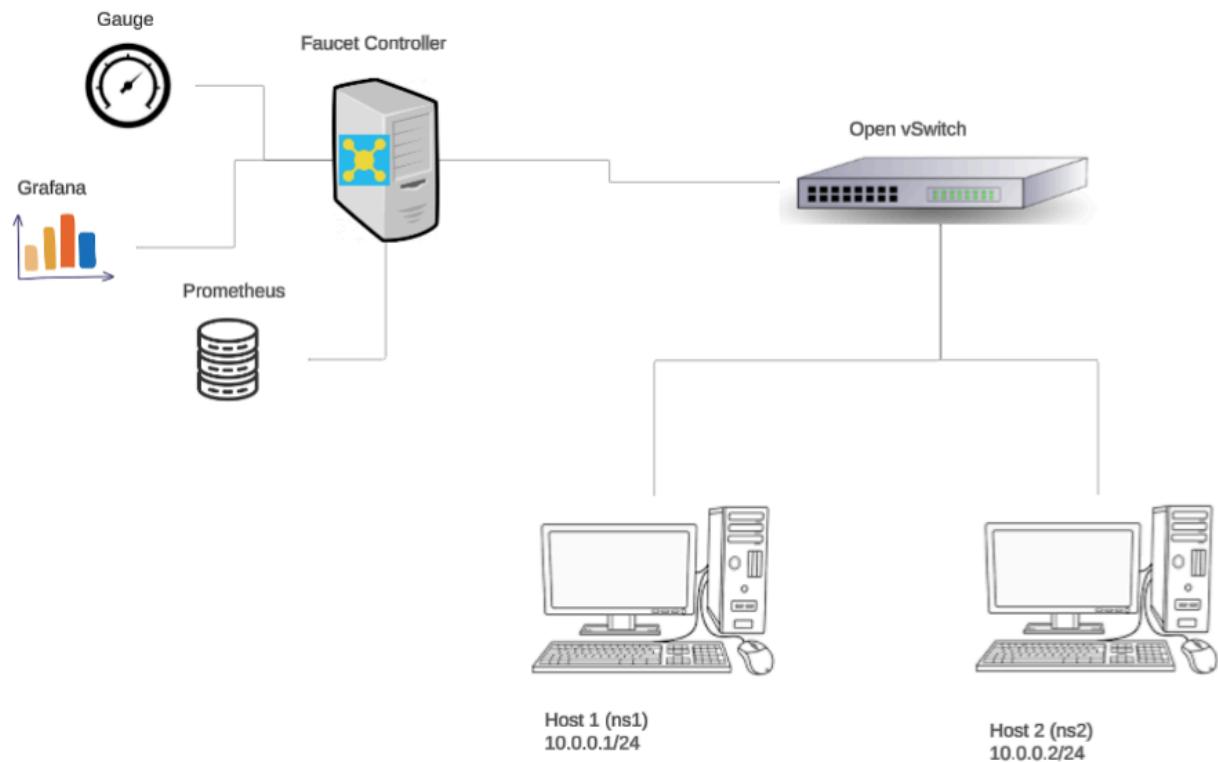


## Lab 6

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

### Topology:



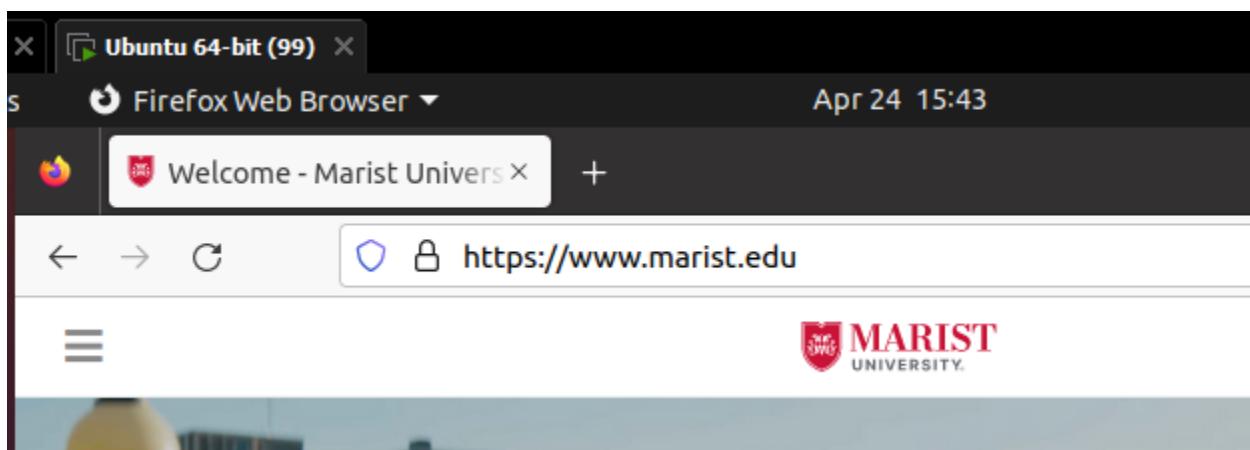
### Syntax:

