FAUCET AT SANDIA NATIONAL LABORATORIES

Timothy Toole

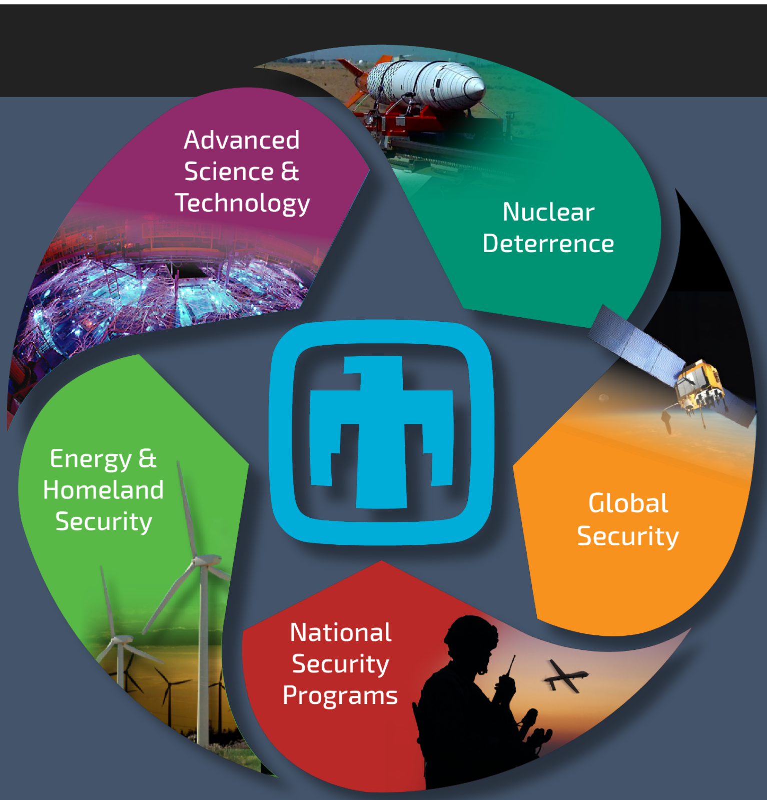
U.S. DEPARTMENT OF ENERGY

NNSA
National Nuclear Security Administration

Sandia National Laboratories

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.
SAND No. SAND2019-12281 C.

