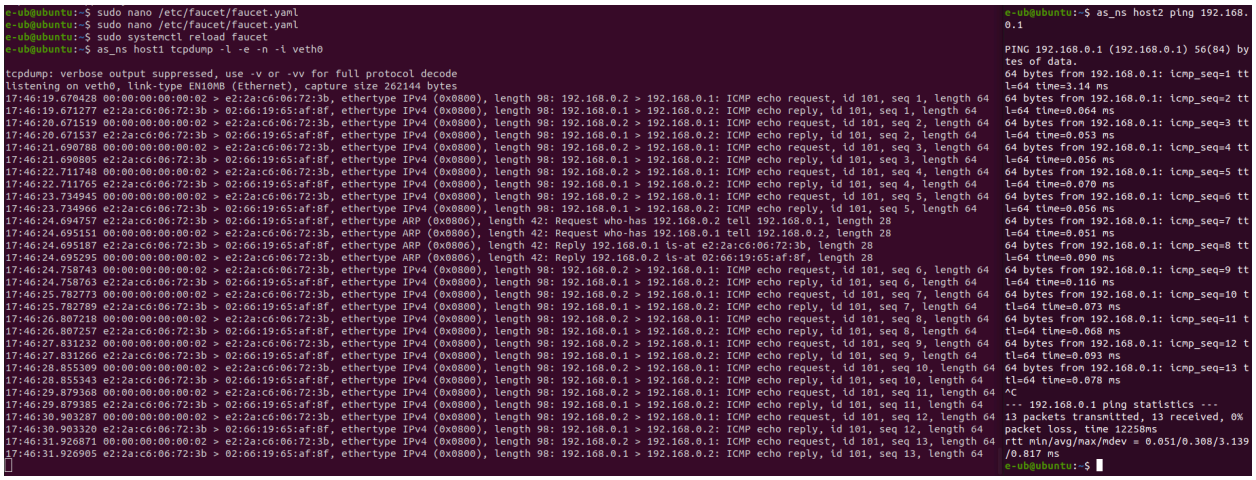
Task Three:

```
e-ub@ubuntu:~$ as_ns host1 ping 192.168.0.3

PING 192.168.0.3 (192.168.0.3) 56(84) bytes of data.
^C
--- 192.168.0.3 ping statistics ---
36 packets transmitted, 0 received, 100% packet loss, time 35858ms

e-ub@ubuntu:~$ as_ns host1 ping 192.168.0.2

PING 192.168.0.2 (192.168.0.2) 56(84) bytes of data.
64 bytes from 192.168.0.2: icmp_seq=1 ttl=64 time=1.34 ms
64 bytes from 192.168.0.2: icmp_seq=2 ttl=64 time=0.046 ms
64 bytes from 192.168.0.2: icmp_seq=3 ttl=64 time=0.046 ms
64 bytes from 192.168.0.2: icmp_seq=4 ttl=64 time=0.049 ms
^C
--- 192.168.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3044ms
rtt min/avg/max/mdev = 0.046/0.370/1.340/0.559 ms
e-ub@ubuntu:~$ as_ns host1 ping 192.168.0.4

PING 192.168.0.4 (192.168.0.4) 56(84) bytes of data.
64 bytes from 192.168.0.4: icmp_seq=1 ttl=64 time=0.405 ms
64 bytes from 192.168.0.4: icmp_seq=2 ttl=64 time=0.059 ms
64 bytes from 192.168.0.4: icmp_seq=3 ttl=64 time=0.053 ms
64 bytes from 192.168.0.4: icmp_seq=4 ttl=64 time=0.048 ms
^C
--- 192.168.0.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3080ms
rtt min/avg/max/mdev = 0.048/0.141/0.405/0.152 ms
e-ub@ubuntu:~$
```

