Using an Open Source SDN Controller to Deploy a Multi-Terabit/s Production Network

Brad Cowie

Faucet

Introduction to faucet

- Lightweight Open Source SDN controller
- OpenFlow v1.3
- Supports Layer 2 and Layer 3
- Policy driven approach to extensibility

A little bit of history

- Started with a set of goals:
    - Robust software
    - Reliable hardware
    - Extensive production testing
    - Operators comfortable using it
    - Interoperate with existing tools
- Born out of collaborative effort of SDN community in New Zealand
    - REANNZ, Google, University of Waikato, Victoria University of Wellington, etc…
- Now up to 47 contributors from around the world
- Our code base just turned 5 years old

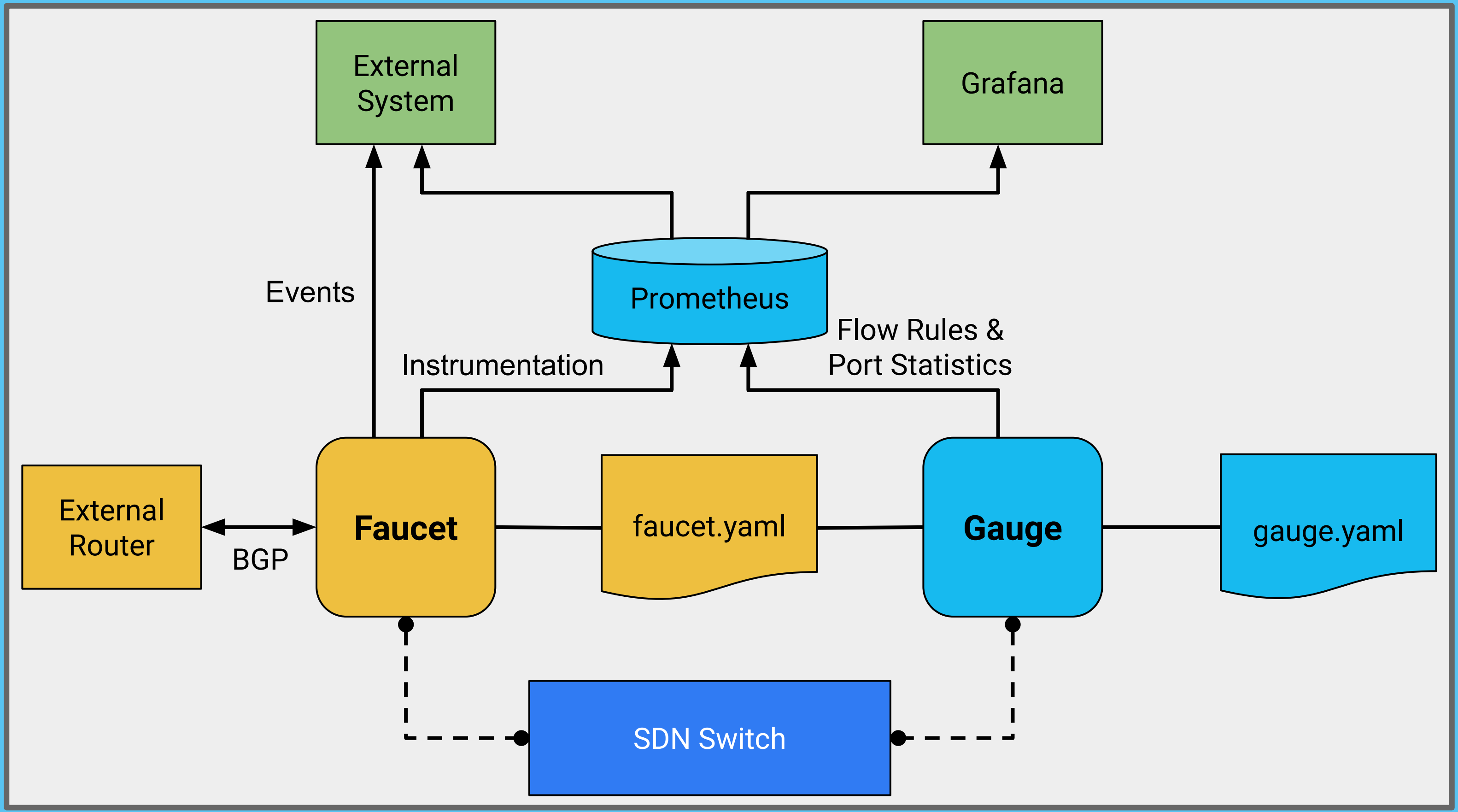
Why use faucet?