ABOUT SANDIA NATIONAL LABORATORIES



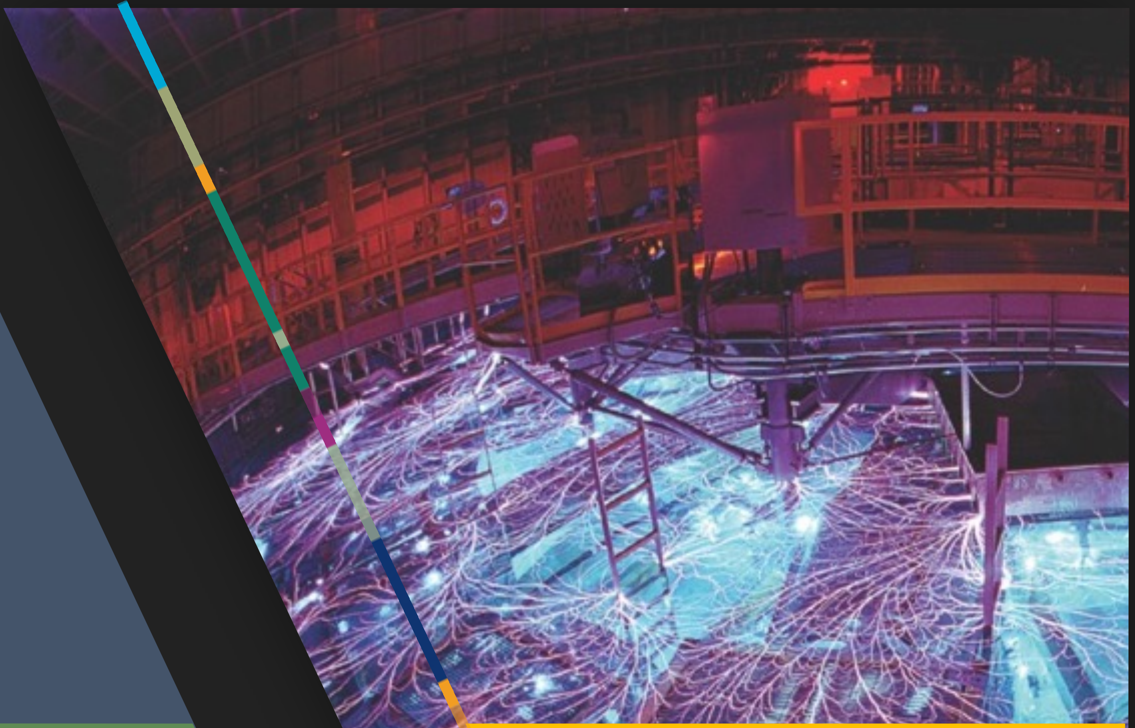
Advanced Science & Technology

Nuclear Deterrence

Global Security

National Security Programs

Energy & Homeland Security



SANDIA IS A FEDERALLY FUNDED RESEARCH AND DEVELOPMENT CENTER

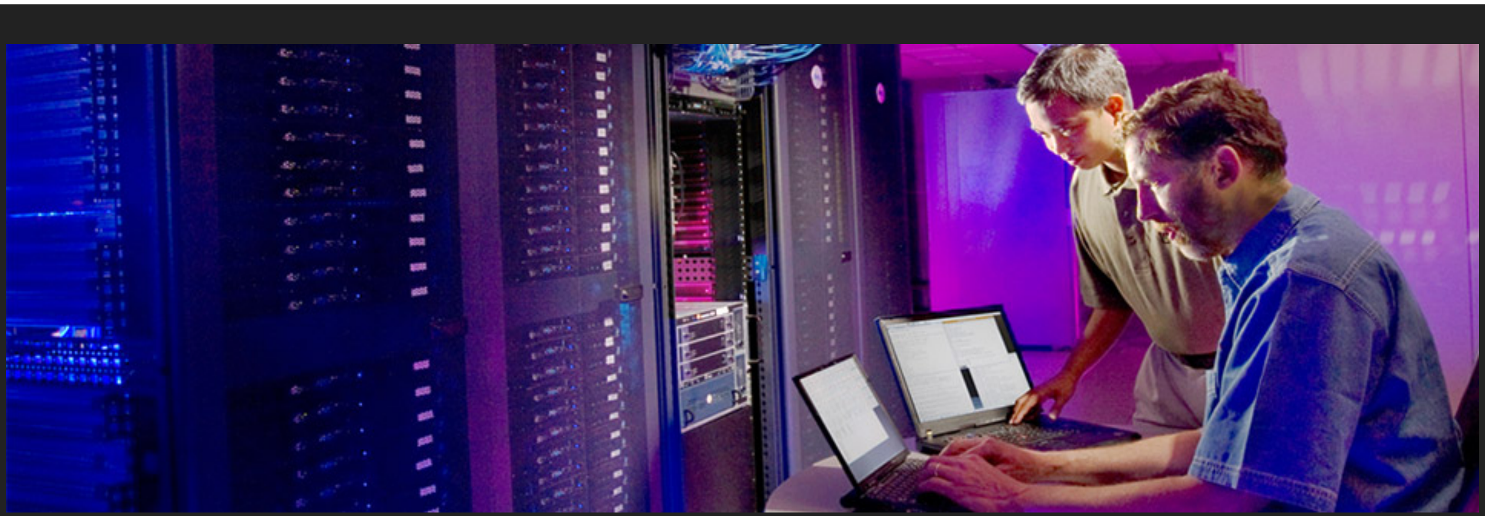
Main sites

- Albuquerque, New Mexico
- Livermore, California

U.S. DEPARTMENT OF ENERGY   NNSA National Nuclear Security Administration

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.
SAND 2018-4900 PE

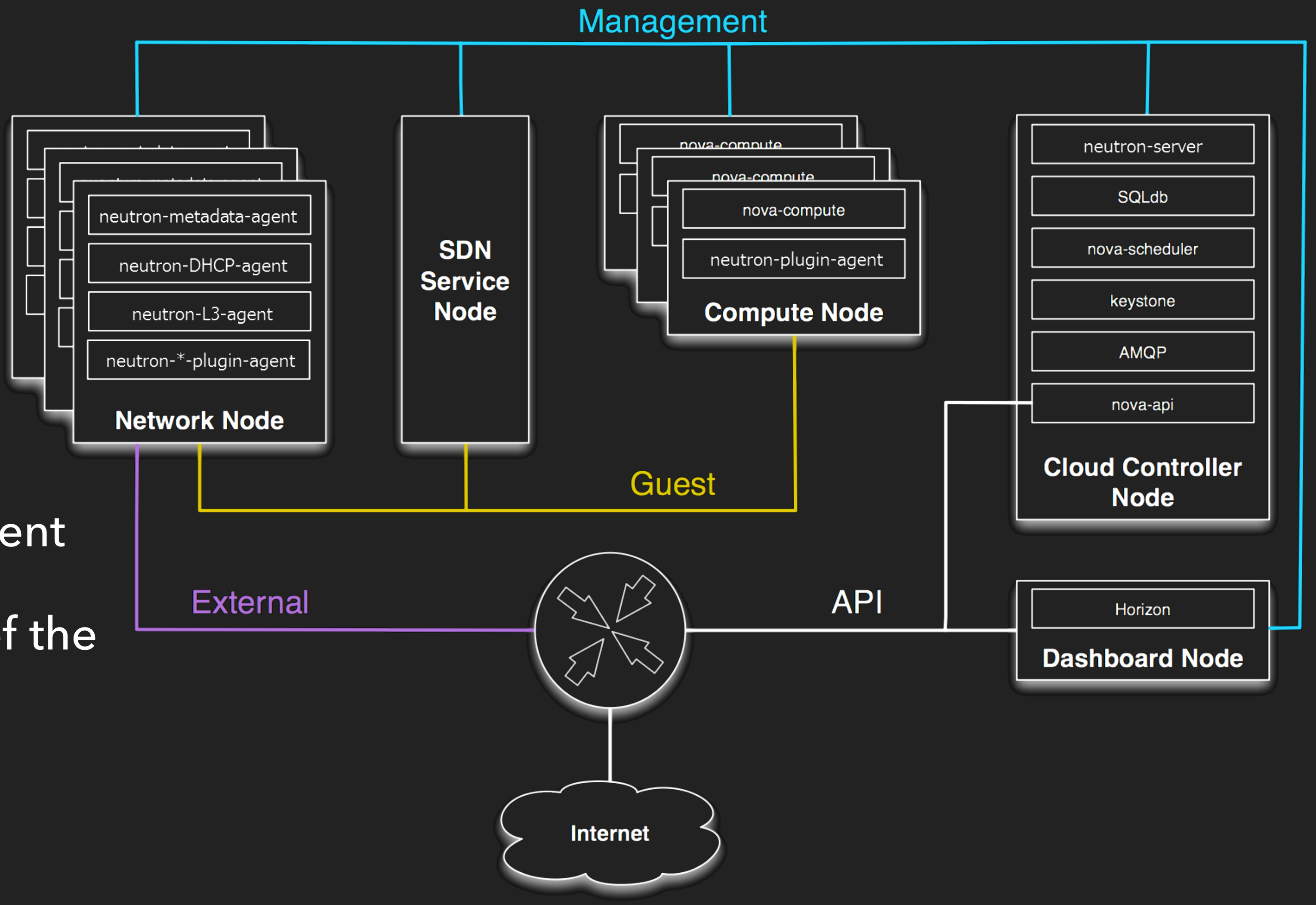
WHO WE REPRESENT AT SANDIA



- Multiple groups at Sandia conducting research & development in computing, information science and cybersecurity
    - Ensures security of critical military, government, and commercial networks using trusted systems to detect anomalies and intrusions
- Representing a team that focuses on system architectures, computer networks and analysis
    - Not necessarily working on fundamental science, but involved in developing and integrating new or novel techniques
    - Share and advise other groups and teams that benefit from the techniques
- Experience with networking, virtualization, computer systems & science

BUILDING AND UNDERSTANDING CLOUD ENVIRONMENTS

- Cloud (orchestration) environments are complex
  - Require multiple isolated physical & logical networks
  - Applies to Openstack, VMware, Kubernetes, etc.
- Desire to build and deploy multiple, isolated cloud environments
  - Full cloud environment per tenant
  - Perform application testing
- Gain real-world experience with SDN
  - Prior controller experience was in a limited scope
  - Extend beyond academic environment
  - Become an operator & maintainer of the network



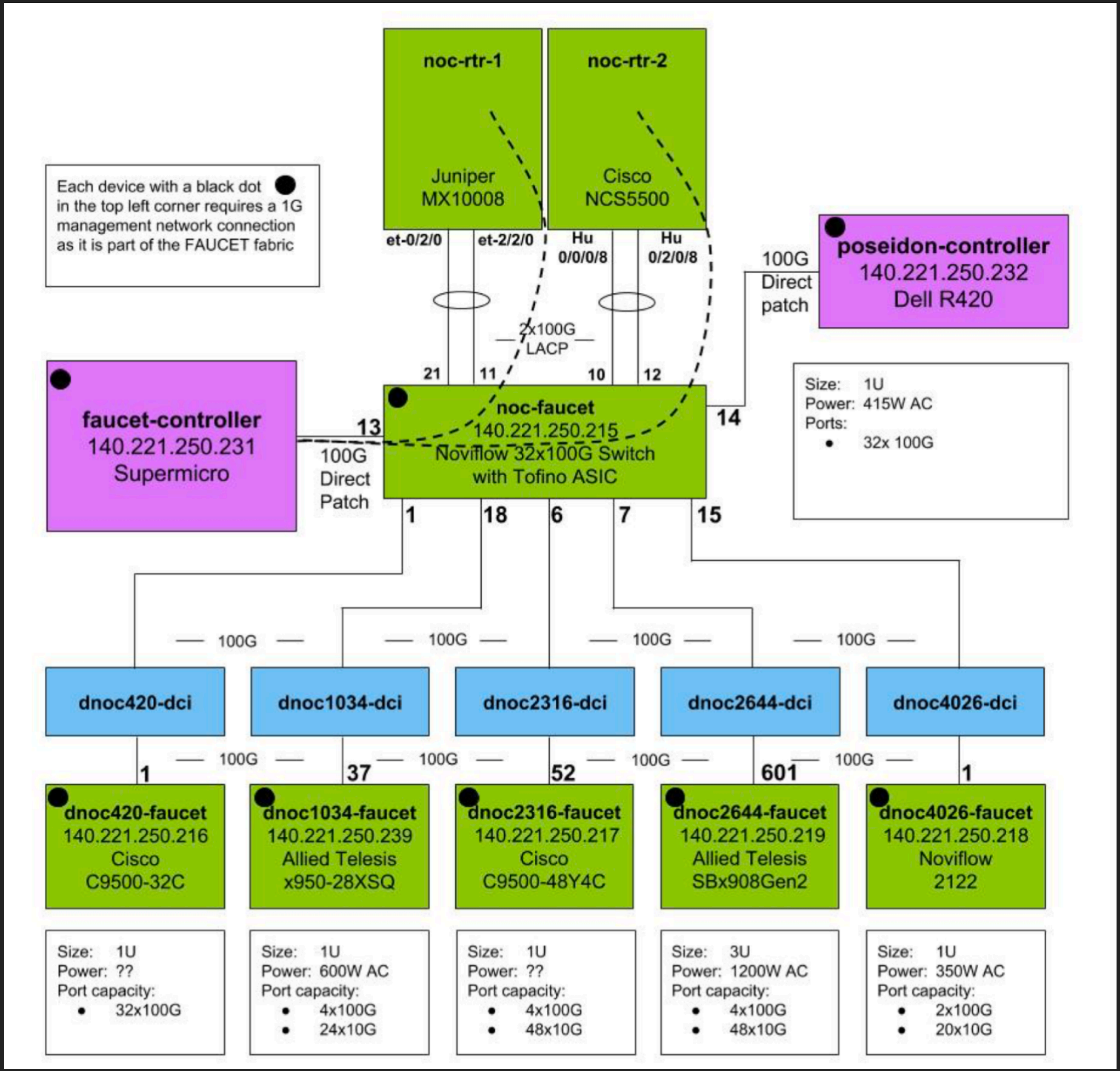
INSPIRATION: WHY THE SUPERCOMPUTING NETWORK WAS IMPORTANT

- **PRODUCTION** - Stability!
- Multi-vendor
- Statistics (Gauge)
- Analytics (Poseidon)
- https://github.com/wandsdn/sc18-faucet-configs

MAJOR Kudos to

the SC18 team!