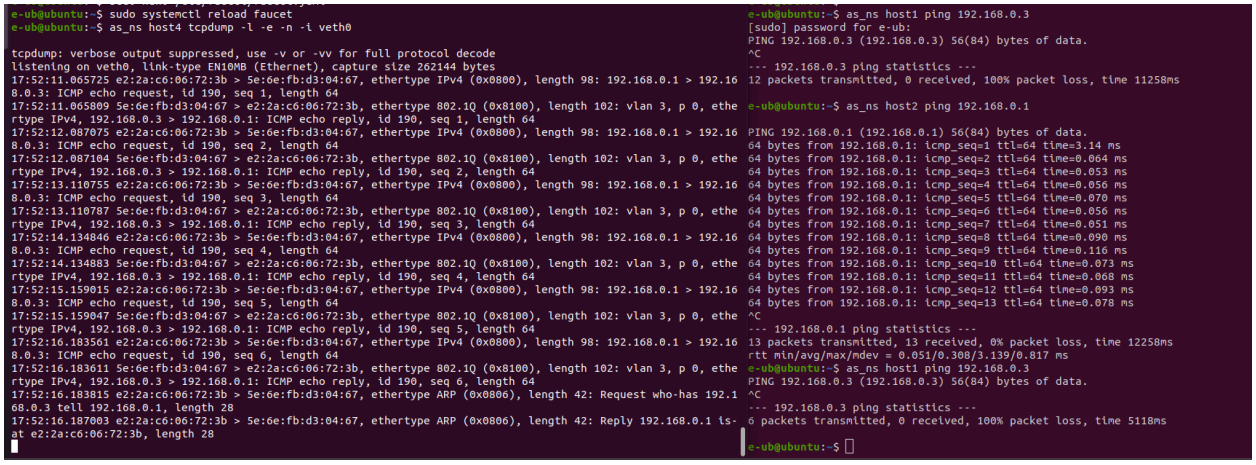
This screenshot shows that host1 is unable to ping host3 (192.168.0.3), giving 100% packet loss, confirming that the ACL is correctly blocking ICMP traffic to that host. Pings to host2 (192.168.0.2) and host4 (192.168.0.4) are successful, proving that the ACL is working properly.



This screenshot shows a tcpdump session showing that there is 100% packet loss from the ping from host1 to host3, confirming that the new ACL prevents traffic properly with the mirror action implemented.



This screenshot shows another tcpdump session but with a spoofed MAC address. The ACL is working properly by rewriting the ethernet header to spoof the MAC address as part of the ACL's output action.



This screenshot shows VLAN 3 tags and ethertype 802.1Q applied to ICMP packets. The ping from host1 to host3 still fails. VLAN tagging alongside the dropped traffic proves that both the ACLs and VLAN configurations are correctly implemented in Faucet.

Task Four:

```
e-ub@ubuntu:~$ as_ns host1 ping 192.168.0.2
PING 192.168.0.2 (192.168.0.2) 56(84) bytes of data.
64 bytes from 192.168.0.2: icmp_seq=1 ttl=64 time=1.75 ms
64 bytes from 192.168.0.2: icmp_seq=2 ttl=64 time=0.052 ms
64 bytes from 192.168.0.2: icmp_seq=3 ttl=64 time=0.061 ms
64 bytes from 192.168.0.2: icmp_seq=4 ttl=64 time=0.045 ms
^C
--- 192.168.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3047ms
rtt min/avg/max/mdev = 0.045/0.475/1.745/0.732 ms
e-ub@ubuntu:~$ as_ns host3 ping 192.168.0.4
PING 192.168.0.4 (192.168.0.4) 56(84) bytes of data.
64 bytes from 192.168.0.4: icmp_seq=1 ttl=64 time=12.6 ms
64 bytes from 192.168.0.4: icmp_seq=2 ttl=64 time=0.055 ms
64 bytes from 192.168.0.4: icmp_seq=3 ttl=64 time=0.049 ms
64 bytes from 192.168.0.4: icmp_seq=4 ttl=64 time=0.057 ms
^C
--- 192.168.0.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3048ms
rtt min/avg/max/mdev = 0.049/3.199/12.636/5.448 ms
e-ub@ubuntu:~$ as_ns host5 ping 192.168.2.6
PING 192.168.2.6 (192.168.2.6) 56(84) bytes of data.
64 bytes from 192.168.2.6: icmp_seq=1 ttl=64 time=0.436 ms
64 bytes from 192.168.2.6: icmp_seq=2 ttl=64 time=0.053 ms
64 bytes from 192.168.2.6: icmp_seq=3 ttl=64 time=0.043 ms
64 bytes from 192.168.2.6: icmp_seq=4 ttl=64 time=0.054 ms
^C
--- 192.168.2.6 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3077ms
rtt min/avg/max/mdev = 0.043/0.146/0.436/0.167 ms
e-ub@ubuntu:~$ as_ns host7 ping 192.168.3.8
PING 192.168.3.8 (192.168.3.8) 56(84) bytes of data.
64 bytes from 192.168.3.8: icmp_seq=1 ttl=64 time=1.41 ms
64 bytes from 192.168.3.8: icmp_seq=2 ttl=64 time=0.052 ms
64 bytes from 192.168.3.8: icmp_seq=3 ttl=64 time=0.051 ms
64 bytes from 192.168.3.8: icmp_seq=4 ttl=64 time=0.048 ms
^C
--- 192.168.3.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3037ms
rtt min/avg/max/mdev = 0.048/0.389/1.406/0.587 ms
e-ub@ubuntu:~$
```

