What is FAUCET?

Brad Cowie
University of Waikato
New Zealand

Faucet

History Lesson

- Brief history on SDN activities in New Zealand

RouteFlow (Dec 2011 - April 2016)

- SDN work at the time was focused on the data centre
- RouteFlow was SDN for the WAN
- Push routes from Linux routing table onto OpenFlow switches
- Let us turn a set of switches into distributed router
- Core technology behind
  - Project W
  - VANDERVECKEN
  - CARDIGAN
- Many issues
  - Too many components, very complicated to debug
  - Prone to crashes due to race conditions

Valve (March 2014 - September 2016)

- “Hello world” app for SDN is a layer 2 learning switch
- Valve was a layer 2 learning switch with useful features
  - Simple YAML configuration file
  - VLAN support
  - Access Control Lists (ACLs)
  - Multi-datapath support
  - Statistics
- Written as research project at WAND Group, University of Waikato
- Very small and simple, ~500 lines of code
- Not production ready
  - MAC learning rules don’t expire

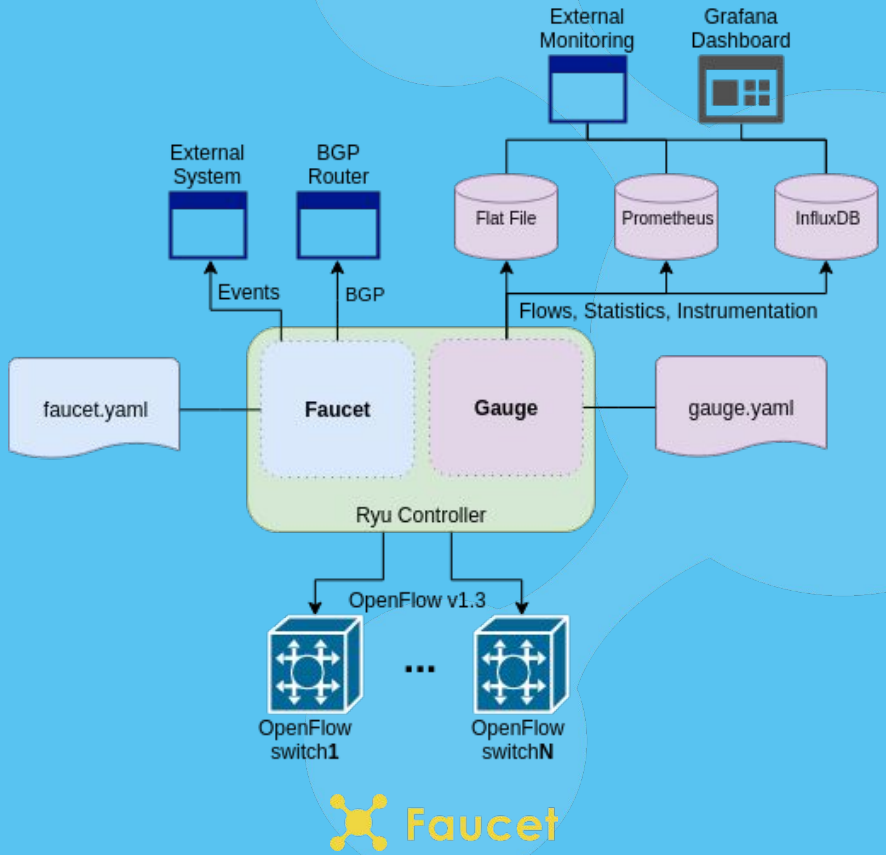
FAUCET (August 2015 - Present)

- REANNZ took Valve and built FAUCET
- Goals:
  - Robust software
  - Reliable hardware
  - Extensive production testing
  - Operators comfortable using it
  - Interoperate with existing tools
- 2 network engineers in 2 months produced initial version
- Deployed FAUCET on their office network
  - Office network is still FAUCET powered to this day
- FAUCET codebase has become a collaborative effort
  - REANNZ, Google, University of Waikato, Victoria University of Wellington, etc...