- Use this design as a foundation for our use



Each device with a black dot ● in the top left corner requires a 1G management network connection as it is part of the FAUCET fabric

noc-rtr-1
Juniper
MX10008
et-0/2/0
et-2/2/0

noc-rtr-2
Cisco
NCS5500
Hu 0/0/0/8
Hu 0/2/0/8

2x100G LACP

21 11 10 12

● **poseidon-controller**
140.221.250.232
Dell R420

100G Direct patch

Size: 1U
Power: 415W AC
Ports:
- 32x 100G

● **faucet-controller**
140.221.250.231
Supermicro

13
100G Direct Patch

● **noc-faucet**
140.221.250.215
Noviflow 32x100G Switch
with Tofino ASIC

14

1 18 6 7 15

100G 100G 100G 100G

dnoc420-dci
dnoc1034-dci
dnoc2316-dci
dnoc2644-dci
dnoc4026-dci

100G 100G 100G 100G

1 37 52 601 1

● **dnoc420-faucet**
140.221.250.216
Cisco
C9500-32C

● **dnoc1034-faucet**
140.221.250.239
Allied Telesis
x950-28XSQ

● **dnoc2316-faucet**
140.221.250.217
Cisco
C9500-48Y4C

● **dnoc2644-faucet**
140.221.250.219
Allied Telesis
SBx908Gen2

● **dnoc4026-faucet**
140.221.250.218
Noviflow
2122

Size: 1U
Power: ??
Port capacity:
- 32x100G

Size: 1U
Power: 600W AC
Port capacity:
- 4x100G
- 24x10G

Size: 1U
Power: ??
Port capacity:
- 4x100G
- 48x10G

Size: 3U
Power: 1200W AC
Port capacity:
- 4x100G
- 48x10G

Size: 1U
Power: 350W AC
Port capacity:
- 2x100G
- 20x10G

STARTING OUT

- Solicited price quotes from Allied Telesis, Cisco, NoviFlow, HPE-Aruba
    - Used model numbers from SC18 diagram and Faucet documentation
    - Several phone conversations with vendor reps
- Cleaned out all of our [legacy] Arista switches (sad day)
- **Switching hardware list:**
    - EdgeCore Wedge 100BF-32X (Tofino) with NoviWare NOS

NoviFlow

    - Allied Telesis x950

    - Many HPE-Aruba 2930F switches for 1G Copper connections

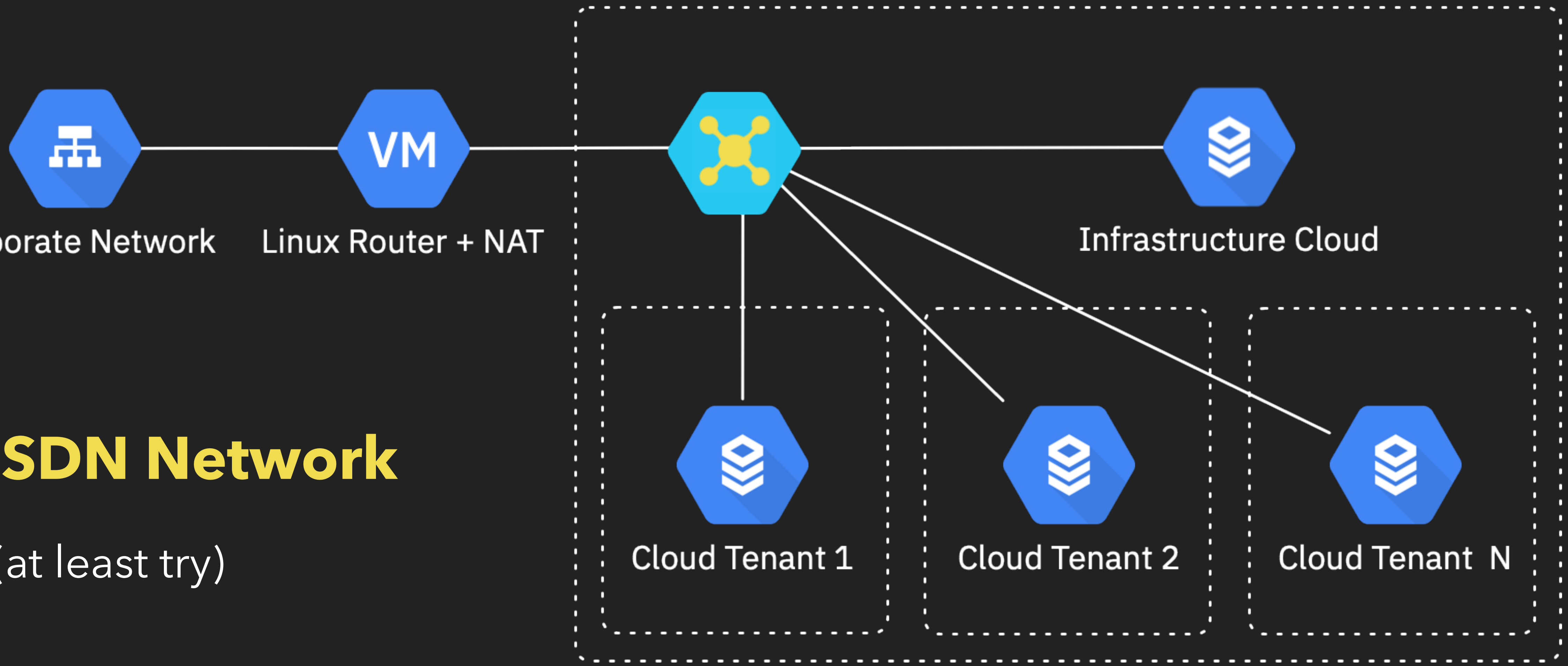
High bandwidth

↕

High bandwidth

GOALS AND APPROACH

- Desire to build out multiple, isolated cloud environments without physically re-wiring or managing a switch CLI
  - Automate testing & deployment process as much as possible - Python + bash
  - Full cloud environment per tenant

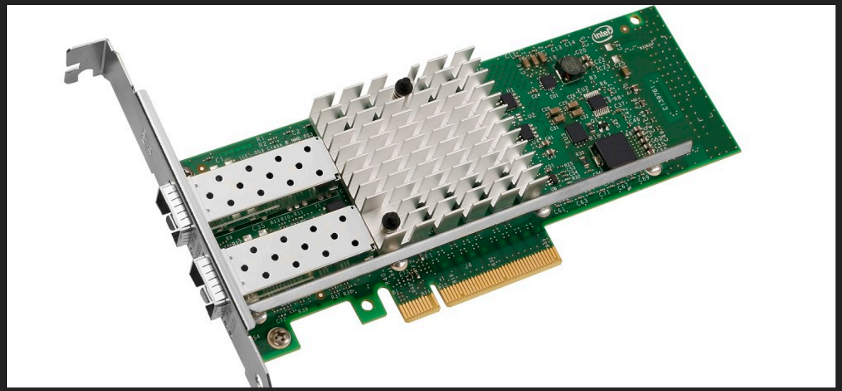


100% SDN Network

(at least try)