This screenshot shows successful pings between hosts in the same VLAN, proving basic VLAN configuration.

```
e-ub@ubuntu:~$ as_ns host1 ping 192.168.0.3

PING 192.168.0.3 (192.168.0.3) 56(84) bytes of data.
64 bytes from 192.168.0.3: icmp_seq=1 ttl=64 time=1.44 ms
64 bytes from 192.168.0.3: icmp_seq=2 ttl=64 time=0.055 ms
64 bytes from 192.168.0.3: icmp_seq=3 ttl=64 time=0.069 ms
64 bytes from 192.168.0.3: icmp_seq=4 ttl=64 time=0.051 ms
^C
--- 192.168.0.3 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3042ms
rtt min/avg/max/mdev = 0.051/0.404/1.441/0.598 ms
e-ub@ubuntu:~$
```

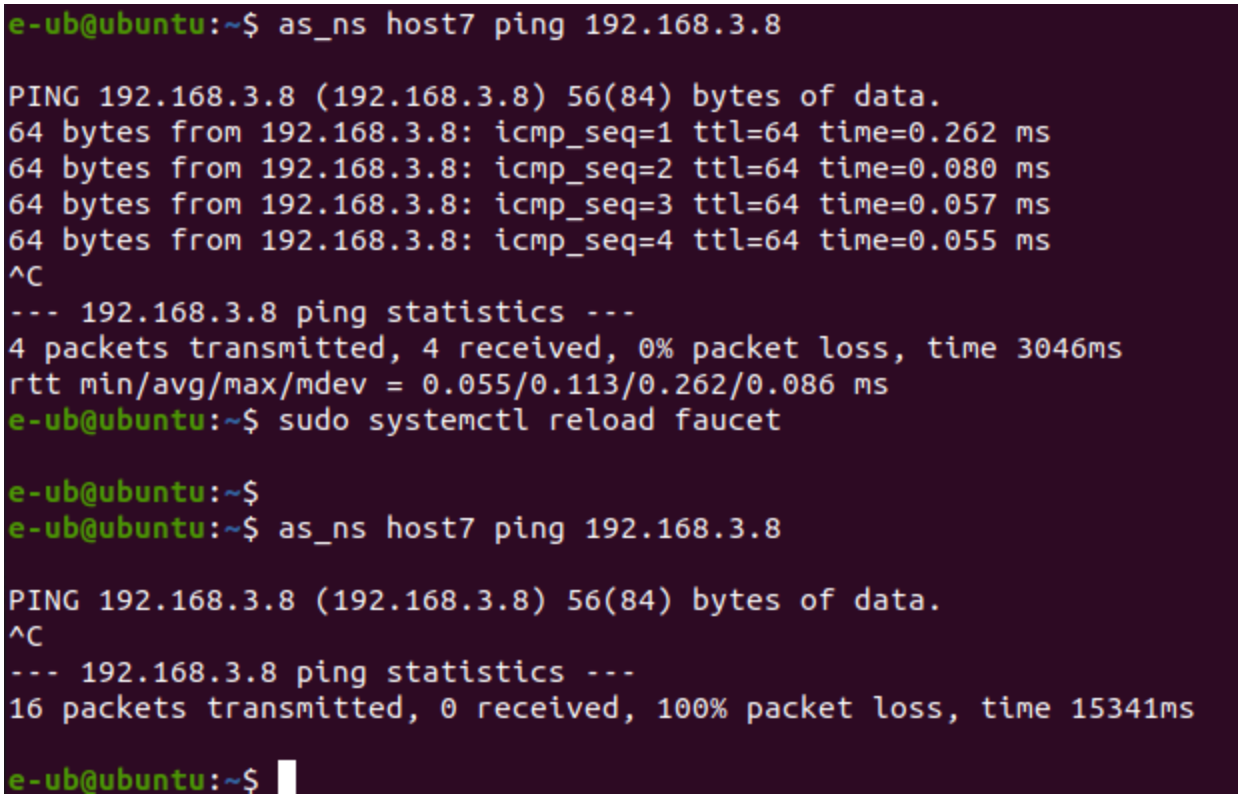
This screenshot shows that pinging between hosts in the same VLAN where the one host is native and the other is tagged also works properly.

```
e-ub@ubuntu:~$ as_ns host5 ip address add 192.168.0.5 dev veth0

e-ub@ubuntu:~$
e-ub@ubuntu:~$ as_ns host1 ping 192.168.0.5

PING 192.168.0.5 (192.168.0.5) 56(84) bytes of data.
From 192.168.0.1 icmp_seq=1 Destination Host Unreachable
From 192.168.0.1 icmp_seq=5 Destination Host Unreachable
From 192.168.0.1 icmp_seq=8 Destination Host Unreachable
From 192.168.0.1 icmp_seq=9 Destination Host Unreachable
^C
--- 192.168.0.5 ping statistics ---
11 packets transmitted, 0 received, +4 errors, 100% packet loss, time 10236ms
pipe 4
e-ub@ubuntu:~$
```