Features

- Lightweight Open Source SDN controller
- OpenFlow v1.3
- Production quality
- Well tested
- Multi-vendor
- Supports Layer 2 and Layer 3
- Policy driven approach to extensibility
- Installs in <30 seconds

Architecture overview



Faucet

Design philosophy - simplicity

- Easy to deploy
- Easy to operate
- Easy to modify
- Easy to upgrade

Design philosophy - SDN

- Move control plane implementation from hardware to general purpose compute
  - Implement forwarding & routing in language that's easy to read & modify
  - Use open standard to push rules to datapath
- Portable across many different vendors without drivers
  - Work with vendors to support the FAUCET packet processing pipeline
- We follow the OpenFlow 1.3 standard
  - No vendor extensions

Throw out the kitchen sink

- FAUCET is intentionally small
  - ~10,000 lines of code
  - ~5,000 lines of tests
- Implement useful primitives in FAUCET that can be built on top of
  - Forwarding, VLANs, ACLs, L3 FIB
- Implement some additional protocols for interop
  - BGP, Stacking, LACP, ARP & IPv6 ND
- Leave protocols modular so they are only turned on when configured

Controller state

- Persistent state is stored in configuration files
- Everything else is ephemeral
  - L2 MAC learning
  - Next hop resolution
- If in doubt, throw it out
- High Availability without a tightly-coupled cluster

Configuration

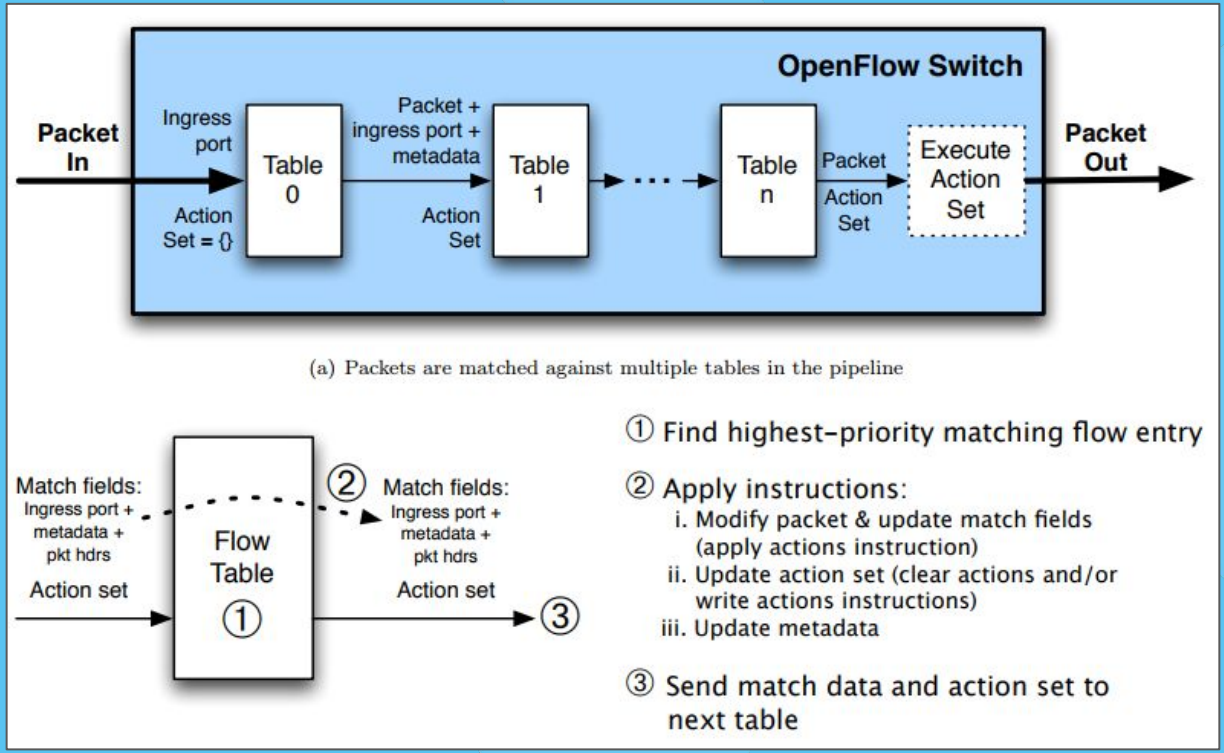
- YAML-based configuration file
- Represents topology & features of network
- FAUCET is idempotent
  - Give 2 controllers same configuration and they will configure the network the same
- Working on adding an abstraction layer on top for real time changes
- Change configuration file then signal FAUCET to reload
- FAUCET will compute diff between configuration and apply to network