LAYER 1 - CONNECT ALL THE THINGS

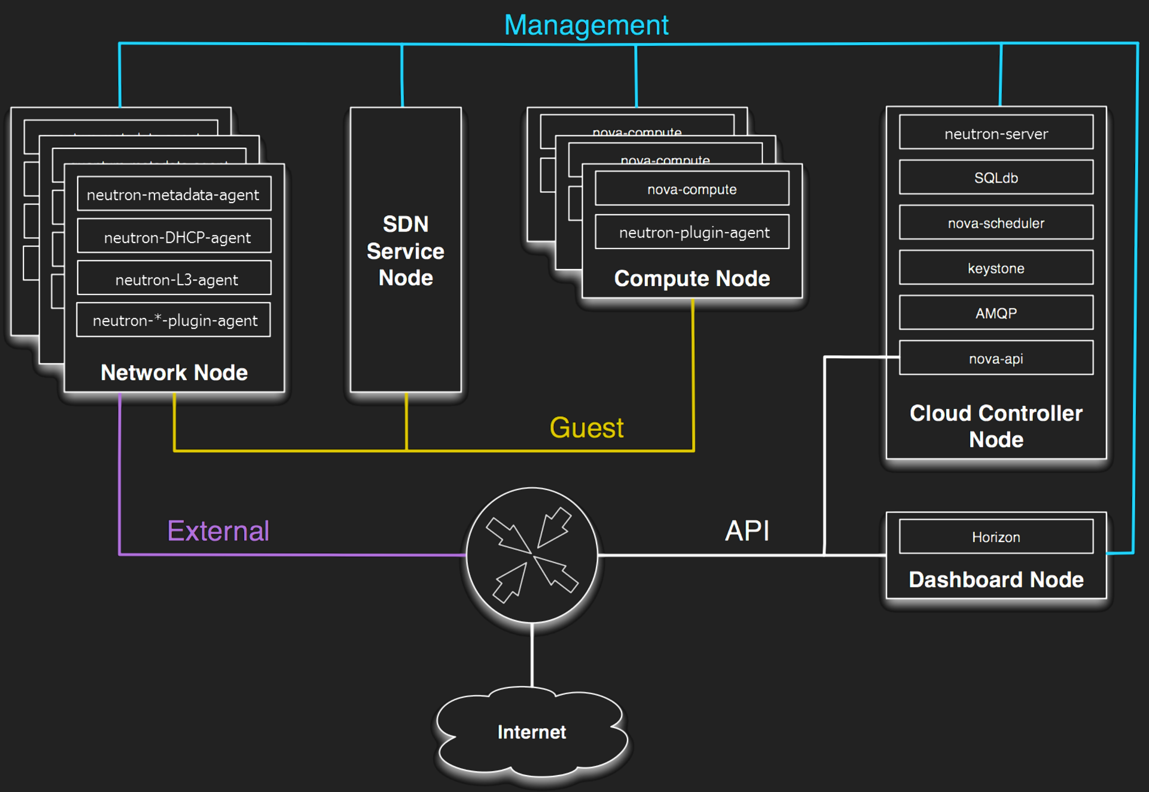
+



- 5 Physical interfaces to connect per compute node
  - IPMI (low bandwidth)
  - Two 10Gb/s interfaces (high bandwidth)
  - Two 1Gb/s interfaces (low bandwidth)
- **Keep good notes:** physical host interface <-> switchport mappings

LAYER 2 - START SIMPLY

```
interfaces:
    1:
        name: "Port 1"
        description: “cluster0 blade2 ipmi"
        native_vlan: ipmi-tenant-a-100
    2:
        name: "Port 2"
        description: “cluster0 blade2 eth1"
        native_vlan: mgmt-tenant-a-101
    3:
        name: "Port 3"
        description: “cluster0 blade2 eth10"
        native_vlan: guest-tenant-a-102
    4:
        name: “Port 4”
        description: “cluster0 blade2 eth11”
        native_vlan: ext-tenant-1-103
```



Routing is handled externally to Faucet:

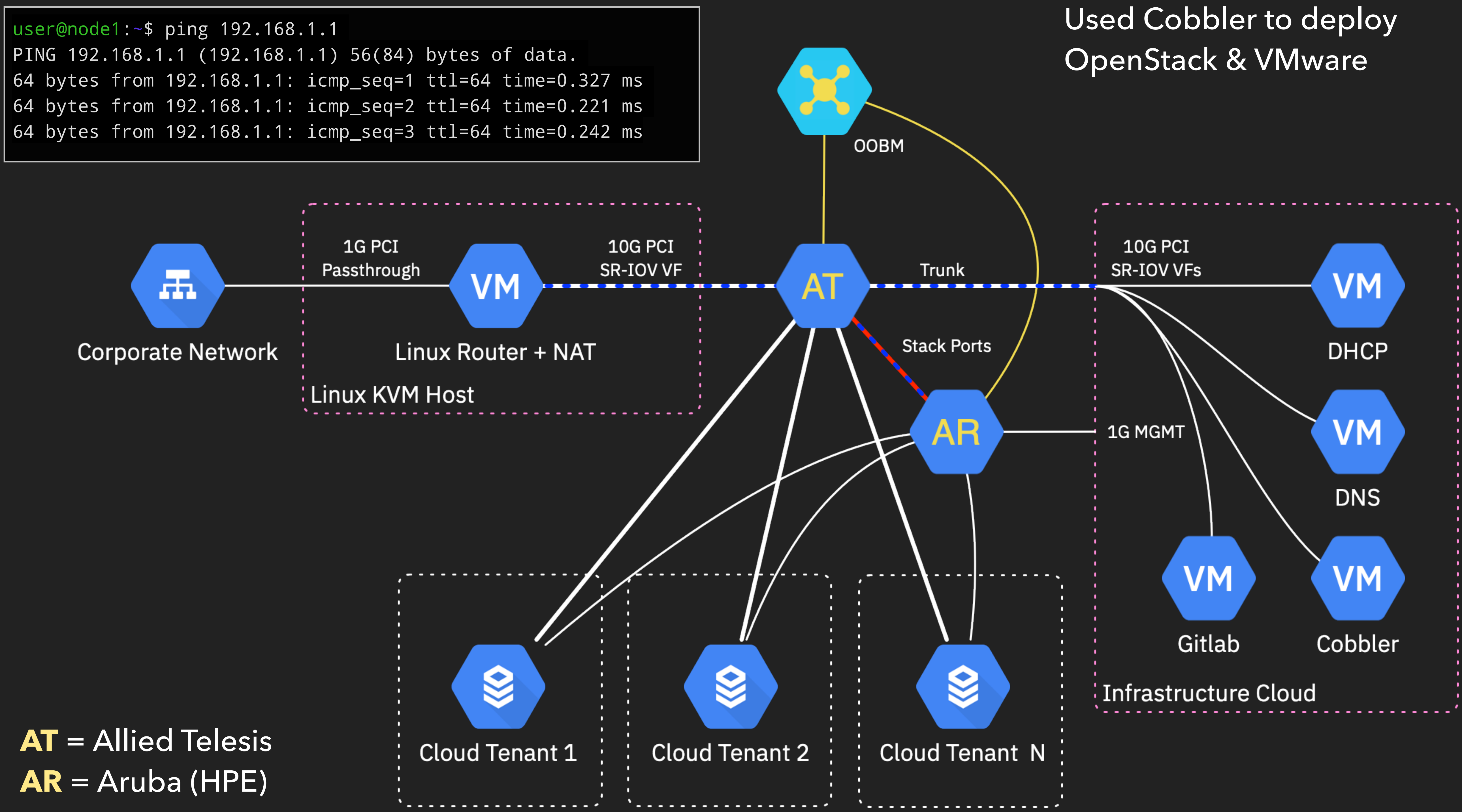
- Static route(s) pushed to a Linux-based “routing VM (+ NAT)”

Incredibly easy to automate:

- Ansible (FAUCET config generation), Python (pushing L3 routes to Linux VM), Cobbler (OS deployment)

Sandia National Laboratories

SIMPLE FAUCET LAYER 2 DESIGN: IT WORKS!