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

```
e-ub@ubuntu:~$ sudo apt-get install faucet-all-in-one
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
daemon docutils-common faucet gauge grafana ieee-data javascript-common libexpat1-dev libjs-bootstrap libjs-d3
libjs-eonasdan-bootstrap-datetimepicker libjs-jquery libjs-jquery-hotkeys libjs-moment libjs-mustache
libjs-rickshaw libpython3-dev libpython3.5 libpython3.5-dev libpython3.5-minimal libpython3.5-stdlib prometheus
prometheus-node-exporter python-babel-locatedata python-pip-whl python3-babel python3-beka python3-certifi
python3-chardet python3-chevie python3-crypto python3-dateutil python3-debtcollector python3-decorator
python3-dev python3-dnspython python3-docutils python3-ecdsa python3-eventlet python3-faucet python3-greenlet
python3-influxdb python3-monotonic python3-msgpack python3-netaddr python3-networkx python3-openvswitch
python3-oslo.config python3-paramiko python3-pbr python3-pip python3-prometheus-client python3-pygments
python3-pytricia python3-repoze.lru python3-requests python3-roman python3-routes python3-ryu python3-setuptools
python3-sortedcontainers python3-stevedore python3-tinyrpc python3-transitions python3-tz python3-urllib3
python3-webob python3-wheel python3-wrapt python3-yaml python3.5 python3.5-dev python3.5-minimal
Suggested packages:
python-faucet-doc apache2 | lighttpd | httpd python3-crypto-dbg python-crypto-doc python-debtcollector-doc
texlive-latex-recommended texlive-latex-base texlive-lang-french fonts-linuxlibertine | ttf-linux-libertine
docutils-doc python-eventlet-doc python-greenlet-doc python-greenlet-dev python3-greenlet-dbg ipython3
python-netaddr-docs python-networkx-doc ttf-bitstream-vera python3-idna python3-openssl python3-socks
python3-paste python-ryu-doc python-setuptools-doc python-sortedcontainers-doc python-tinyrpc-doc
python-webob-doc python3.5-venv python3.5-doc binfmt-support
The following NEW packages will be installed:
daemon docutils-common faucet faucet-all-in-one gauge grafana ieee-data javascript-common libexpat1-dev
libjs-bootstrap libjs-d3 libjs-eonasdan-bootstrap-datetimepicker libjs-jquery libjs-jquery-hotkeys libjs-moment
libjs-mustache libjs-rickshaw libpython3-dev libpython3.5-dev prometheus prometheus-node-exporter
python-babel-locatedata python-pip-whl python3-babel python3-beka python3-certifi python3-chevie python3-crypto
```

This screenshot shows that Faucet was installed successfully.

The screenshot shows the Prometheus configuration page at `localhost:9090/config`. The page has a dark header with navigation links for Prometheus, Alerts, Graph, Status, and Help. Below the header, the title "Configuration" is displayed, followed by a blue button labeled "Copy to clipboard". The main content area contains the following YAML configuration:

```
global:
  scrape_interval: 15s
  scrape_timeout: 10s
  evaluation_interval: 15s
rule_files:
- /etc/faucet/prometheus/faucet.rules.yml
scrape_configs:
- job_name: prometheus
  honor_timestamps: true
  scrape_interval: 15s
  scrape_timeout: 10s
  metrics_path: /metrics
  scheme: http
  static_configs:
  - targets:
    - localhost:9090
- job_name: faucet
  honor_timestamps: true
  scrape_interval: 15s
  scrape_timeout: 10s
  metrics_path: /metrics
  scheme: http
  static_configs:
  - targets:
    - localhost:9302
- job_name: gauge
  honor_timestamps: true
  scrape_interval: 15s
  scrape_timeout: 10s
  metrics_path: /metrics
  scheme: http
  static_configs:
  - targets:
    - localhost:9303
```