This screenshot shows that pinging from hosts in two different VLANs does not work (VLAN 100 to 200).

```
e-ub@ubuntu:~$ as_ns host1 ping 192.168.0.9

PING 192.168.0.9 (192.168.0.9) 56(84) bytes of data.
64 bytes from 192.168.0.9: icmp_seq=1 ttl=64 time=1.32 ms
64 bytes from 192.168.0.9: icmp_seq=2 ttl=64 time=0.051 ms
64 bytes from 192.168.0.9: icmp_seq=3 ttl=64 time=0.049 ms
64 bytes from 192.168.0.9: icmp_seq=4 ttl=64 time=0.056 ms
^C
--- 192.168.0.9 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3049ms
rtt min/avg/max/mdev = 0.049/0.370/1.324/0.550 ms
e-ub@ubuntu:~$ as_ns host3 ping 192.168.0.9

PING 192.168.0.9 (192.168.0.9) 56(84) bytes of data.
64 bytes from 192.168.0.9: icmp_seq=1 ttl=64 time=5.45 ms
64 bytes from 192.168.0.9: icmp_seq=2 ttl=64 time=0.080 ms
64 bytes from 192.168.0.9: icmp_seq=3 ttl=64 time=0.053 ms
64 bytes from 192.168.0.9: icmp_seq=4 ttl=64 time=0.060 ms
64 bytes from 192.168.0.9: icmp_seq=5 ttl=64 time=0.056 ms
^C64 bytes from 192.168.0.9: icmp_seq=6 ttl=64 time=0.057 ms
^C
--- 192.168.0.9 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5092ms
rtt min/avg/max/mdev = 0.053/0.959/5.452/2.009 ms
e-ub@ubuntu:~$ as_ns host5 ping 192.168.2.9

PING 192.168.2.9 (192.168.2.9) 56(84) bytes of data.
64 bytes from 192.168.2.9: icmp_seq=1 ttl=64 time=0.357 ms
64 bytes from 192.168.2.9: icmp_seq=2 ttl=64 time=0.056 ms
64 bytes from 192.168.2.9: icmp_seq=3 ttl=64 time=0.062 ms
64 bytes from 192.168.2.9: icmp_seq=4 ttl=64 time=0.081 ms
64 bytes from 192.168.2.9: icmp_seq=5 ttl=64 time=0.053 ms
^C
--- 192.168.2.9 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4078ms
rtt min/avg/max/mdev = 0.053/0.121/0.357/0.118 ms
e-ub@ubuntu:~$ as_ns host7 ping 192.168.3.9

PING 192.168.3.9 (192.168.3.9) 56(84) bytes of data.
64 bytes from 192.168.3.9: icmp_seq=1 ttl=64 time=1.69 ms
64 bytes from 192.168.3.9: icmp_seq=2 ttl=64 time=0.091 ms
64 bytes from 192.168.3.9: icmp_seq=3 ttl=64 time=0.054 ms
64 bytes from 192.168.3.9: icmp_seq=4 ttl=64 time=0.084 ms
```

This screenshot shows that pings from different hosts work on the trunk link to host 9.