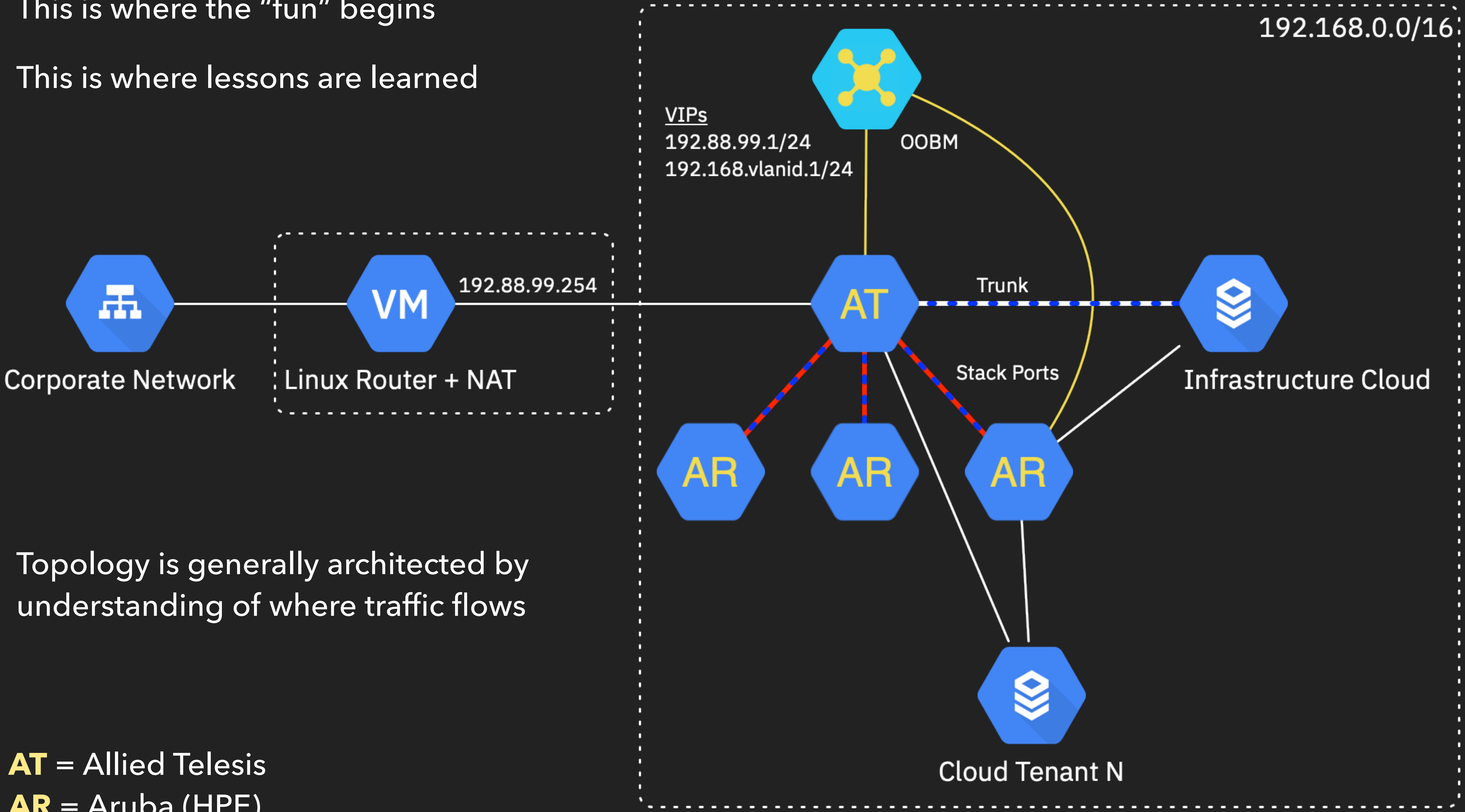
```
user@node1:~$ ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
64 bytes from 192.168.1.1: icmp_seq=1 ttl=64 time=0.327 ms
64 bytes from 192.168.1.1: icmp_seq=2 ttl=64 time=0.221 ms
64 bytes from 192.168.1.1: icmp_seq=3 ttl=64 time=0.242 ms
```

Used Cobbler to deploy OpenStack & VMware

AT = Allied Telesis
AR = Aruba (HPE)

NEXT STEP: L3 STACK ROUTING

- This is where the “fun” begins
- This is where lessons are learned



- Topology is generally architected by understanding of where traffic flows

AT = Allied Telesis
AR = Aruba (HPE)

* 192.88.99.0/24 - IANA Reserved: Deprecated (6to4 Relay Anycast)

TESTING SCRIPTS - ALL DONE INSIDE A VIRTUAL MACHINE

- Use the Faucet tutorial method, but for testing configs: **Open vSwitch** and **Network Namespaces**
- **./make_network.sh** - *based solely on how the human wired up the switches*

```
# Create DP atx950
sudo ovs-vsctl add-br atx950 \
        -- set bridge atx950 other-config:datapath-id=0xe01aea517a38 \
        -- set bridge atx950 other-config:disable-in-band=true \
        -- set bridge atx950 fail_mode=secure \
        -- set-controller atx950 tcp:127.0.0.1:6653 tcp:127.0.0.1:6654

# Create Trunk Ports between atx950 and ar2930f-1
sudo ovs-vsctl add-port atx950 atx950-p1 \
        -- set interface atx950-p1 type=patch \
        -- set interface atx950-p1 options:peer=ar2930f-1-p49 \
        -- set interface atx950-p1 ofport_request=1

sudo ovs-vsctl add-port ar2930f-1 ar2930f-1-p49 \
        -- set interface ar2930f-1-p49 type=patch \
        -- set interface ar2930f-1-p49 options:peer=atx950-p1 \
        -- set interface ar2930f-1-p49 ofport_request=49
```

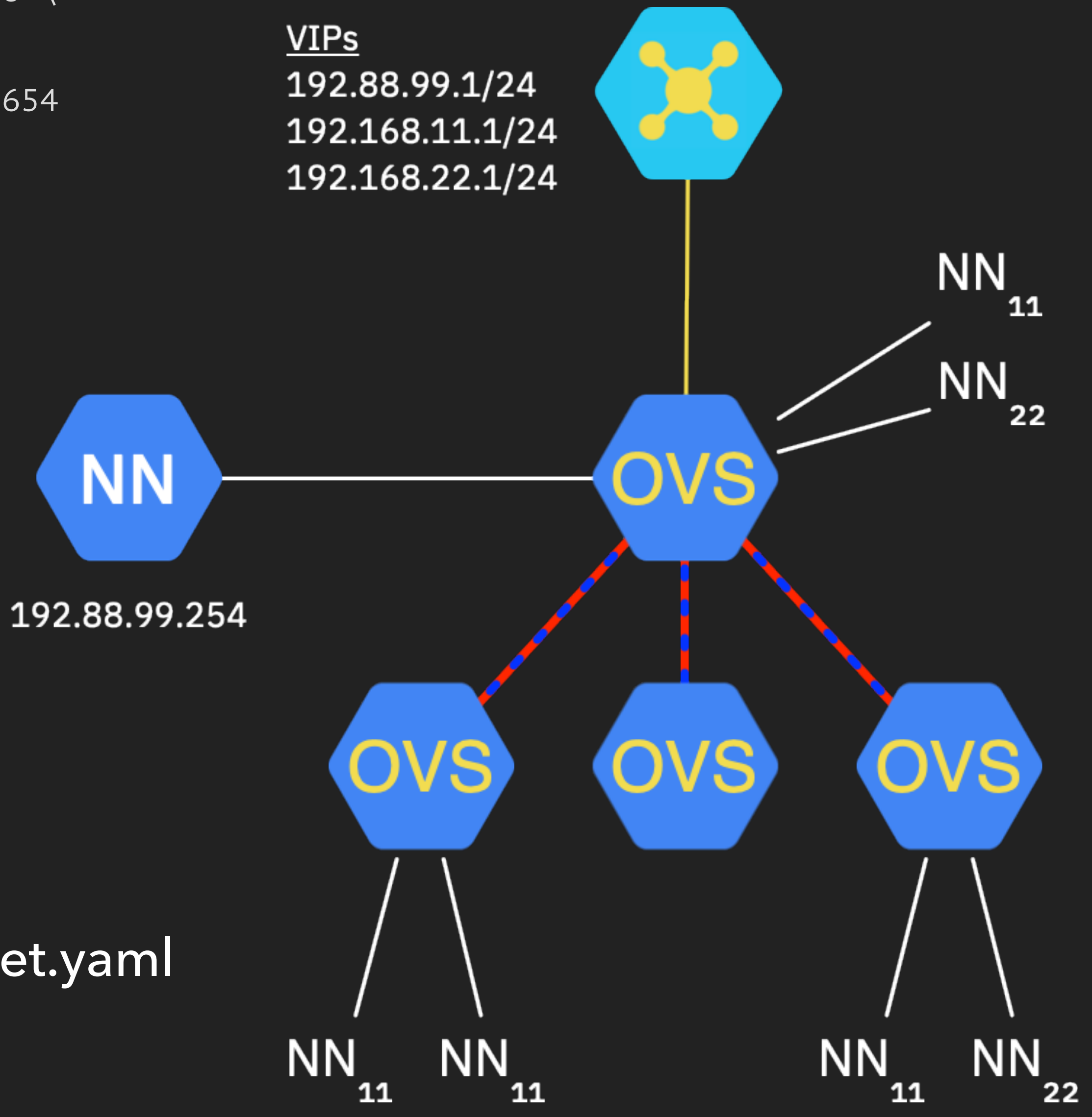
- **./make_hosts.sh**

```
# Create Test Namespaces and Connect them - atx950
create_ns host1 192.168.11.2/24
as_ns host1 ip route add default via 192.168.11.1
sudo ovs-vsctl add-port atx950 veth-host1 \
                    -- set interface veth-host1 ofport_request=5
```