Configuration Example

```
vlans:
    office:
        vid: 100
        description: "office network"
        faucet_vips: ['10.0.100.254/24']
dps:
    sw1:
        dp_id: 0x1
        hardware: "Open vSwitch"
        interfaces:
            1:
                description: "host1 container"
                native_vlan: office
            2:
                description: "host2 container"
                native_vlan: office
```

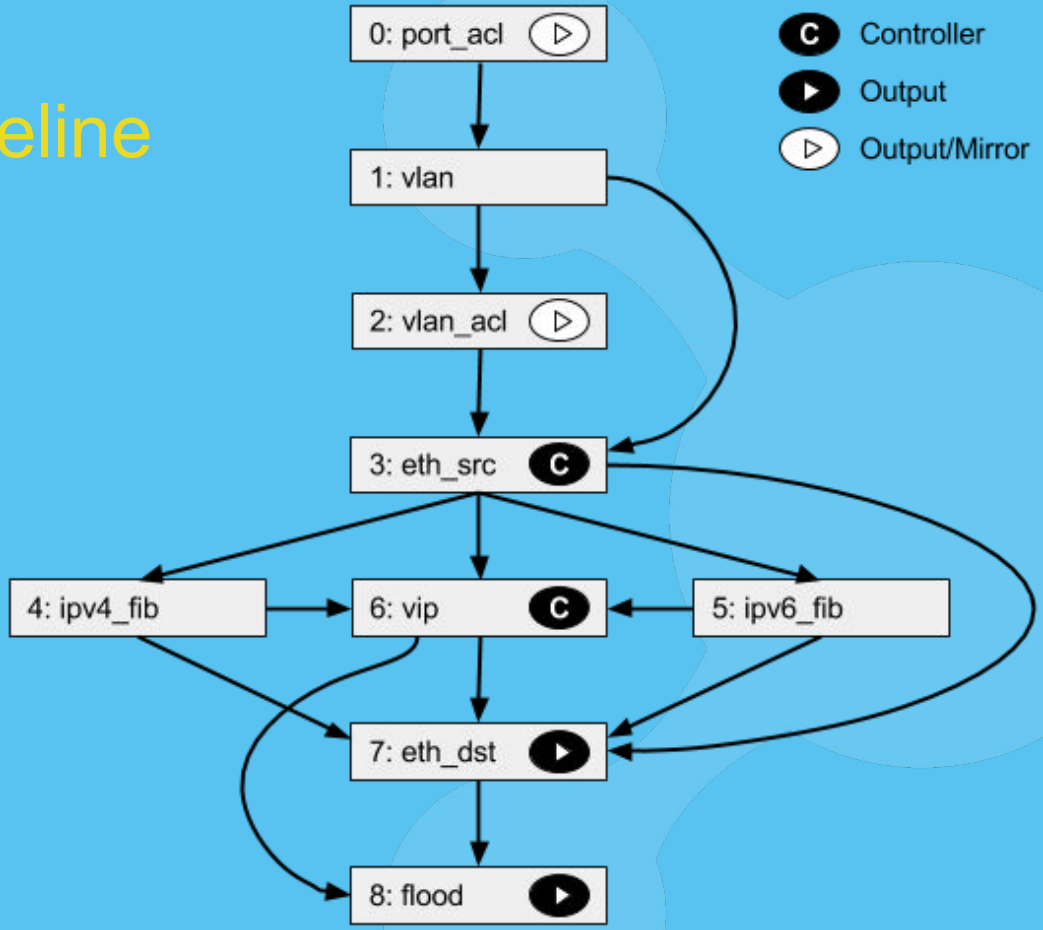
Faucet

OpenFlow processing pipeline



Faucet

Faucet pipeline



Faucet

Testing

- Testing allows us to change code without breaking features
- Test suite includes
  - Integration tests
  - Unit tests

Unit tests

- Helps developers test small pieces of code for correctness
- Easy to write
- Tests run very quickly
- >90% test coverage