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

The screenshot shows the Prometheus /targets view page. The browser address bar displays "localhost:9090/targets". The page header includes links for Prometheus, Alerts, Graph, Status, and Help. The main section is titled "Targets" and has two tabs: "All" (selected) and "Unhealthy".

**faucet (1/1 up)** [show less]

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="http://localhost:9302/metrics">http://localhost:9302/metrics</a>	UP	instance="localhost:9302" job="faucet"	14.19s ago	3.425ms	

**gauge (1/1 up)** [show less]

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="http://localhost:9303/metrics">http://localhost:9303/metrics</a>	UP	instance="localhost:9303" job="gauge"	12.779s ago	1.459ms	

**prometheus (1/1 up)** [show less]

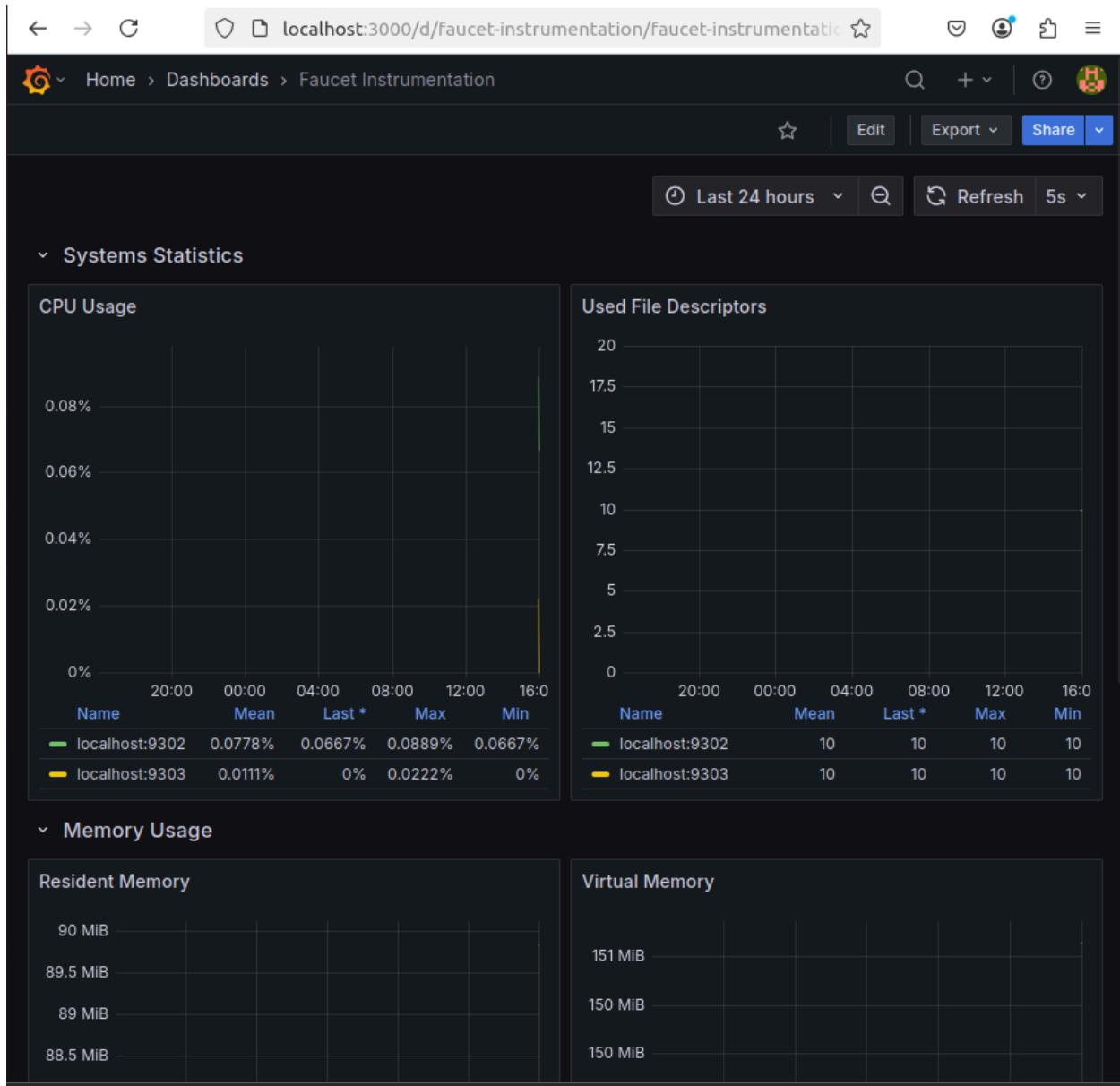
Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<a href="http://localhost:9090/metrics">http://localhost:9090/metrics</a>	UP	instance="localhost:9090" job="prometheus"	10.506s ago	7.743ms	