- **./run_tests.sh**

```
# as_ns host3 ping -q -c3 192.168.10.2 > /dev/null
```

- Modify the "**`Hardware: Open vSwitch`**" line in faucet.yaml



HARDWARE ISSUES

```
faucet.valve ERROR    DPID 282562769570368 (0x100fd4581ca40) ar2930f-2 OFError type: OFPET_TABLE_FEATURES_FAILED
code: OFPTFFC_EPERM
version=0x4,msg_type=0x1,msg_len=0x4c,xid=0x3f6fcf42,OFPErrorMsg(code=5,data=bytearray(b’\x04\x12\x04\x18?
o\xcfB\x00\x0c\x00\x00\x00\x00\x00\x00\x00x\x00\x00\x00\x00\x00\x00port_acl\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'),type=13)Frame 653:
142 bytes on wire (1136 bits), 142 bytes captured (1136 bits)
```

```
Ethernet II, Src: ec:eb:b8:33:05:c0, Dst: 0c:c4:7a:54:a5:65
Internet Protocol Version 4, Src: 192.168.1.13, Dst: 192.168.1.60
Transmission Control Protocol, Src Port: 61578, Dst Port: 6653, Seq: 987130365, Ack: 1243404425, Len: 76
OpenFlow 1.3
    Version: 1.3 (0x04)
    Type: OFPT_ERROR (1)
    Length: 76
    Transaction ID: 3080866189
    Type: OFPET_TABLE_FEATURES_FAILED (13)
    Code: OFPTFFC_EPERM (5)
    Body: 04120428b7a2498d000c0000000000000088000000000000...
        Version: 1.3 (0x04)
        Type: OFPT_MULTIPART_REQUEST (18)
        Length: 1064
        Transaction ID: 3080866189
        Type: OFPMP_TABLE_FEATURES (12)
        Flags: 0x0000
        Pad: 00000000
        Table features
[Malformed Packet: openflow_v4]
    [Expert Info (Error/Malformed): Malformed Packet (Exception occurred)]
        [Malformed Packet (Exception occurred)]
        [Severity level: Error]
        [Group: Malformed]
```

JOSH: [RTFM] ENABLE DEBUGGING ON THE SWITCH

```
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:{ "error_code":"OFPTFFC_EPERM","error_reason":"There is no space available in the H/W to
accomodate the new pipeline","process_time":"0.166 ms","pipeline":[{"table_id":0,"name":"port_acl","
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:config":"0x3","max_entries":32,"metadata_match":"0x0","metadata_write":"0x0","match":
["in_port"],"wildcards":["in_port"],"next_tables":["1","5","6","7"],"instructions":["goto_table","apply_
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:actions"],"apply_actions":["output","pop_vlan","group"]},{"table_id":
1,"name":"vlan","config":"0x3","max_entries":288,"metadata_match":"0x0","metadata_write":"0x0","match":["in_port","vlan_
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:vid","eth_type","eth_dst/has_mask"],"wildcards":
["in_port","vlan_vid","eth_type","eth_dst"],"next_tables":["2"],"instructions":["goto_table","apply_actions"],"apply_setfield":["vlan_vid"],"
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:apply_actions":["output","set_field","pop_vlan","push_vlan","group"]},{"table_id":
2,"name":"eth_src","config":"0x3","max_entries":800,"metadata_match":"0x0","metadata_write":"0x0","match":[
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:"vlan_vid","in_port","eth_src","eth_type","eth_dst/has_mask"],"wildcards":
["vlan_vid","in_port","eth_src","eth_type","eth_dst"],"next_tables":["3","4","5","6","7"],"instructions":["goto_tab
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:le","apply_actions"],"apply_setfield":["vlan_vid","eth_dst"],"apply_actions":
["output","set_field","pop_vlan","push_vlan","group"],"next_tables_miss":["6"],"instructions_miss":["goto_table"
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:]},{"table_id":3,"name":"ipv4_fib","config":"0x3","max_entries":
608,"metadata_match":"0x0","metadata_write":"0x0","match":["vlan_vid","eth_type","ipv4_dst/has_mask"],"wildcards":["vlan_vid"
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:,"eth_type","ipv4_dst"],"next_tables":["5","6","7"],"instructions":
["goto_table","apply_actions"],"apply_setfield":["eth_dst","eth_src","vlan_vid"],"apply_actions":["output","set_field","po
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:p_vlan","push_vlan","group"]},{"table_id":4,"name":"ipv6_fib","config":"0x3","max_entries":
608,"metadata_match":"0x0","metadata_write":"0x0","match":["ipv6_dst/has_mask","vlan_vid","eth_typ
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:e"],"wildcards":["ipv6_dst","vlan_vid","eth_type"],"next_tables":
["5","6","7"],"instructions":["goto_table","apply_actions"],"apply_setfield":["eth_dst","eth_src","vlan_vid"],"apply_actions
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:":["output","set_field","pop_vlan","push_vlan","group"]},{"table_id":
5,"name":"vip","config":"0x3","max_entries":64,"metadata_match":"0x0","metadata_write":"0x0","match":["ip_proto","icmpv6
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:_type","arp_tpa","eth_dst","eth_type"],"wildcards":
["ip_proto","icmpv6_type","arp_tpa","eth_dst","eth_type"],"next_tables":["6","7"],"instructions":["goto_table","apply_actions"],"apply_act
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:ions":["output","pop_vlan","group"]},{"table_id":
6,"name":"eth_dst","config":"0x3","max_entries":800,"metadata_match":"0x0","metadata_write":"0x0","match":["vlan_vid","eth_dst"],"instructio
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:ns":["apply_actions"],"apply_actions":["output","pop_vlan","group"],"next_tables_miss":
["7"],"instructions_miss":["goto_table"]},{"table_id":7,"name":"flood","config":"0x3","max_entries":96
Sep 23 13:14:34 192.168.1.12 OPFL: OPFL eOFNetTask:,”metadata_match":"0x0","metadata_write":"0x0","match":["in_port","vlan_vid","eth_dst/
has_mask"],"wildcards":["in_port","vlan_vid","eth_dst"],"instructions":["apply_actions"],"apply_actions
```

Lessons:

- Read the documentation. Understand it.
- Setup a syslog server to capture OF messages and errors from switches