Integration tests

- Test entire system
- Boot up many different network scenarios in parallel
- Ensure features work and packets correctly move through network
- Runs against Open vSwitch and real hardware
  - Lets us prequalify devices for FAUCET support
- Much slower to run
  - 30 - 60 minutes

Monitoring

- To be considered operational need to satisfy the needs of network engineers
- Visibility over network
- Monitoring and control are separate network functions
- FAUCET controls the network (read/write)
- GAUGE monitors the network (read-only)

Network visibility - GAUGE

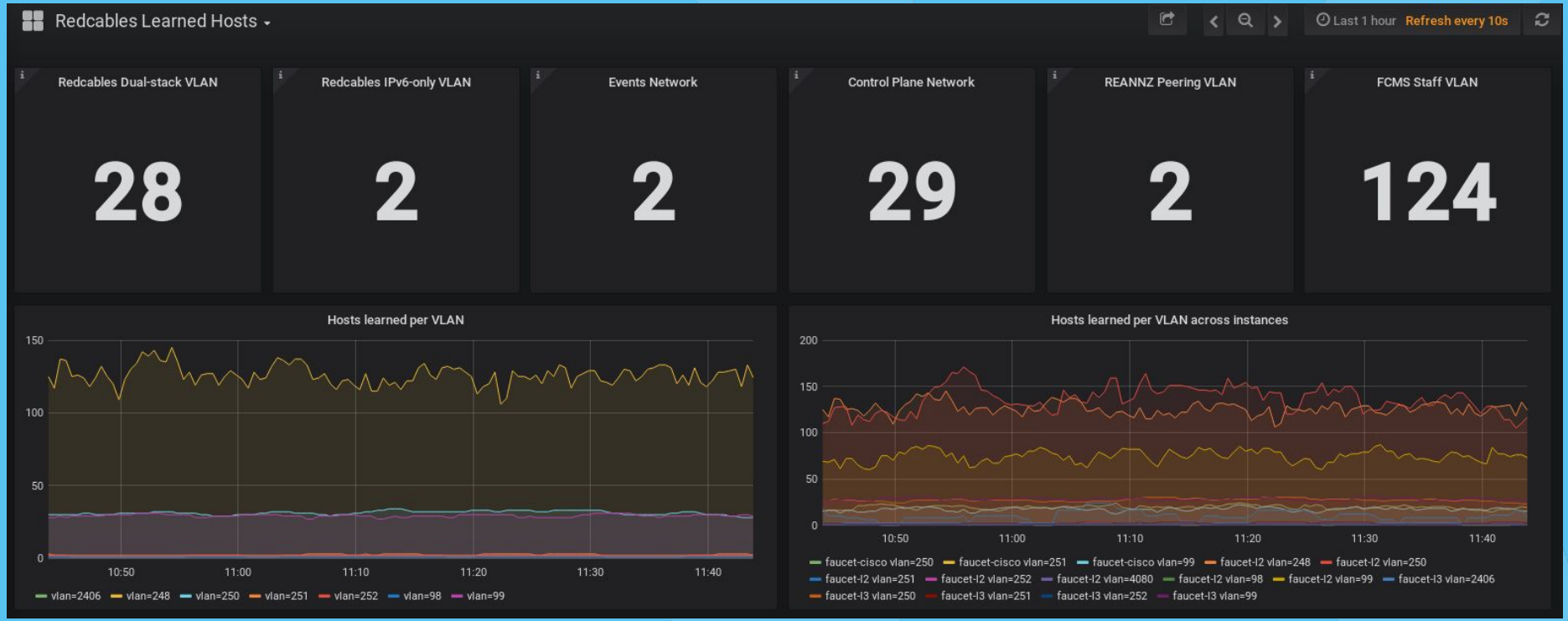
- Fetches metrics from OpenFlow v1.3 dataplanes
  - MAC learning information
  - Port state
  - Port counters (bytes in/out, packets in/out, errors)
- Pushes metrics to a database
  - InfluxDB
  - Prometheus
- Can use `fctl` tool to query database manually
- Can use grafana to make real-time dashboards