This screenshot shows that initially before we apply the VLAN ACL, pinging works. After we restart Faucet and apply the ACL, ICMP traffic does not go through.

Task Five:

```
e-ub@ubuntu:~$
e-ub@ubuntu:~$ as_ns host1 ip route add default via 10.0.0.254 dev veth0

e-ub@ubuntu:~$
e-ub@ubuntu:~$ as_ns host2 ip route add default via 10.0.1.254 dev veth0

e-ub@ubuntu:~$
e-ub@ubuntu:~$ as_ns host1 ping 10.0.1.2

PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data.
^C
--- 10.0.1.2 ping statistics ---
10 packets transmitted, 0 received, 100% packet loss, time 9209ms

e-ub@ubuntu:~$ sudo nano /etc/faucet/faucet.yaml
e-ub@ubuntu:~$ sudo systemctl reload faucet

e-ub@ubuntu:~$
e-ub@ubuntu:~$ as_ns host1 ping 10.0.1.2

PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data.
64 bytes from 10.0.1.2: icmp_seq=2 ttl=63 time=12.0 ms
64 bytes from 10.0.1.2: icmp_seq=3 ttl=63 time=0.046 ms
64 bytes from 10.0.1.2: icmp_seq=4 ttl=63 time=0.047 ms
64 bytes from 10.0.1.2: icmp_seq=5 ttl=63 time=0.058 ms
^C
--- 10.0.1.2 ping statistics ---
5 packets transmitted, 4 received, 20% packet loss, time 4068ms
rtt min/avg/max/mdev = 0.046/3.035/11.989/5.169 ms
e-ub@ubuntu:~$
```

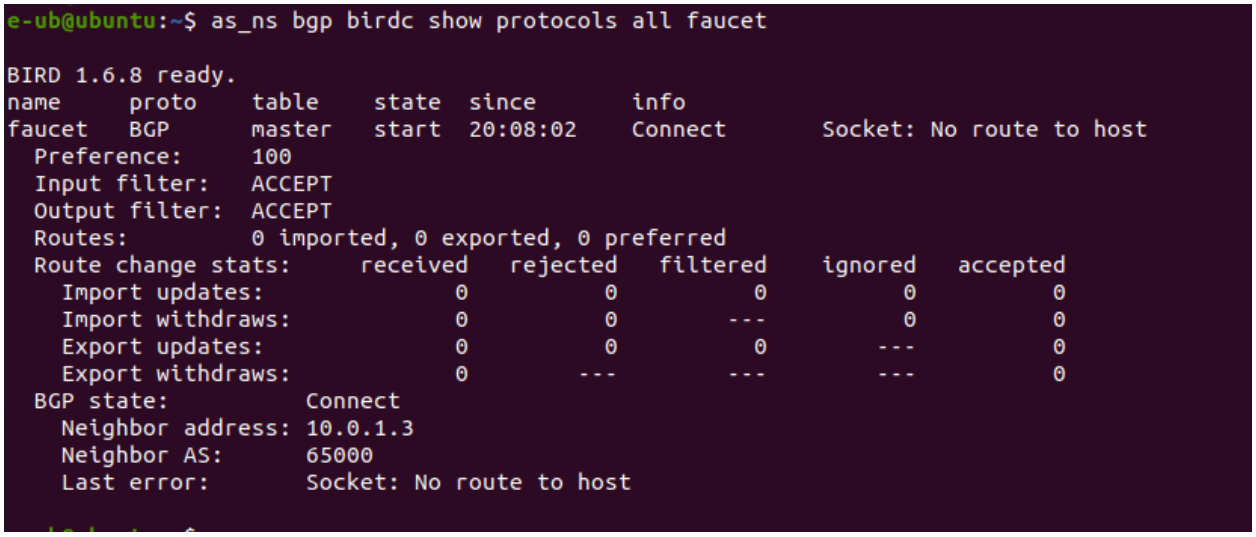
This screenshot shows that host1 and host2 were given default routes and the ping between them failed since they were in different VLANs with different subnets. After enabling inter VLAN routing, the pings work properly.

```
e-ub@ubuntu:~$ as_ns host1 ping 10.0.1.1

PING 10.0.1.1 (10.0.1.1) 56(84) bytes of data.
64 bytes from 10.0.1.1: icmp_seq=2 ttl=63 time=0.220 ms
64 bytes from 10.0.1.1: icmp_seq=3 ttl=63 time=0.052 ms
64 bytes from 10.0.1.1: icmp_seq=4 ttl=63 time=0.059 ms
64 bytes from 10.0.1.1: icmp_seq=5 ttl=63 time=0.043 ms
^C
--- 10.0.1.1 ping statistics ---
5 packets transmitted, 4 received, 20% packet loss, time 4099ms
rtt min/avg/max/mdev = 0.043/0.093/0.220/0.073 ms
e-ub@ubuntu:~$ as_ns server ip address add 192.0.2.1/24 dev veth0

e-ub@ubuntu:~$
e-ub@ubuntu:~$ as_ns host1 ping 192.0.2.1

PING 192.0.2.1 (192.0.2.1) 56(84) bytes of data.
64 bytes from 192.0.2.1: icmp_seq=1 ttl=63 time=0.229 ms
64 bytes from 192.0.2.1: icmp_seq=2 ttl=63 time=0.047 ms
64 bytes from 192.0.2.1: icmp_seq=3 ttl=63 time=0.043 ms
64 bytes from 192.0.2.1: icmp_seq=4 ttl=63 time=0.064 ms
^C
--- 192.0.2.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3052ms
rtt min/avg/max/mdev = 0.043/0.095/0.229/0.077 ms
e-ub@ubuntu:~$
```