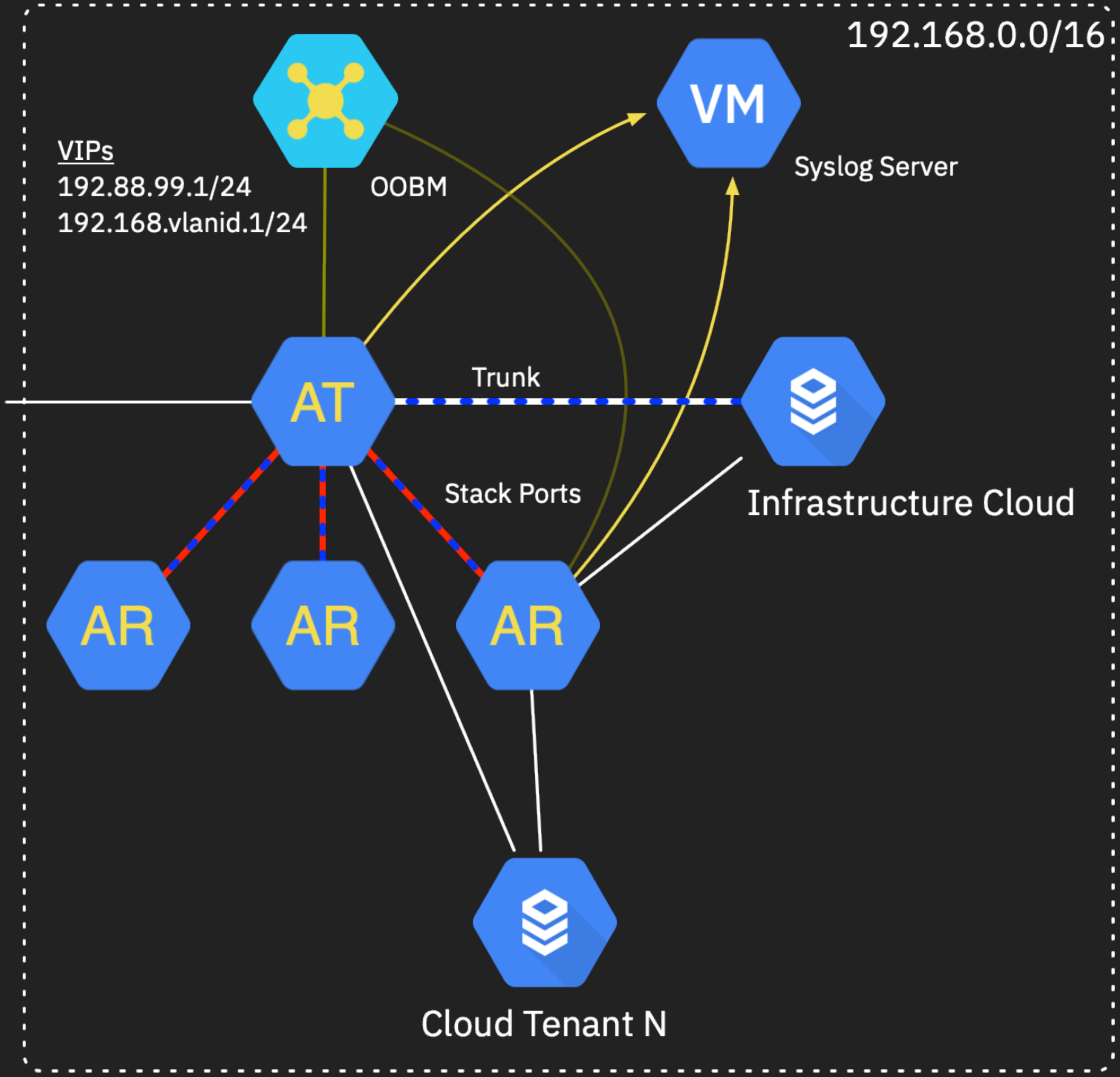
ADD A SYSLOG SERVER TO THE OOBM NETWORK

```
ar2930f-2# show openflow instance aggregate flow-table

 OpenFlow Instance Flow Table Information

 Table                          Flow     Miss
 ID    Table Name               Count    Count          Goto Table
 ----- --------------------- -------- ------------- -------------
 0     port_acl                 1        0              1, 5, 6, 7
 1     vlan                     1        0              2
 2     eth_src                  16       0              3, 4, 5, 6, 7
 3     ipv4_fib                 3        0              5, 6, 7
 4     ipv6_fib                 6        0              5, 6, 7
 5     vip                      17       0              6, 7
 6     eth_dst                  1        0              *
 7     flood                    14       0              *
```

- TFM = **Table Features Message**

OpenFlow Specification 1.3
7.3.5.5 Table Features

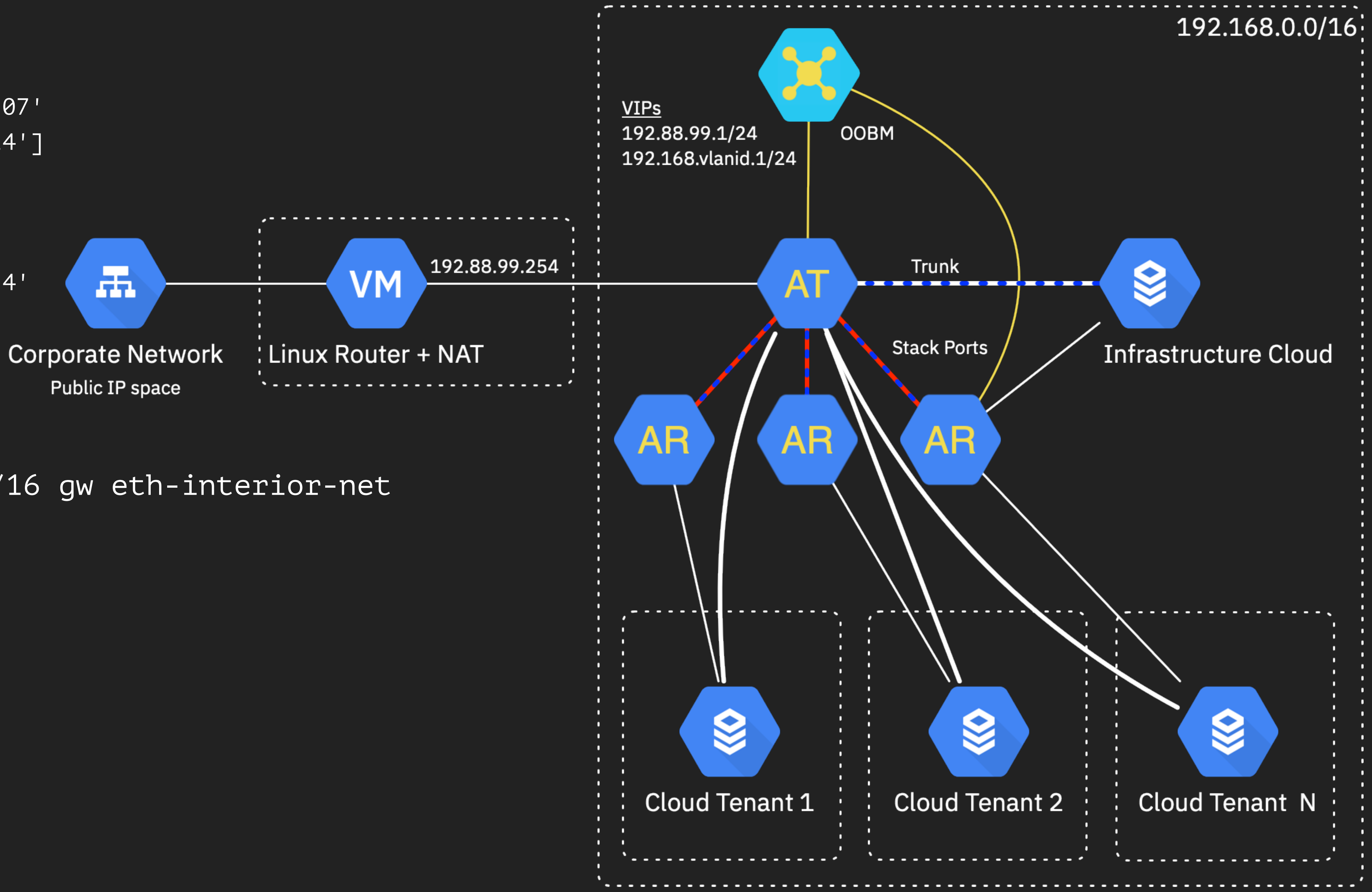
The OFPMP_TABLE_FEATURES multipart type allows a controller to both query for the capabilities of existing tables, and to optionally ask the switch to reconfigure its tables to match a supplied configuration.