- Low system resource requirements
- Uses multi-table OpenFlow pipeline
- Multi-vendor without drivers
- Well tested, production quality
- High Availability through idempotency
- Controller is not in the forwarding path

Faucet

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


External System
Grafana
Events
Prometheus
Flow Rules &
Port Statistics
Instrumentation
External Router
BGP
Faucet
faucet.yaml
Gauge
gauge.yaml
SDN Switch

Faucet

Faucet configuration

- YAML configuration file
- Represents topology & features of network
- Faucet is idempotent
  - Give 2 controllers same configuration and they will configure the network the same

```yaml
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"
                native_vlan: office
            2:
                description: "host2"
                native_vlan: office
```

Faucet

Faucet pipeline

Apply custom policy to port

Ensure packet has correct VLAN tag applied

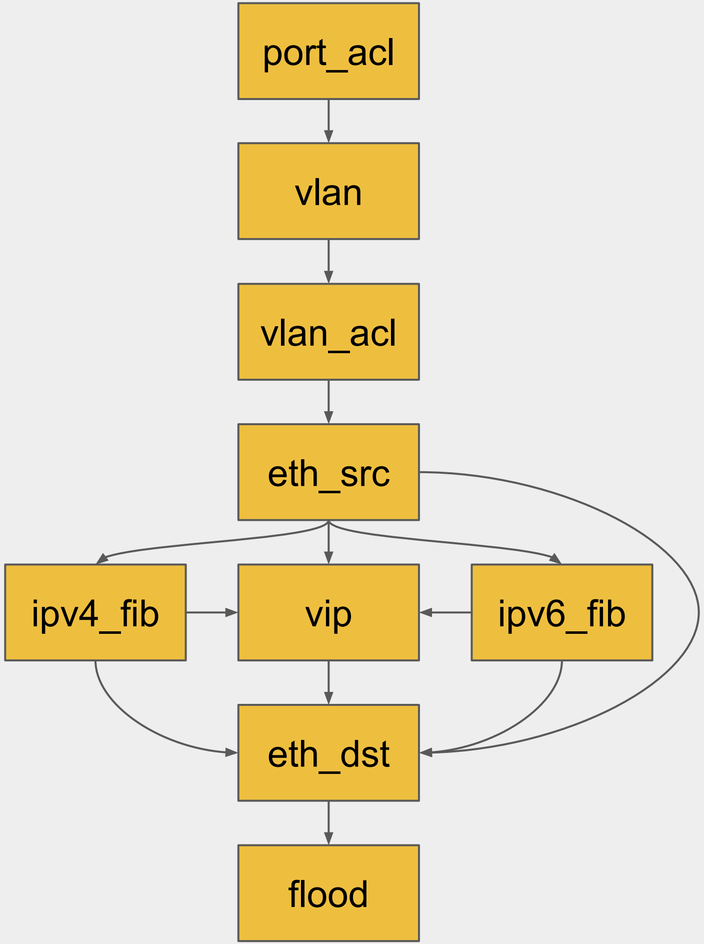
Apply custom policy to VLAN

Determine if packet is forwarded or routed. Learn from unknown MAC

Perform routing & neighborhood discovery

For known MAC, output packet on correct port

Flood packet for unknown MAC inside VLAN



Faucet policy

- Can modify behaviour without changing code by adding ACLs
- ACL has a match an action
    - Matches on 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
  - NFV offload, output 802.1x EAPOL frames to NAC
- VLAN-based ACLs
  - Drop anything other than IPv6 ethertype on IPv6-only network
- Inter-VLAN Routing ACLs
  - Limit traffic between VLANs
  - Implement Policy Based Routing

Example faucet policy

- IPv6 Router Advertisement Guard

```
acls:
  ipv6_ra_guard:
    - rule:
        dl_type: 0x86dd   # ipv6
        nw_proto: 58      # icmpv6
        icmpv6_type: 134  # router advertisement
        actions:
          allow: 0        # drop
```

Faucet

Faucet test suite

- Unit test coverage (>90%)
- Integration test suite (>180 scenarios)
- Run both against every PR
  - Avoid regressions
  - Allows us to refactor frequently
- Integration tests can be run against hardware too
  - Easy to qualify devices for faucet support
  - Vendors can use as part of QA for new firmwares

Faucet

Push on green

- Leverage integration tests as part of deploy process
  - Implement test cases that cover deployment
- Only deploy new faucet code/configuration when tests pass
- This allows for frequent (automated) deploys
  - I deploy on my faucet network at work once a week

SC18

- Annual SuperCompute Conference in the US
- 30th anniversary was November 2018 in Dallas, Texas
- “SCinet” network built each year to power conference
  - One of the world’s fastest temporary networks
- Faucet was deployed on SCinet for SC18

SCinet 2018

- 4.02 terabit/s connectivity
- $52 million in hardware
- 4.25 tons of equipment
- 225 volunteers

Photo credit: SC Conference Series

Faucet

Let’s take a step back

- How did we get involved?
- Approached conference committee April 2018
- Proposed a mutually beneficial faucet deployment
- Our proposal was accepted and work begun...

Phase 1: Planning

- Months of video conferences from NZ to USA/Europe
- Faucet SCinet team distributed between
  - Hamilton, New Zealand
  - Wellington, New Zealand
  - Berkeley, California, USA
  - Champaign, Illinois, USA
  - Dallas, Texas, USA