This screenshot shows another instance of inter VLAN routing working. It also shows the addition of another IP alias that tests the static routes, proving they work.

```
e-ub@ubuntu:~$ sudo nano /etc/faucet/faucet.yaml
e-ub@ubuntu:~$ sudo systemctl restart faucet
e-ub@ubuntu:~$ as_ns host1 ping 192.0.2.1
PING 192.0.2.1 (192.0.2.1) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
From 10.0.0.1 icmp_seq=4 Destination Host Unreachable
From 10.0.0.1 icmp_seq=5 Destination Host Unreachable
From 10.0.0.1 icmp_seq=6 Destination Host Unreachable
^C
--- 192.0.2.1 ping statistics ---
7 packets transmitted, 0 received, +6 errors, 100% packet loss, time 6150ms
pipe 4
```

This screenshot shows that we can not ping the address we were previously statically routing.

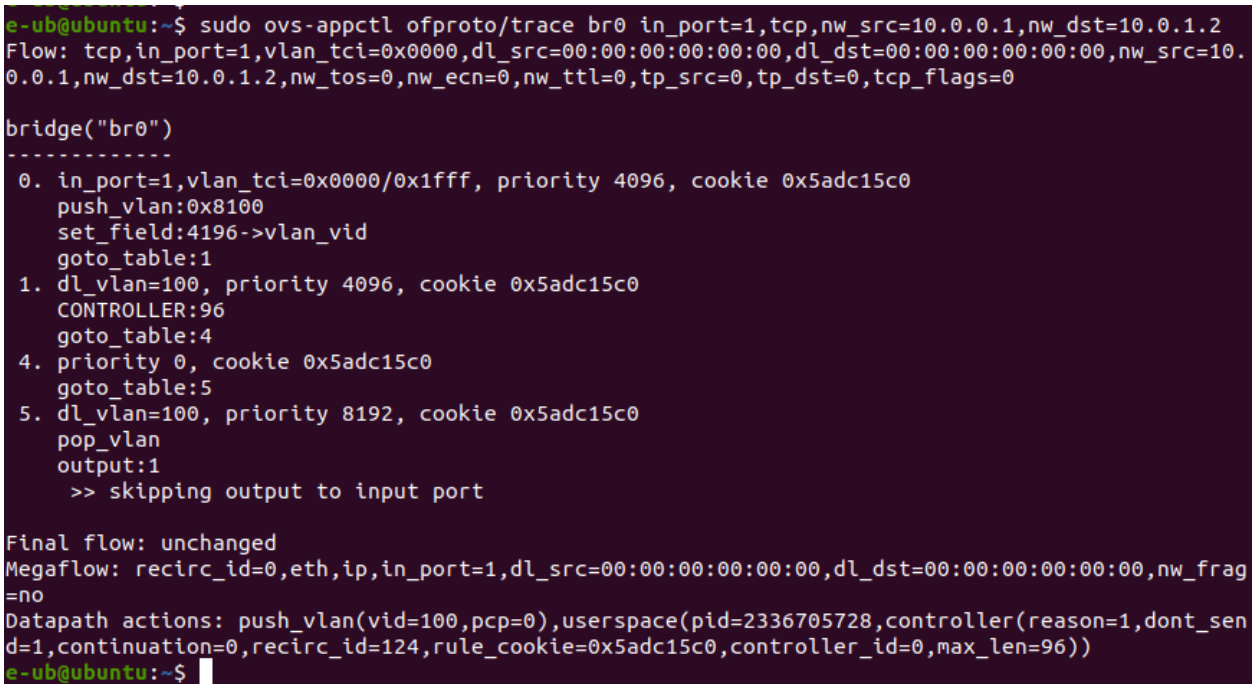


Unfortunately, I was not able to get the BGP portion of task 5 working. I followed the tutorial closely, but nothing seemed to work. I suspect it might have been a YAML syntax issue but I am not completely sure. I kept getting a no route to host error but am sure I set one properly. This screenshot shows output from the “as_ns bgp birdc show protocols all faucet” command.

Task Six:

```
e-ub@ubuntu:~$
e-ub@ubuntu:~$ as_ns host1 ip route add default via 10.0.0.254 dev veth0