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

The screenshot shows the Grafana interface for managing dashboards. At the top, there's a navigation bar with a gear icon, "Home", and "Dashboards". On the right side of the header are search, add, help, and user icons. Below the header, the title "Dashboards" is displayed, followed by a "New" button with a dropdown arrow. A sub-instruction "Create and manage dashboards to visualize your data" is present. There's a search bar labeled "Search for dashboards and folders" and a filter section with "Filter by tag" (set to "FAUCET"), "Starred" (unchecked), and "Sort" (dropdown menu). The main area lists three dashboards:

Name	Tags
Faucet Instrumentation	FAUCET
Faucet Inventory	FAUCET
Faucet Port Statistics	FAUCET

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



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

```
e-ub@ubuntu:~$ check_faucet_config /etc/faucet/faucet.yaml
```

```
[ '{\n'
  '  "advertise_interval": 30,\n'
  '  "arp_neighbor_timeout": 30,\n'
  '  "cache_update_guard_time": 150,\n'
  '  "combinatorial_port_flood": false,\n'
  '  "cookie": 1524372928,\n'
  '  "description": "sw1",\n'
  '  "dot1x": {},\n'
  '  "dp_id": 1,\n'
  '  "drop_broadcast_source_address": true,\n'
  '  "drop_spoofed_faucet_mac": true,\n'
  '  "egress_pipeline": false,\n'
  '  "fast_advertise_interval": 5,\n'
  '  "faucet_dp_mac": "0e:00:00:00:00:01",\n'
  '  "global_vlan": 0,\n'
  '  "group_table": false,\n'
  '  "hardware": "Open vSwitch",\n'
  '  "high_priority": 9001,\n'
  '  "highest_priority": 9099,\n'
  '  "idle_dst": true,\n'
  '  "ignore_learn_ins": 10,\n'
  '  "interface_ranges": {},\n'
  '  "interfaces": {\n'
    '    "1": {\n'
      '      "description": "host1 network namespace",\n'
      '      "name": "host1",\n'
      '      "native_vlan": "office"\n'
    },\n'
    '    "2": {\n'
      '      "description": "host2 network namespace",\n'
      '      "name": "host2",\n'
      '      "native_vlan": "office"\n'
    },\n'
  },\n'
  "lacp_timeout": 30,\n'
  "learn_ban_timeout": 51,\n'
  "learn_jitter": 51,\n'
  "lldp_beacon": {},\n'
  "low_priority": 9000,\n'
  "lowest_priority": 0,\n'
  "max_host_fib_retry_count": 10,\n'
  "max_hosts_per_resolve_cycle": 5,\n'
```