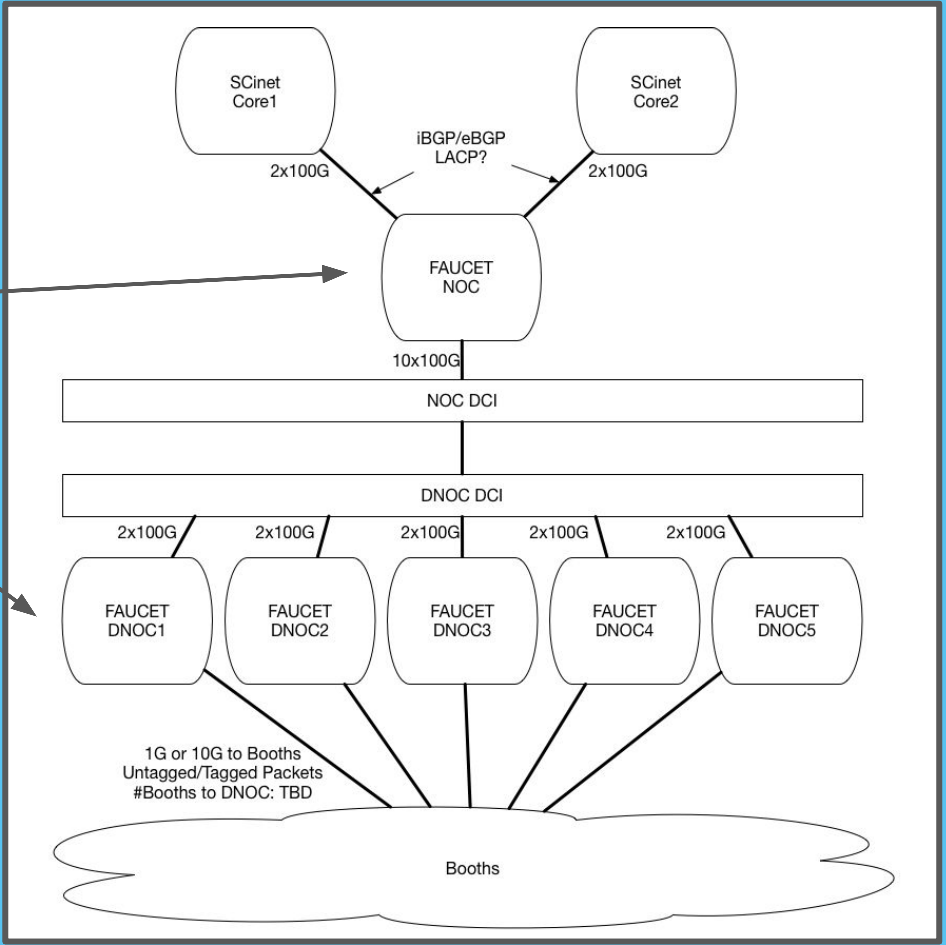
Faucet

Scope for faucet deployment

- Faucet will run in parallel with regular network
- Peer with regular SCinet core and advertise/receive routes
- Will provide percentage of booths on show floor Internet access
- Each booth has separate VLAN and subnet
- Booth can be connected at 1GbE, 10GbE or 100GbE
- Customer information comes from Django web app

Initial design

- Faucet needs physical presence in
  - Core layer (NOC)
  - Access layer (DNOC)

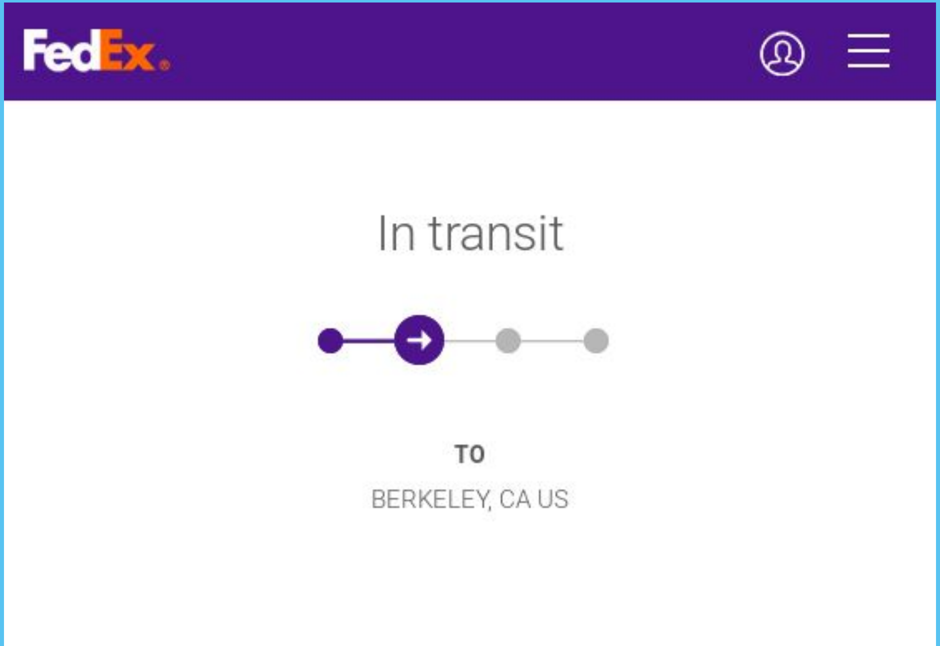


Faucet

Phase 2: Source hardware

- Approach hardware vendors and ask nicely for hardware
- Thanks Allied-Telesis, Cisco and NoviFlow!
    - 2 devices each
- Coordinate shipping to test lab in Berkeley, California