e-ub@ubuntu:~$
e-ub@ubuntu:~$ as_ns host2 ip route add default via 10.0.1.254 dev veth0

e-ub@ubuntu:~$
e-ub@ubuntu:~$ as_ns host1 ping 10.0.1.2

PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data.
64 bytes from 10.0.1.2: icmp_seq=2 ttl=63 time=1.87 ms
64 bytes from 10.0.1.2: icmp_seq=3 ttl=63 time=0.039 ms
64 bytes from 10.0.1.2: icmp_seq=4 ttl=63 time=0.049 ms
64 bytes from 10.0.1.2: icmp_seq=5 ttl=63 time=0.048 ms
^C
--- 10.0.1.2 ping statistics ---
5 packets transmitted, 4 received, 20% packet loss, time 4047ms
rtt min/avg/max/mdev = 0.039/0.501/1.871/0.790 ms
e-ub@ubuntu:~$ as_ns host2 ping 10.0.0.1

PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=63 time=0.043 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=63 time=0.049 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=63 time=0.051 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=63 time=0.047 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=63 time=0.047 ms
^C
--- 10.0.0.1 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4093ms
rtt min/avg/max/mdev = 0.043/0.047/0.051/0.002 ms
e-ub@ubuntu:~$
```

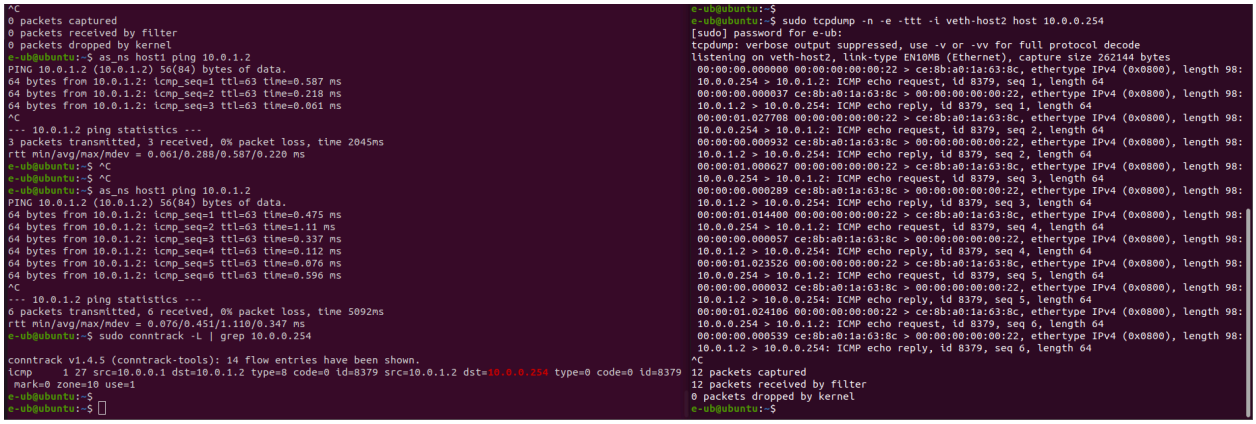
This screenshot shows successful pings to hosts one and two after the default routes were added.