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 differences
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:fe00:1001/64']
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.valve 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 differences
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 are 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:~$ 
```

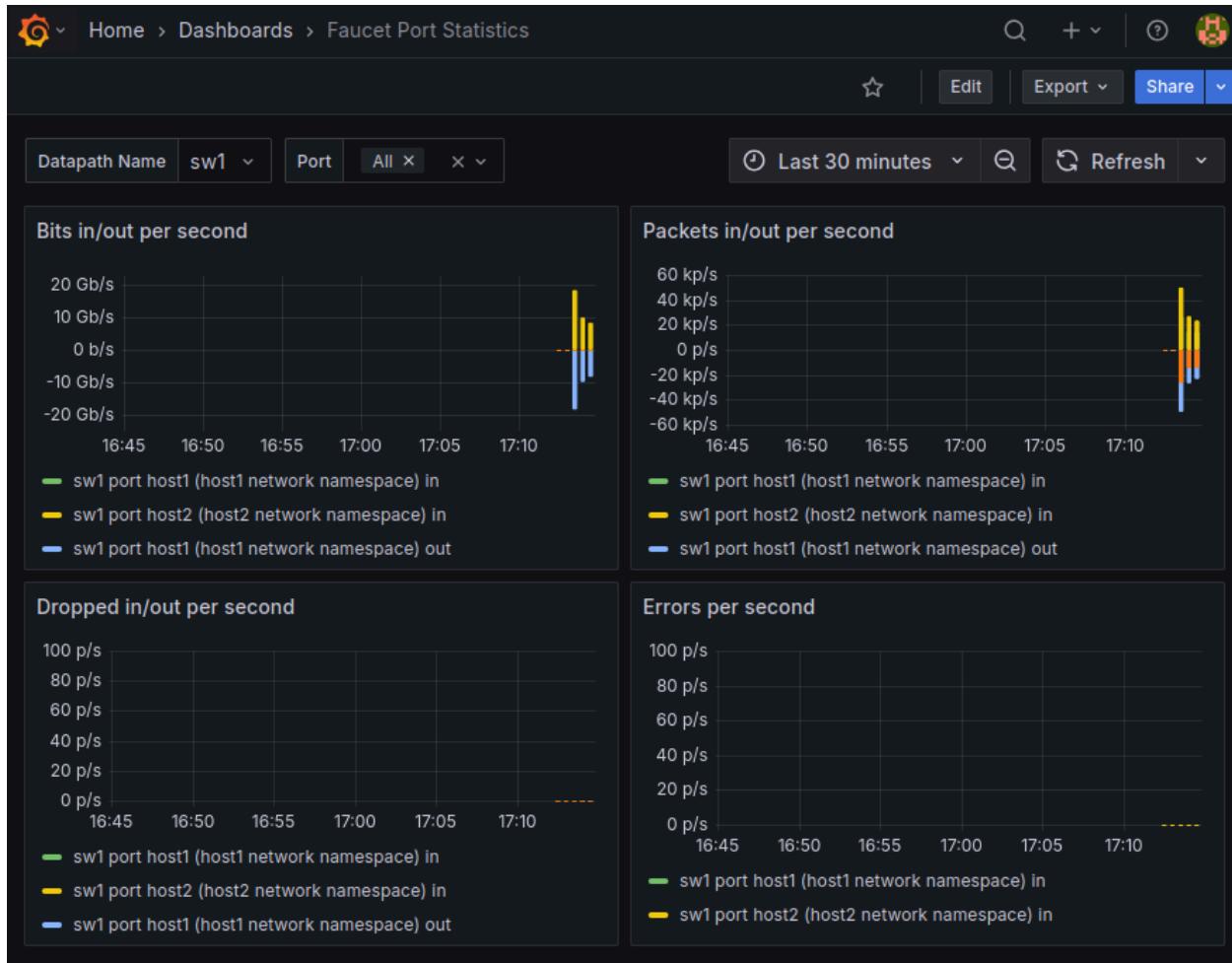
This screenshot shows the ovs-vsctl show command and that my controllers were connected successfully.

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

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.