Wait

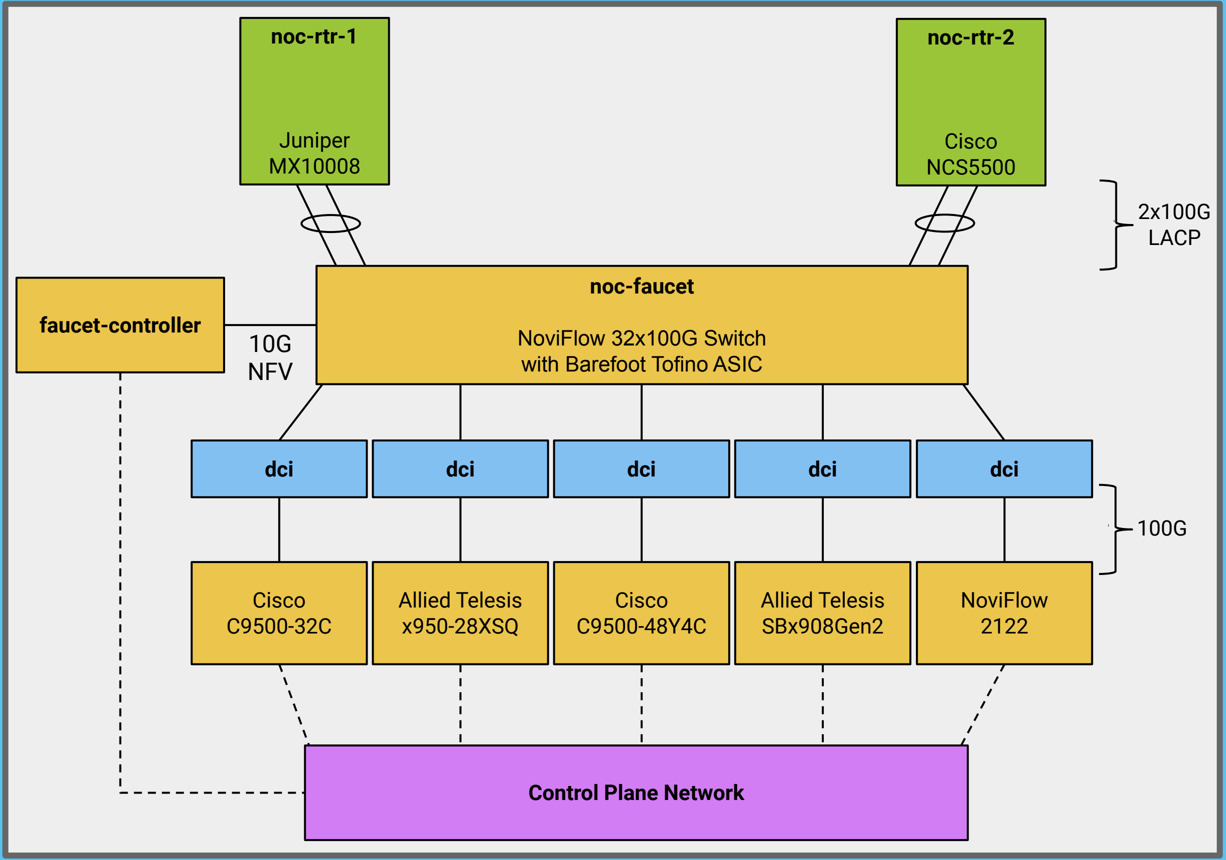


Iterate on network design

Port counts:

- 75x 100GbE
- 146x 10GbE
- 48x 1GbE

9.008 Terabit per second



Faucet

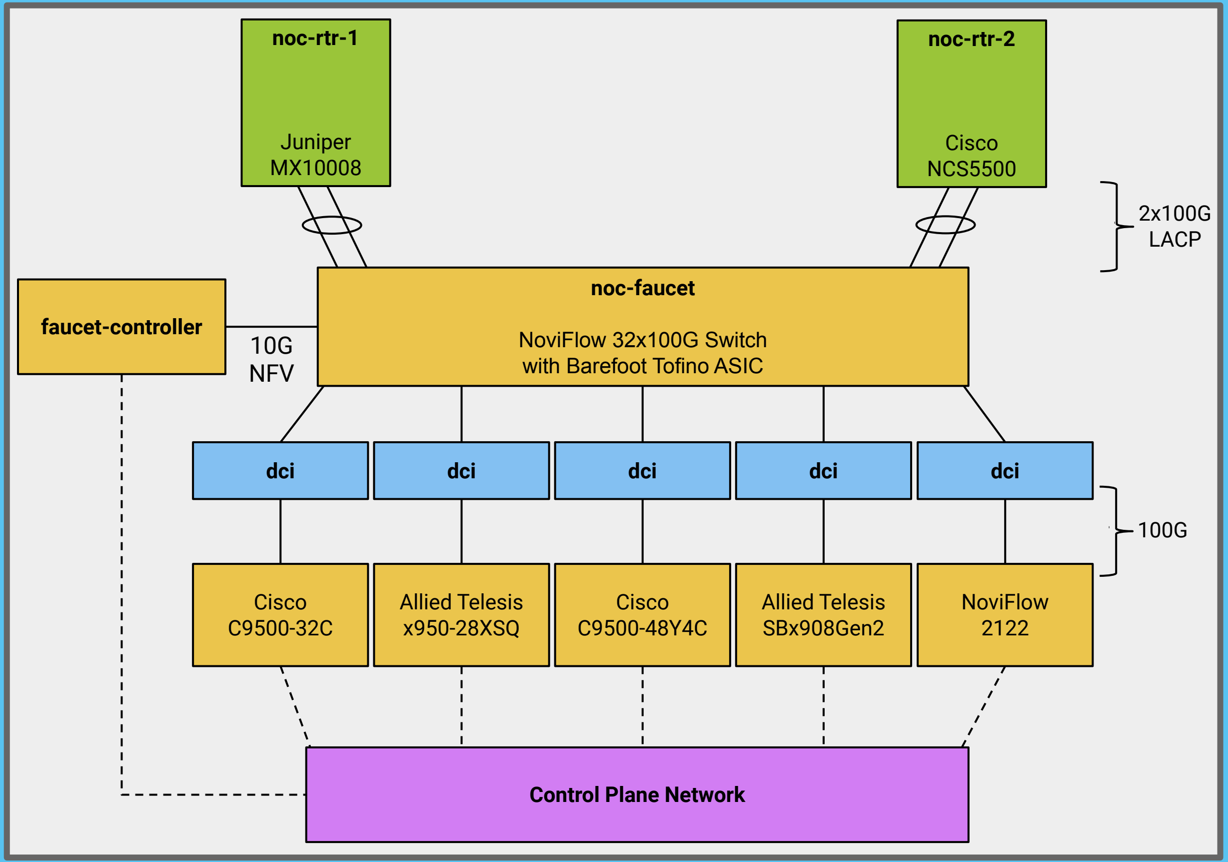
Network functions

Faucet will provide:

- VLANs
- IPv6 Router Advertisements
- Inter-VLAN Routing
- Network Security Policy

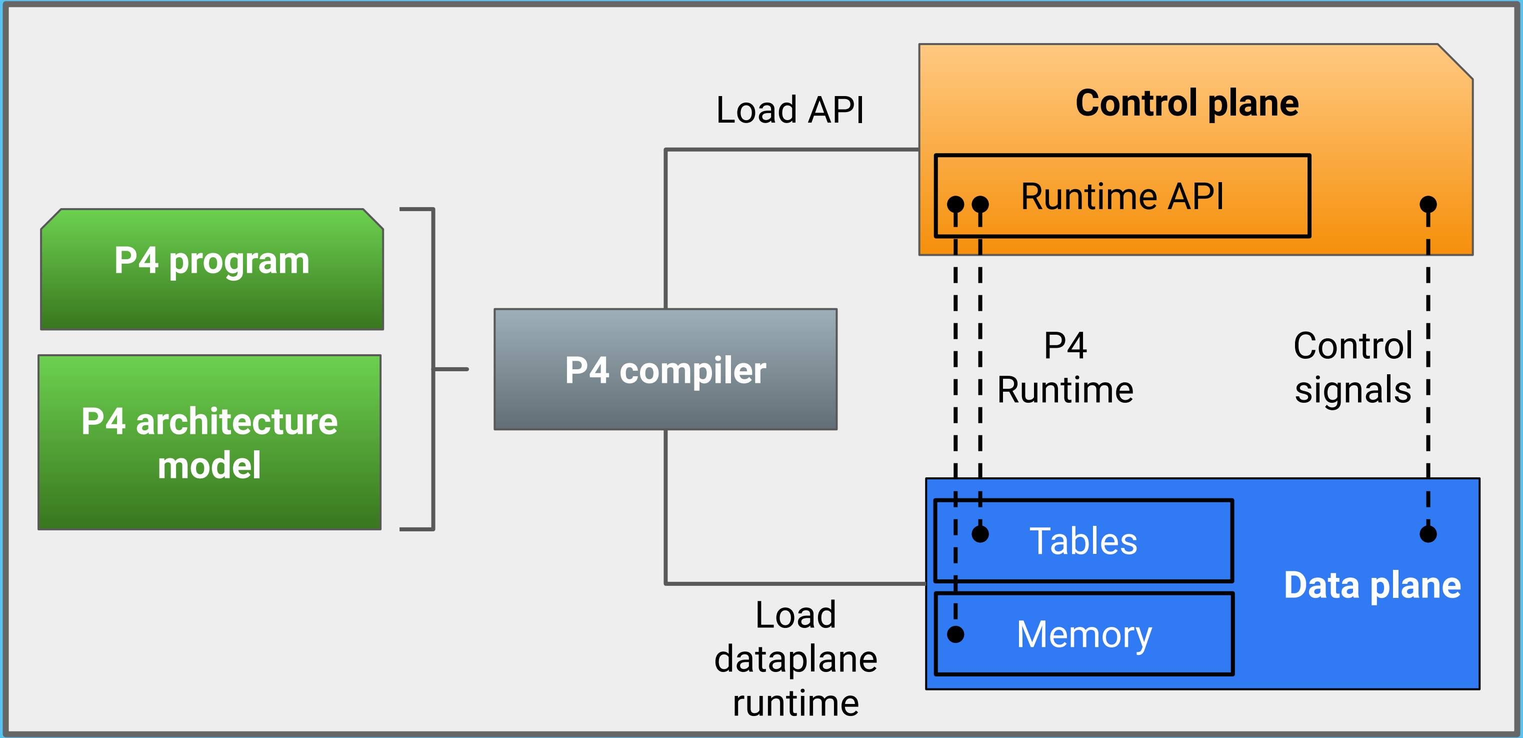
NFV Services:

- DHCP
- BGP



Faucet

Brief introduction to P4



Faucet

Combining P4 and OpenFlow

1. Write OpenFlow as P4 application
2. Compile application with Barefoot's P4 compiler to run on Tofino ASIC
3. Faucet uses OpenFlow as runtime API (instead of P4 Runtime) to push rules to P4 ASIC

Feature written by NoviFlow and available in their NoviWare Network OS

Development

- Make sure required features work by adding integration tests
- Built a scale model back in New Zealand
- Started work on automation

Automation

- Define a set of tasks once, can then repeat tasks at no cost
- Applies perfectly for network service definitions
- Faucet SCinet deployment heavily used ansible

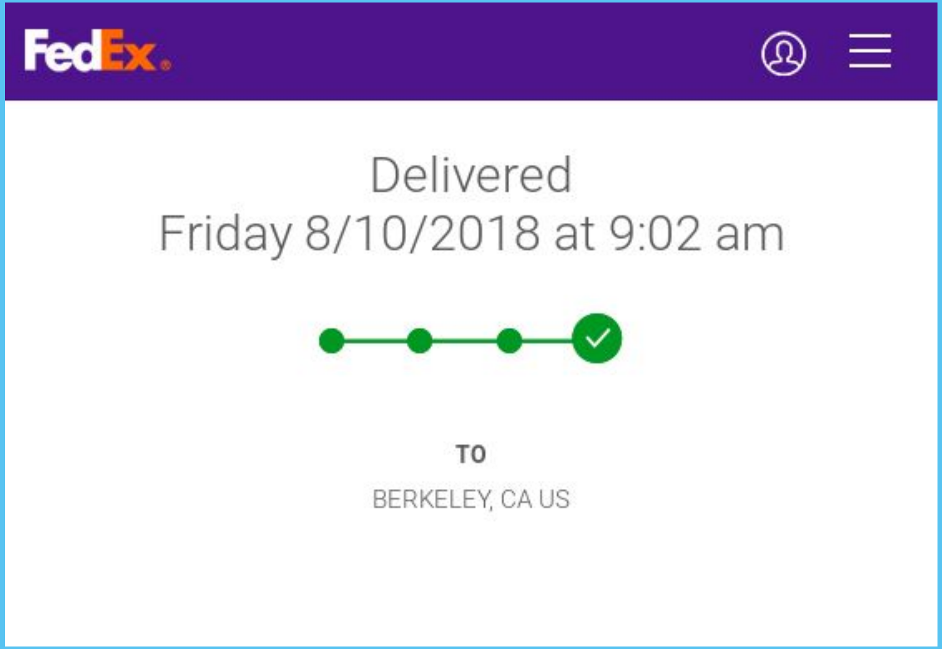
Automation

SCinet Customer DB	Python	Ansible	Configuration
Source of truth for network	Use django models to dump all customer connections into YAML	Take YAML and use jinja2 templates to output configuration	• Faucet • DHCP • BGP • Linux Interfaces
	290 lines of YAML	180 lines of ansible 260 lines of jinja2 templates	1200 lines of config

Faucet

Available on GitHub

- Ansible code
  - https://github.com/wandsdn/sc18-ansible
- Generated faucet configuration
  - https://github.com/wandsdn/sc18-faucet-configs

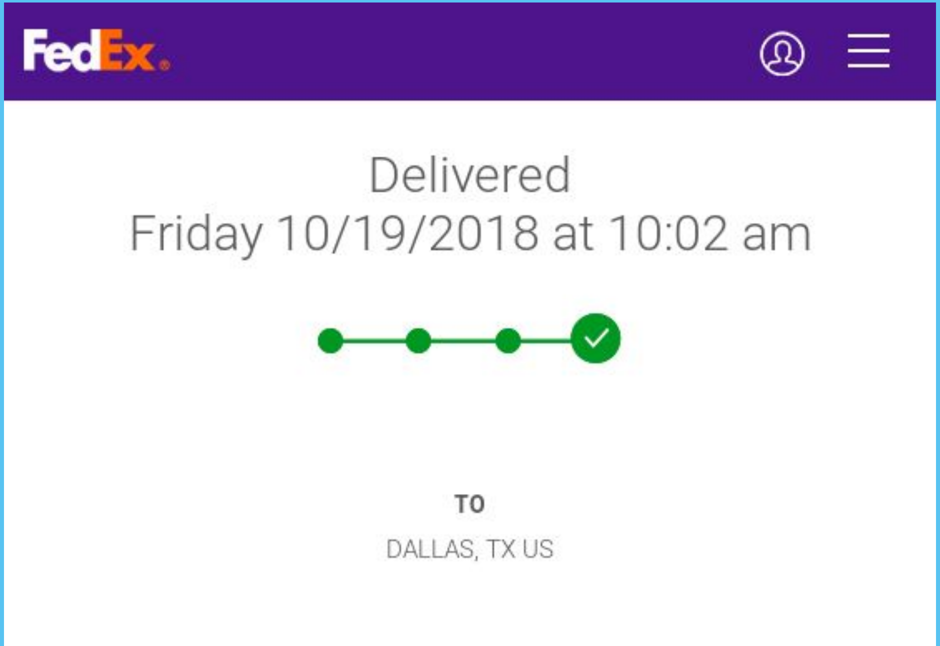

FedEx
Delivered
Friday 8/10/2018 at 9:02 am
TO
BERKELEY, CA US

Phase 3: Testing

- Use ansible to deploy configurations
- Throw a lot of traffic at it
- Network namespaces are great!
  - Can easily spin up thousands of network namespaces on a single machine
  - Use a network namespace to represent fake booth
- Write some bash scripts to manage
  - Test reachability (*onall.sh pingall.sh*)
  - Test throughput (*randomiperf.sh*)