Network visibility - FAUCET

- FAUCET controller also exports metrics to prometheus
- Allows us to instrument control plane
  - Learning capacity
  - CPU/Memory usage
  - Inventory

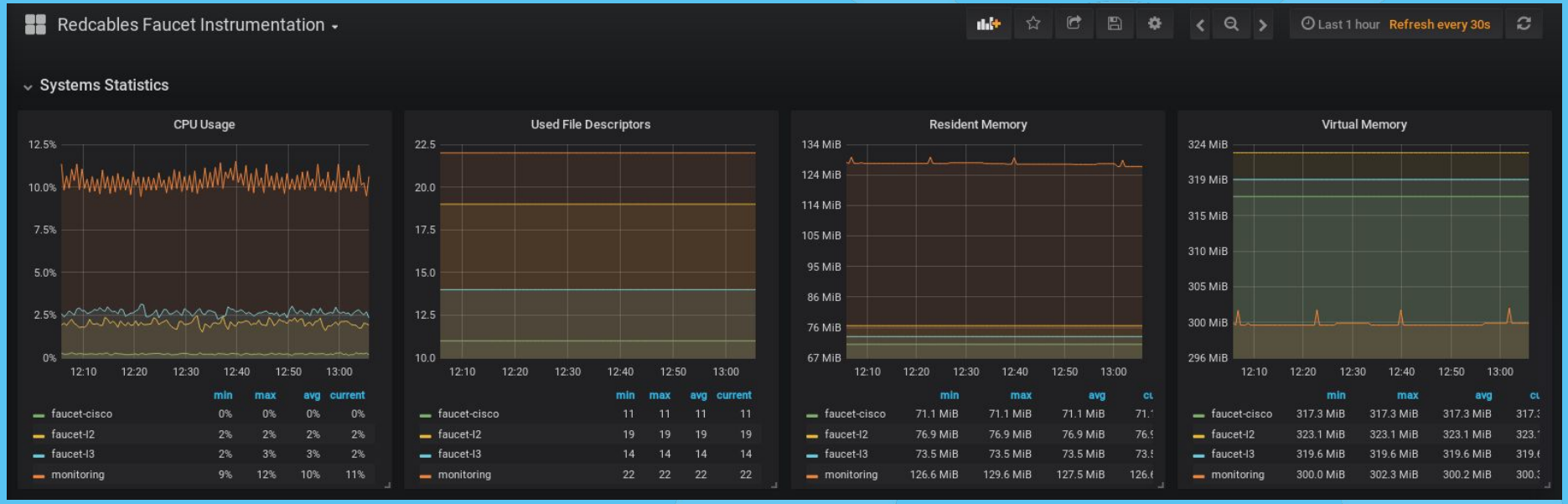
Faucet

Network visibility



Faucet

Controller visibility



Faucet

Policy

- Network policy is implemented with FAUCET ACLs
- A FAUCET ACL has a match and action
  - Matches anything OpenFlow can
  - Action can be DROP, ALLOW, OUTPUT, MODIFY

Policy use cases

- Port-based ACLs
  - DHCP and DHCPv6 spoofing protection
  - IPv6 Router Advertisement Guard
  - BCP38
  - NFV offload, output 802.1x EAPOL frames to NAC
- VLAN-based ACLs
  - Drop anything other than IPv6 ethertype on IPv6-only network
- IVR ACLs
  - Limit traffic between VLANs
- PBR ACLs
  - Assign client subnets to a specific upstream

Policy example

- IPv6 Router Advertisement Guard

```
- rule:
    dl_type: 0x86dd    # ipv6
    nw_proto: 58       # icmpv6
    icmpv6_type: 134   # router advertisement
    actions:
        allow: 0       # drop
```

Faucet

Use cases

- Enterprise
  - REANNZ
  - WAND Network Research Group
  - University of Waikato
- Security
  - CyberReboot Poseidon
- IXP
  - University of Tokyo
  - Osaka NSPIXP-3
  - TouIX
- HPC
  - SC18

What makes an enterprise network?

