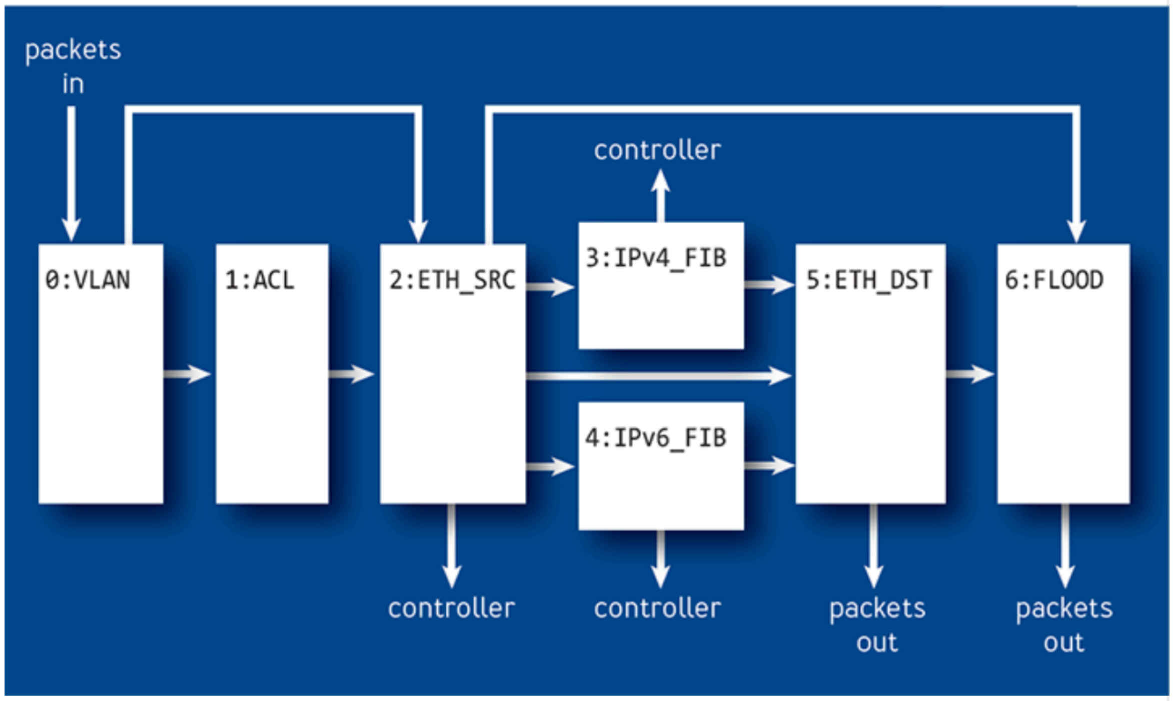
Faucet Guide

What is Faucet? (Notes from the paper introducing Faucet at https://queue.acm.org/detail.cfm?id=3015763)

- Faucet is an open source OpenFlow controller that implements OpenFlow protocol 1.3
- OpenFlow protocol enables software defined networking (SDN)
- Allows you to treat networks like mutable software as opposed to immutable infrastructure
- Add new network services and patch bugs dynamically
- Faucet is an OpenFlow SDN controller for enterprise
- Faucet is deployed as a unit of two systems: a controller host and an OpenFlow switch, which are connected.
- Controller is defined by a config file (.yaml) which defines which VLAN (Virtualised Local Area Network) each port of the switch is connected to.
- Installation process only takes a minute.
- Faucet is lightweight. Because the switch hardware does the packet forwarding, Faucet Controller can be deployed on a Raspberry Pi.
- Faucet software stack includes a unit-testing framework so that tests can be run against simulated (mininet) and real (hardware) switches.

FIGURE 2: **FAUCET'S ALL OPENFLOW PIPELINE**



Why Faucet?

Design

- Faucet is small and simple (1000s of lines of code vs millions)
- Faucet doesn’t need implementation-specific driver code
- Faucet does not need connectivity to external databases to generate forwarding decisions
- Faucet scales well through replication

Performance and scaling

- Faucet offloads all forwarding to the OpenFlow switch.
- Faucet programs the switch pre-emptively.
- Faucet includes multiple flow tables for different types of packet.
- Faucet can be deployed on a large network via replication

Testing

- Faucet follows bes

Configuring faucet

- Faucet is configured via .yaml files.
- Here’s an example of a .yaml config file for a simple network topology consisting of a single switch with two ports:
- Note: Edit the faucet config file at /etc/faucet/faucet.yaml
- Note: No tabs in YAML files, only spaces (yuck!)

```
vlans:
    office:
        vid: 100
        description: "office network"

dps:
    sw1:
        dp_id: 0x1
        hardware: "Open vSwitch"
        interfaces:
            1:
                name: "host1"
                description: "host1 network namespace"
                native_vlan: office
            2:
                name: "host2"
                description: "host2 network namespace"
                native_vlan: office
```

- This creates a single VLAN and a single switch(datapath) with two

ports.
- Check you have defined a valid configuration with check_faucet_config:

```
# check_faucet_config /etc/faucet/faucet.yaml
```

- Returns the JSON object containing the faucet information if successful.
- Otherwise returns an error.
- Now reload faucet to let the changes propagate

```
# sudo systemctl reload faucet
```

- Check logs at /var/log/faucet/faucet.log to make sure the config was loaded successfully.

Connecting a first datapath/switch

- Here's an example connecting a Open vSwitch to the faucet instance.
- Install openvswitch:

```
# sudo apt-get install openvswitch-switch
```

- Add 2 linux network namespace to simulate hosts inside of
- Define useful bash functions to create and run commands inside of network namespace

```
# Run command inside network namespace
as_ns () {
    NAME=$1
    NETNS=faucet-${NAME}
    shift
    sudo ip netns exec ${NETNS} $@
}

# Create network namespace
create_ns () {
    NAME=$1
    IP=$2
    NETNS=faucet-${NAME}
    sudo ip netns add ${NETNS}
    sudo ip link add dev veth-${NAME} type veth peer
name veth0 netns ${NETNS}
    sudo ip link set dev veth-${NAME} up
    as_ns ${NAME} ip link set dev lo up
    [ -n "${IP}" ] && as_ns ${NAME} ip addr add dev
veth0 ${IP}
    as_ns ${NAME} ip link set dev veth0 up
}
```

- Create host1 and host2 and assign them ip addresses.

```
# create_ns host1 192.168.0.1/24
# create_ns host2 192.168.0.2/24
```

- configure Open vSwitch bridge (our switch) and add two ports

```
# sudo ovs-vsctl add-br br0 \
-- set bridge br0 other-config:datapath-
id=0000000000000001 \
-- set bridge br0 other-config:disable-in-band=true \
-- set bridge br0 fail_mode=secure \
-- add-port br0 veth-host1 -- set interface veth-host1
ofport_request=1 \
-- add-port br0 veth-host2 -- set interface veth-host2
ofport_request=2 \
-- set-controller br0 tcp:127.0.0.1:6653
tcp:127.0.0.1:6654
```

- Test network by getting h1 to ping h2

```
# as_ns host1 ping 192.168.0.2
```

- Traffic should then show up in /var/log/faucet/faucet.log