```

tcpdump: verbose output suppressed, type -v or -vv for full protocol decode
listening on wlo, link-type EN10MB (Ethernet), capture size 262144 bytes
17:35:55.773673 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:35:56.775430 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:35:57.814748 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:35:58.839738 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:36:00.878819 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:36:01.910781 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:36:02.935374 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:36:03.959150 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:36:04.983375 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:36:06.007098 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:36:07.030923 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv4 (0x0800), length 98: 192.168.0.3 > 192.168.0.1: ICMP echo r
17:36:08.054146 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv6 (0x86dd), length 303: fe80::48d4:ffff%fe23:3b65_E53>ff02::1:IC
17:36:09.079373 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv6 (0x86dd), length 70: fe80::4d3d:ffff%fe38:74ab>ff02::2:I
17:38:00.007937 0:0:0:0:0:0>e2:2a:c6:06:72:3b, ethertype IPv6 (0x86dd), length 70: fe80::5c6e:fbff%fe3d:467>ff02::2:IC
MP6, router solicitation, length 16
MP6, router solicitation, length 16
16 packets captured
16 packets received by filter
0 packets dropped by kernel
-e-ubuntu: ~ $ sudo nano /etc/faucet/faucet.yaml
-e-ubuntu: ~ $
```

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.

```

e-ub@ubuntu:~$ sudo systemctl reload faucet
e-ub@ubuntu:~$ as_ns host4 tcpdump -l -e -n -i veth0
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on veth0, link-type EN10MB (Ethernet), capture size 202144 bytes
17:52:11.065725 e2:2a:c6:06:72:3b > 5e:0e:fb:d3:04:67, ethertype IPv4 (0x8000), length 98: 192.168.0.1 > 192.16
8.0.3: ICMP echo request, id 190, seq 1, length 64
17:52:12.065801 e2:2a:c6:06:72:3b > 5e:0e:fb:d3:04:67, ethertype ARP (0x8006), length 64: 192.168.0.3 > 192.168.0.1, ethe
rtype ARP, 192.168.0.3 > 192.168.0.1, length 64
17:52:12.087075 e2:2a:c6:06:72:3b > 5e:0e:fb:d3:04:67, ethertype IPv4 (0x8000), length 98: 192.168.0.1 > 192.16
8.0.3: ICMP echo request, id 190, seq 2, length 64
17:52:12.087104 5e:0e:fb:d3:04:67 > e2:2a:c6:06:72:3b, ethertype 802.1Q (0x8100), length 102: vlan 3, p 0, ethe
rtype IPv4, 192.168.0.3 > 192.168.0.1: ICMP echo reply, id 190, seq 1, length 64
17:52:13.110755 e2:2a:c6:06:72:3b > 5e:0e:fb:d3:04:67, ethertype IPv4 (0x8000), length 98: 192.168.0.1 > 192.16
8.0.3: ICMP echo request, id 190, seq 3, length 64
17:52:13.110787 5e:0e:fb:d3:04:67 > e2:2a:c6:06:72:3b, ethertype 802.1Q (0x8100), length 102: vlan 3, p 0, ethe
rtype IPv4, 192.168.0.3 > 192.168.0.1: ICMP echo reply, id 190, seq 3, length 64
17:52:14.134844 e2:2a:c6:06:72:3b > 5e:0e:fb:d3:04:67, ethertype IPv4 (0x8000), length 98: 192.168.0.1 > 192.16
8.0.3: ICMP echo request, id 190, seq 4, length 64
17:52:14.134881 5e:0e:fb:d3:04:67 > e2:2a:c6:06:72:3b, ethertype 802.1Q (0x8100), length 102: vlan 3, p 0, ethe
rtype IPv4, 192.168.0.3 > 192.168.0.1: ICMP echo reply, id 190, seq 4, length 64
17:52:15.159015 e2:2a:c6:06:72:3b > 5e:0e:fb:d3:04:67, ethertype IPv4 (0x8000), length 98: 192.168.0.1 > 192.16
8.0.3: ICMP echo request, id 190, seq 5, length 64
17:52:15.159047 5e:0e:fb:d3:04:67 > e2:2a:c6:06:72:3b, ethertype 802.1Q (0x8100), length 102: vlan 3, p 0, ethe
rtype IPv4, 192.168.0.3 > 192.168.0.1: ICMP echo reply, id 190, seq 5, length 64
17:52:16.183561 e2:2a:c6:06:72:3b > 5e:0e:fb:d3:04:67, ethertype IPv4 (0x8000), length 98: 192.168.0.1 > 192.16
8.0.3: ICMP echo request, id 190, seq 6, length 64
17:52:16.183611 5e:0e:fb:d3:04:67 > e2:2a:c6:06:72:3b, ethertype 802.1Q (0x8100), length 102: vlan 3, p 0, ethe
rtype IPv4, 192.168.0.3 > 192.168.0.1: ICMP echo reply, id 190, seq 6, length 64
17:52:16.183815 e2:2a:c6:06:72:3b > 5e:0e:fb:d3:04:67, ethertype ARP (0x8006), length 42: Request who-has 192.1
68.0.3 tell 192.168.0.1, length 28
17:52:16.187003 e2:2a:c6:06:72:3b > 5e:0e:fb:d3:04:67, ethertype ARP (0x8006), length 42: Reply 192.168.0.1 is-
at 2:2a:c6:06:72:3b, length 28
e-ub@ubuntu:~$ as_ns host1 ping 192.168.0.3
[sudo] password for e-ub:
PING 192.168.0.3 (192.168.0.3) 56(84) bytes of data.
^C
--- 192.168.0.3 ping statistics ---
12 packets transmitted, 0 received, 100% packet loss, time 11258ms
e-ub@ubuntu:~$ as_ns host2 ping 192.168.0.1
[sudo] password for e-ub:
PING 192.168.0.1 (192.168.0.1) 56(84) bytes of data.
^C
--- 192.168.0.1 ping statistics ---
12 packets transmitted, 0 received, 100% packet loss, time 11258ms
e-ub@ubuntu:~$ as_ns host1 ping 192.168.0.3
[sudo] password for e-ub:
PING 192.168.0.3 (192.168.0.3) 56(84) bytes of data.
^C
--- 192.168.0.3 ping statistics ---
64 bytes from 192.168.0.1: icmp_seq=1 ttl=64 time=0.064 ms
64 bytes from 192.168.0.1: icmp_seq=2 ttl=64 time=0.056 ms
64 bytes from 192.168.0.1: icmp_seq=3 ttl=64 time=0.053 ms
64 bytes from 192.168.0.1: icmp_seq=4 ttl=64 time=0.056 ms
64 bytes from 192.168.0.1: icmp_seq=5 ttl=64 time=0.070 ms
64 bytes from 192.168.0.1: icmp_seq=6 ttl=64 time=0.056 ms
64 bytes from 192.168.0.1: icmp_seq=7 ttl=64 time=0.051 ms
64 bytes from 192.168.0.1: icmp_seq=8 ttl=64 time=0.090 ms
64 bytes from 192.168.0.1: icmp_seq=9 ttl=64 time=0.116 ms
64 bytes from 192.168.0.1: icmp_seq=10 ttl=64 time=0.073 ms
64 bytes from 192.168.0.1: icmp_seq=11 ttl=64 time=0.068 ms
64 bytes from 192.168.0.1: icmp_seq=12 ttl=64 time=0.093 ms
64 bytes from 192.168.0.1: icmp_seq=13 ttl=64 time=0.078 ms
e-ub@ubuntu:~$
```

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.