- Connects users to services and Internet
- Lots of copper ports and many wireless APs
- Hard to design a standard build-out
  - Too many special cases
  - Odd building layouts
- Often have no control over devices at the access layer
  - BYOD
- Network design has to scale to support all these edge-cases

Problems in enterprise networks

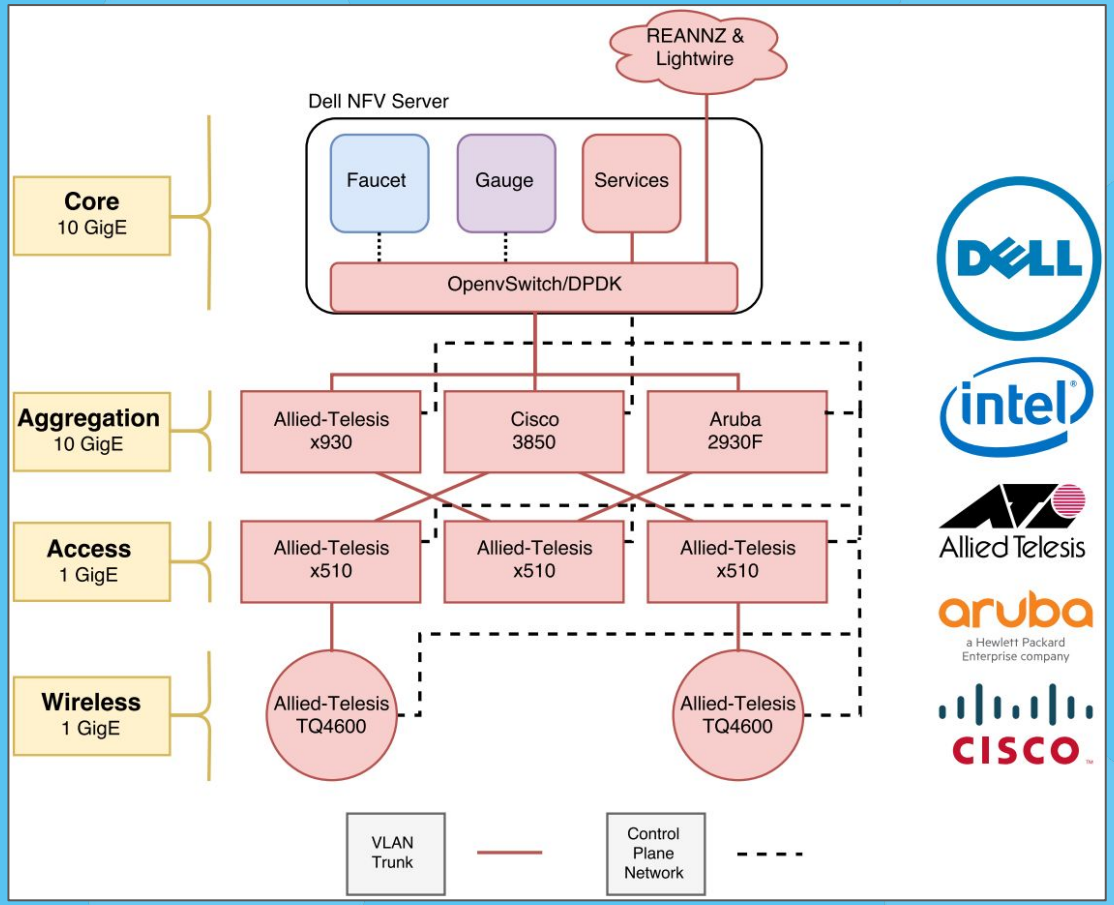
- Hand configuration….
- Configuration automation isn’t always the answer
  - Multi-vendor is difficult
  - Can’t define our own learning behaviour
- We want Devops for networks
  - Faucet: Deploying SDN in the Enterprise
    Using OpenFlow and DevOps for rapid development
    https://queue.acm.org/detail.cfm?id=3015763

Devops approach to networking

- Zero Touch Networking (ZTN)
  - gNMI
- Automate network behaviour
- Capture network definition in version control
- Run network scenarios with automated test suite
- Continuous Integration
  - Push on green
- Automate hardware purchasing