Faucet

Ship network off to conference venue

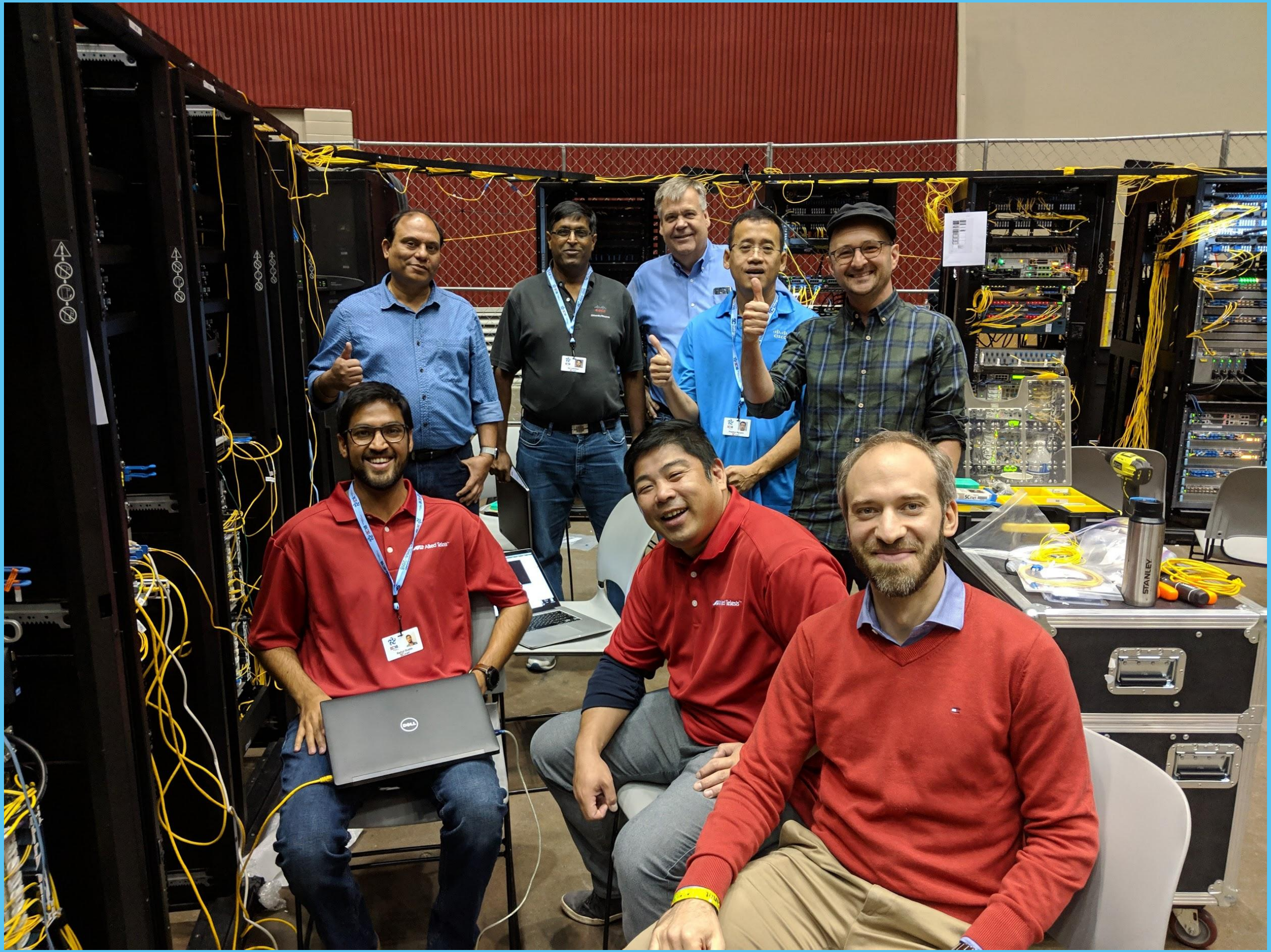


Faucet

Phase 4: Staging

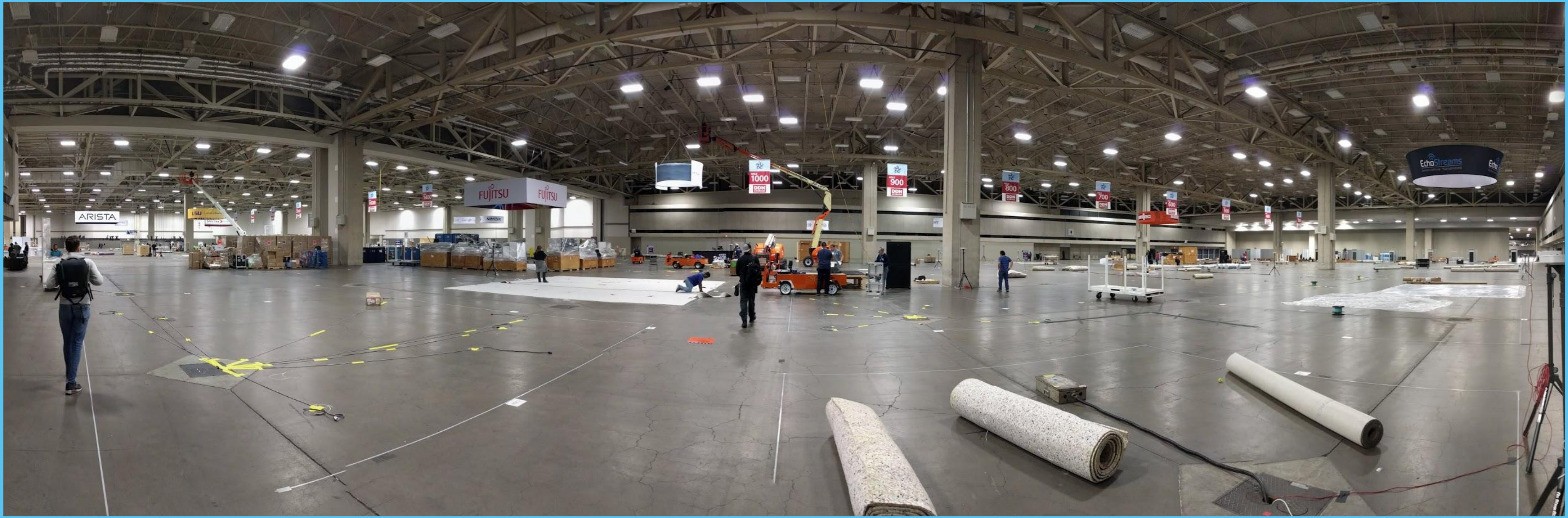
- SCinet equipment is staged away from show floor
- Network is turned on for the first time
- Interop testing and diagnosing problems

Faucet staging team photo after network booted up without issue



Faucet

Phase 5: Show Setup



Faucet

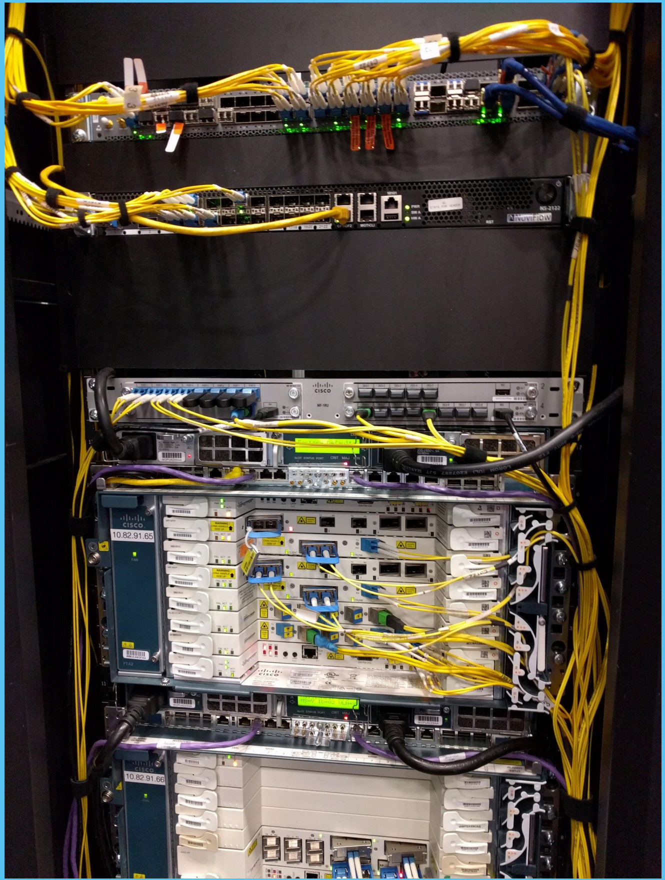
Faucet

Phase 5: Show Setup

- Customer database is finalised
- Trigger automation to deploy configuration
- Work with booths as they are built to ensure access works

Faucet

DNOC



Regular switch

Faucet switch

Optical transport

Faucet

NOC

Photo credit: SC Conference Series

Faucet

Load test

- Borrow some linux boxes full of 100G interfaces to load test network

Faucet

Phase 6: Show Time

- Look for issues
  - Watch tickets queue
  - Watch monitoring
- Fix issues
  - Deploy light meters
  - Swap optics
  - Rolling fibres

Phase 6: Cleanup

- 7 months of planning
- 3 weeks of testing
- 3 weeks setup
- 1 day of tear down

Photo credit: SC Conference Series

Faucet

What did we learn?

- Automating the network services that 90% of customers use helped operators have more time to spend working on the interesting 10% of customers
- Faucet allowed us to effectively ignore the underlying network equipment
- Proved 10,000 lines of python can easily handle many terabit/s

Want to try faucet out yourself?

- Complete the tutorial series on our website
- In an hour you will be able to:
    - Install faucet in virtual environment
    - Configure policy/ACLs
    - Configure VLANs
    - Configure Routing
    - Configure NFV

https://docs.faucet.nz/en/latest/tutorials/

Questions?

https://faucet.nz

@faucetsdn

brad@waikato.ac.nz

@nzgizmoguy

Photo credit: SC Conference Series

Faucet