```
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=0.262 ms
64 bytes from 192.168.3.8: icmp_seq=2 ttl=64 time=0.080 ms
64 bytes from 192.168.3.8: icmp_seq=3 ttl=64 time=0.057 ms
64 bytes from 192.168.3.8: icmp_seq=4 ttl=64 time=0.055 ms
^C
--- 192.168.3.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3046ms
rtt min/avg/max/mdev = 0.055/0.113/0.262/0.086 ms
e-ub@ubuntu:~$ sudo systemctl reload faucet

e-ub@ubuntu:~$
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.
^C
--- 192.168.3.8 ping statistics ---
16 packets transmitted, 0 received, 100% packet loss, time 15341ms
```

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.

```
e-ub@ubuntu:~$ as_ns bgp birdc show protocols all faucet

BIRD 1.6.8 ready.
name    proto   table   state   since      info
faucet  BGP     master  start  20:08:02  Connect      Socket: No route to host
Preference: 100
Input filter: ACCEPT
Output filter: ACCEPT
Routes:      0 imported, 0 exported, 0 preferred
Route change stats: received rejected filtered ignored accepted
  Import updates:      0        0        0        0        0
  Import withdraws:    0        0        ---      0        0
  Export updates:     0        0        0        ---      0
  Export withdraws:   0        ---      ---      ---      0
BGP state:      Connect
Neighbor address: 10.0.1.3
Neighbor AS:    65000
Last error:     Socket: No route to host
```

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.