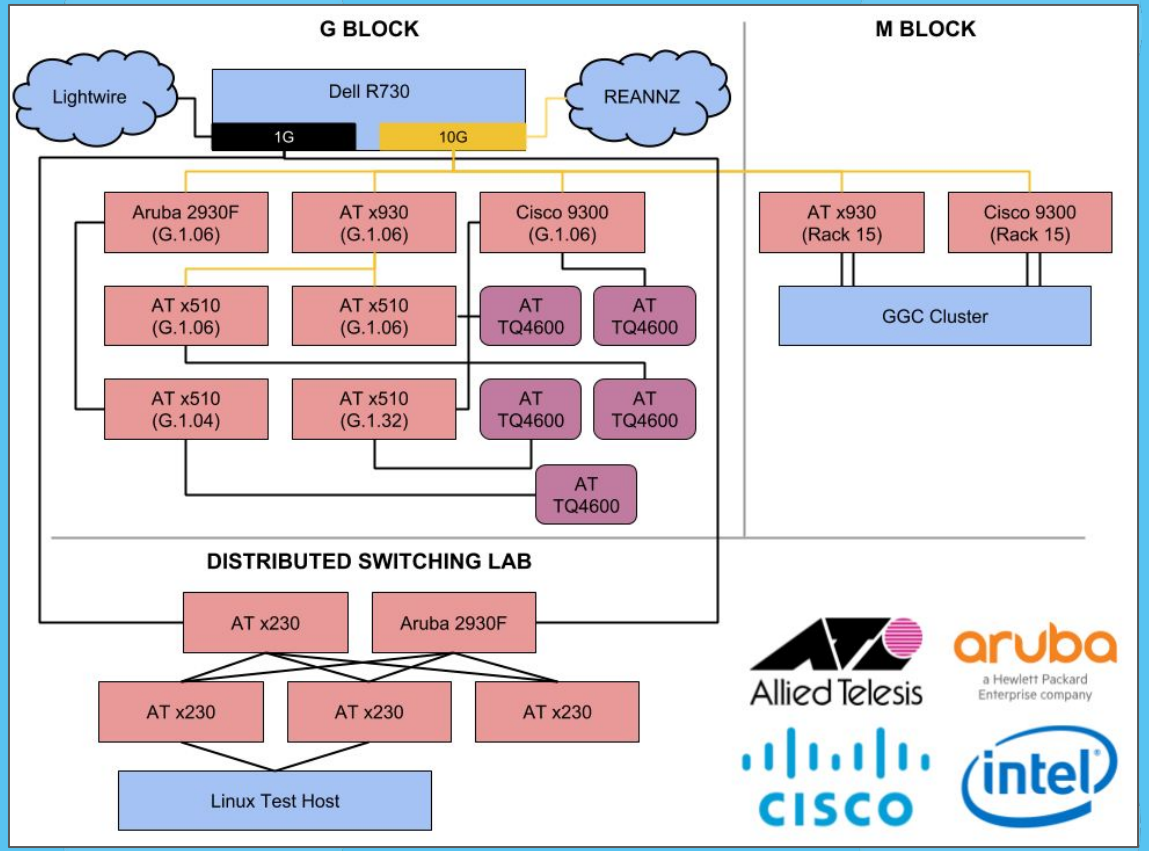
WAND Network

AS 134227
192.107.171.0/24
192.107.172.0/24
2001:df2:9d00::/45
248 OpenFlow ports



Faucet

WAND Network v2



Faucet

Trying out FAUCET

```
$ echo "deb https://packagecloud.io/faucetsdn/faucet/ubuntu/bionic main" \
  | sudo tee /etc/apt/sources.list.d/faucet.list
$ curl -L https://packagecloud.io/faucetsdn/faucet/gpgkey \
  | sudo apt-key add -
$ sudo apt-get update
$ sudo apt-get install faucet gauge
```

or

```
$ docker run -d --name faucet \
     -v /etc/ryu/faucet/:/etc/ryu/faucet/ \
     -v /var/log/ryu/faucet/:/var/log/ryu/faucet/ \
     -p 6653:6653 -p 9244:9244 \
     faucet/faucet
```

Learn more **https://docs.faucet.nz**

Trying out FAUCET

- Watch me configure a network after coffee
- Try for yourself in Hands on Hacking session this afternoon

Questions?

Learn more **https://faucet.nz**

Follow us on Twitter **@faucetsdn**