This screenshot shows the output of “sudo ovs-appctl ofproto/trace br0 in_port=1,tcp,nw_src=10.0.0.1,nw_dst=10.0.1.2” showing how the packet was processed through VLAN tagging, through some flow tables, forwarded via the controller to reach the final destination. This proves that the Faucet rules are functioning correctly.

```
e-ub@ubuntu:~$ as_ns host2 ping 10.0.1.2
PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data.
64 bytes from 10.0.1.2: icmp_seq=1 ttl=64 time=0.072 ms
64 bytes from 10.0.1.2: icmp_seq=2 ttl=64 time=0.066 ms
64 bytes from 10.0.1.2: icmp_seq=3 ttl=64 time=0.024 ms
64 bytes from 10.0.1.2: icmp_seq=4 ttl=64 time=0.041 ms
^C
--- 10.0.1.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3072ms
rtt min/avg/max/mdev = 0.024/0.050/0.072/0.019 ms
e-ub@ubuntu:~$ sudo conntrack -L | grep 10.0.1.2

conntrack v1.4.5 (conntrack-tools): 23 flow entries have been shown.
e-ub@ubuntu:~$
e-ub@ubuntu:~$ sudo ovs-appctl ofproto/trace br0 in_port=1,tcp,nw_src=10.0.1.2,nw_dst=10.0.0.1
Flow: tcp,in_port=1,vlan_tci=0x0000,dl_src=00:00:00:00:00:00,dl_dst=00:00:00:00:00:00,nw_src=10.
0.1.2,nw_dst=10.0.0.1,nw_tos=0,nw_ecn=0,nw_ttl=0,tp_src=0,tp_dst=0,tcp_flags=0

bridge("br0")
-------------
 0. in_port=1,vlan_tci=0x0000/0x1fff, priority 4096, cookie 0x5adc15c0
    push_vlan:0x8100
    set_field:4196->vlan_vid
    goto_table:1
 1. dl_vlan=100, priority 4096, cookie 0x5adc15c0
    CONTROLLER:96
    goto_table:4
 4. priority 0, cookie 0x5adc15c0
    goto_table:5
 5. dl_vlan=100, priority 8192, cookie 0x5adc15c0
    pop_vlan
    output:1
     >> skipping output to input port

Final flow: unchanged
Megaflow: recirc_id=0,eth,ip,in_port=1,dl_src=00:00:00:00:00:00,dl_dst=00:00:00:00:00:00,nw_frag
=no
Datapath actions: push_vlan(vid=100,pcp=0),userspace(pid=2336705728,controller(reason=1,dont_ser
d=1,continuation=0,recirc_id=154,rule_cookie=0x5adc15c0,controller_id=0,max_len=96))
e-ub@ubuntu:~$ as_ns host2 ping 10.0.0.1

PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
From 10.0.1.2 icmp_seq=1 Destination Host Unreachable
From 10.0.1.2 icmp_seq=2 Destination Host Unreachable
From 10.0.1.2 icmp_seq=3 Destination Host Unreachable
```

This screenshot shows that ping traffic is not being let through, proving that the conntrack ACL is denying new inbound connections from host2, while still allowing return traffic and new outbound connections from host1.



This screenshot shows that the ping from host1 to host2 still works after applying source NAT using Faucet. The traffic was captured on tcpdump, and confirms that NAT is working properly. The conntrack output also shows an active connection with the NATed source address.

Conclusion:

I liked the tutorials that were included in this lab. There were a lot of screenshots that needed to be taken to show verification. The only part I struggled with was the BGP part in task 5, I was not able to get the right results. The rest of the lab went pretty smoothly though.