```
e-ub@ubuntu:~$ sudo ovs-appctl ofproto/trace br0 in_port=1,tcp,nw_src=10.0.0.1,nw_dst=10.0.1.2
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.
0.0.1,nw_dst=10.0.1.2,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_sen
d=1,continuation=0,recirc_id=124,rule_cookie=0x5adc15c0,controller_id=0,max_len=96))
e-ub@ubuntu:~$
```

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_send=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.

```

^C
0 packets captured
0 packets received by filter
0 packets dropped by kernel
^C
e-ubuntu:~$ ns ns host ping 10.0.1.2
PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data.
40 bytes from 10.0.1.2: icmp_seq=1 ttl=63 time=0.587 ms
40 bytes from 10.0.1.2: icmp_seq=2 ttl=63 time=0.218 ms
40 bytes from 10.0.1.2: icmp_seq=3 ttl=63 time=0.061 ms
...
--> 10.0.1.2 ping statistics --
3 packets transmitted, 3 received, 0% packet loss, time 2045ms
rtt min/avg/mdev = 0.061/0.288/0.587/0.220 ms
e-ubuntu:~$ ^C
e-ubuntu:~$ ns ns host ping 10.0.1.2
PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data.
40 bytes from 10.0.1.2: icmp_seq=1 ttl=63 time=0.475 ms
40 bytes from 10.0.1.2: icmp_seq=2 ttl=63 time=0.237 ms
40 bytes from 10.0.1.2: icmp_seq=3 ttl=63 time=0.337 ms
40 bytes from 10.0.1.2: icmp_seq=4 ttl=63 time=0.112 ms
40 bytes from 10.0.1.2: icmp_seq=5 ttl=63 time=0.076 ms
40 bytes from 10.0.1.2: icmp_seq=6 ttl=63 time=0.596 ms
...
--> 10.0.1.2 ping statistics --
6 packets transmitted, 6 received, 0% packet loss, time 5092ms
rtt min/avg/mdev = 0.076/0.451/0.119/0.347 ms
e-ubuntu:~$ sudo contrack -L | grep 10.0.0.254
contrack v1.4.5 (contrack-tools): 14 flow entries have been shown.
icmp      1 27 return:10.0.0.1 dst=10.0.1.2 type#8 code#0 id=8379 src=10.0.1.2 dst=10.0.0.254 type#0 code#0 id=8379
mark#0 zone=10 use=1
e-ubuntu:~$ ^C
e-ubuntu:~$ [sudo] password for e-ubuntu:
e-ubuntu:~$ sudo tcpdump -n -e -ttt -i veth-host2 host 10.0.0.254
[sudo] password for e-ubuntu:
tcpdump: interface 'veth-host2' not found, suppressed, use -v or -vv for full protocol decode
listening on veth-host2, link-type EN16BHP (Ethernet), capture size 262144 bytes
00:00:00:00:00:00 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo request, id 8379, seq 1, length 64
00:00:00:00:00:00 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo reply, id 8379, seq 1, length 64
00:00:00:00:00:00 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo reply, id 8379, seq 1, length 64
00:00:01:01:27 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo request, id 8379, seq 2, length 64
00:00:00:00:00:00 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo reply, id 8379, seq 2, length 64
00:00:01:00:00:07 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo request, id 8379, seq 3, length 64
00:00:00:00:28 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo reply, id 8379, seq 3, length 64
00:00:01:01:44 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo request, id 8379, seq 4, length 64
00:00:00:00:00:07 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo reply, id 8379, seq 4, length 64
00:00:01:02:35 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo request, id 8379, seq 5, length 64
00:00:00:00:00:02 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo reply, id 8379, seq 5, length 64
00:00:00:00:00:09 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo request, id 8379, seq 6, length 64
00:00:00:00:00:09 00:00:00:00:00:22 > :ce:8b:a0:1a:63:8c, ethertype IPv4 (0x0800), length 98: 10.0.1.2> 10.0.1.2: ICMP echo reply, id 8379, seq 6, length 64
e-ubuntu:~$ ^C
12 packets captured
12 packets received by filter
0 packets dropped by kernel
e-ubuntu:~$ ^C

```

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.