IT WORKS! (AGAIN)

Faucet vlans.yaml:

```
vlans:
    routing:
        vid: 7
        description: "Gateway Net"
        faucet_mac: 'de:ad:be:ef:00:07'
        faucet_vips: ['192.88.99.1/24']
        routes:
            - route:
                ip_dst: "0.0.0.0/0"
                ip_gw: ‘192.88.99.254'
```



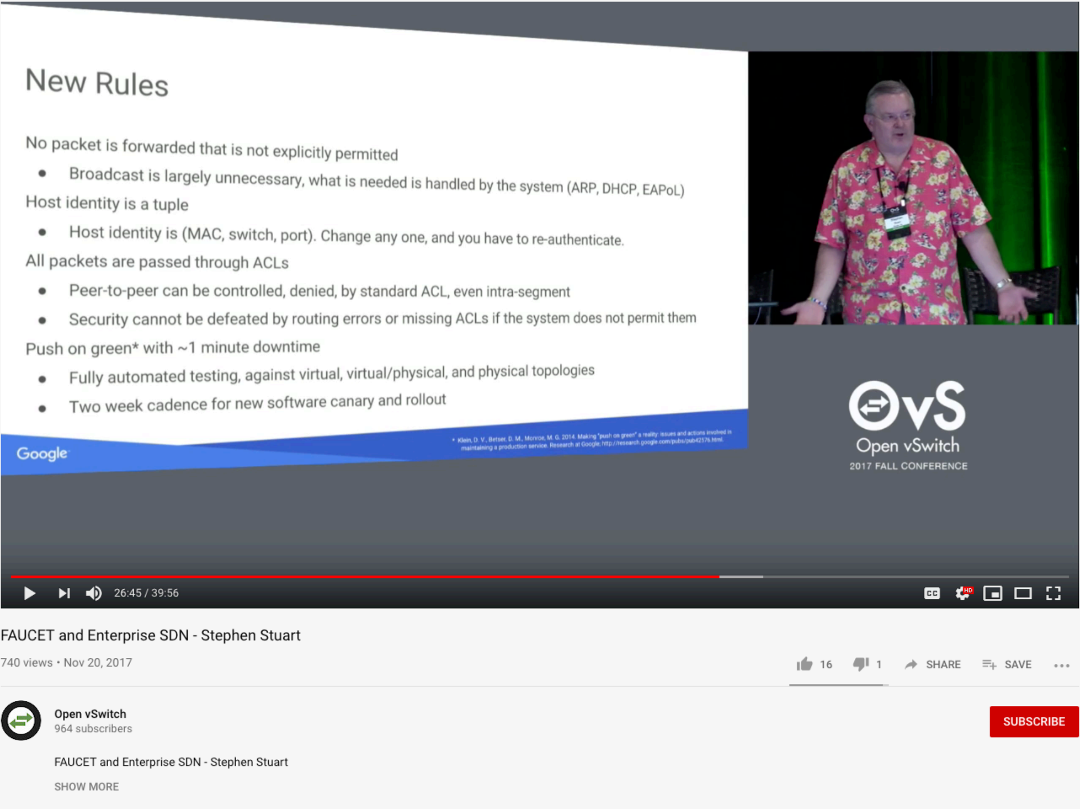
Linux Router:

```
# route add -net 192.168.0.0/16 gw eth-interior-net
```

LESSONS LEARNED (1 OF 3)

- Getting up and running with Faucet is super easy - follow the tutorial
  - Anther required reading: http://docs.openvswitch.org/en/latest/tutorials/faucet/
- Open vSwitch is awesome!
  - 90%+ of design + testing can be done with software
- Allied Telesis hardware has been rock solid
- **Start small, grow slowly**
  - ***Follow the packet!***
  - Try out every feature available
  - Keep an open mind. Many traditional networking concepts apply, many don’t https://www.youtube.com/watch?v=BDje6HGBwso Go to 23:10 mark
- Don’t run your controller off a switch it’s controlling (so much for a 100% OF controlled network)

  ¯_(ツ)_/¯



New Rules

No packet is forwarded that is not explicitly permitted

- Broadcast is largely unnecessary, what is needed is handled by the system (ARP, DHCP, EAPoL)

Host identity is a tuple

- Host identity is (MAC, switch, port). Change any one, and you have to re-authenticate.

All packets are passed through ACLs

- Peer-to-peer can be controlled, denied, by standard ACL, even intra-segment
- Security cannot be defeated by routing errors or missing ACLs if the system does not permit them

Push on green* with ~1 minute downtime

- Fully automated testing, against virtual, virtual/physical, and physical topologies
- Two week cadence for new software canary and rollout

* Klein, D. V., Betser, D. M., Monroe, M. G. 2014. Making "push on green" a reality: issues and actions involved in maintaining a production service. Research at Google. http://research.google.com/pubs/pub42576.html

FAUCET and Enterprise SDN - Stephen Stuart

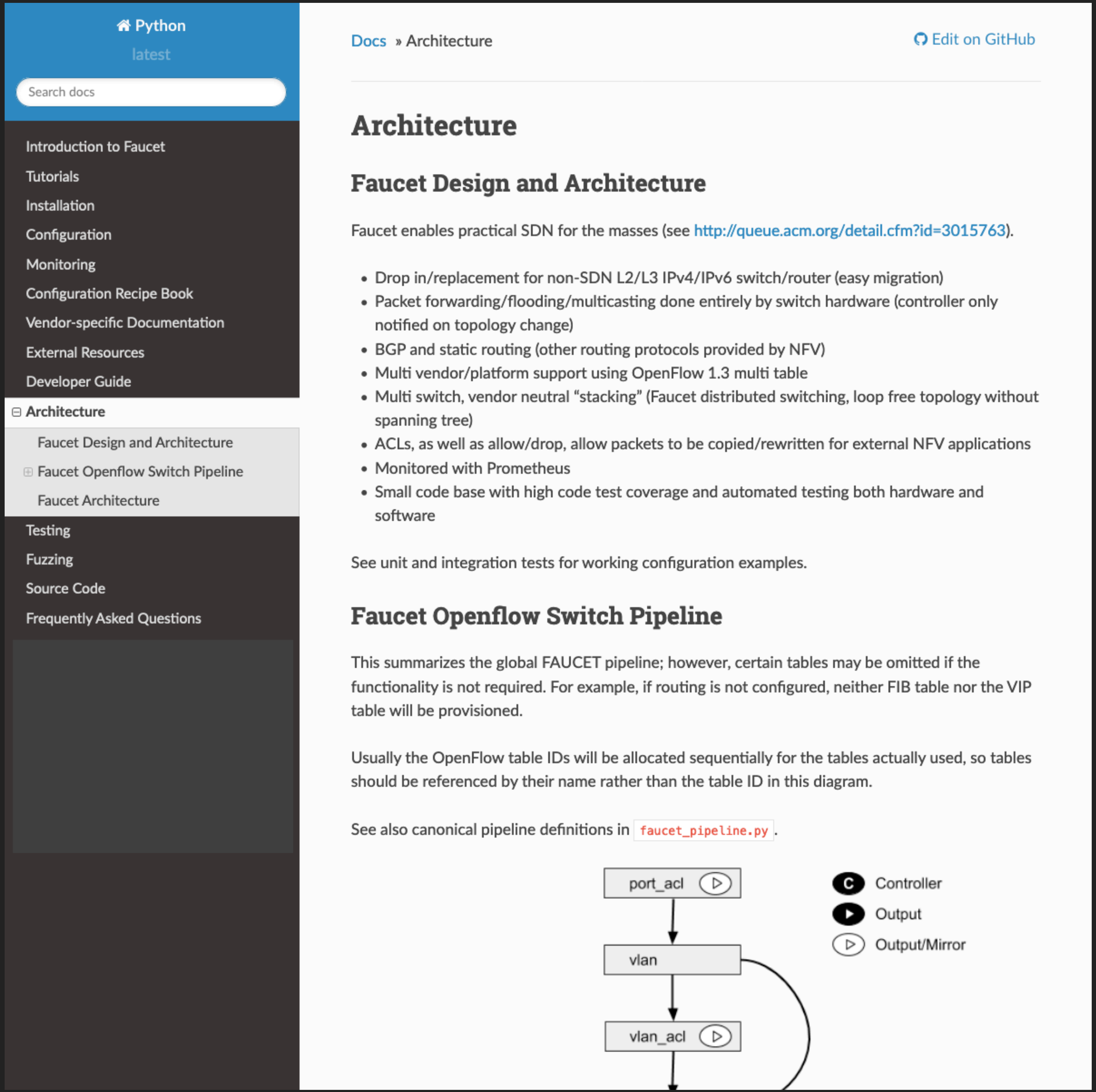
740 views • Nov 20, 2017

Open vSwitch
964 subscribers

FAUCET and Enterprise SDN - Stephen Stuart

LESSONS LEARNED (2 OF 3)

- Highlights from the documentation
  - Faucet Design and Architecture - **Faucet Openflow Switch Pipeline**
  - Vendor-Specific documentation (Allied Telesis, Aruba, etc.)
- OpenFlow 1.3.x Specification
- Diagrams - need more
  - Began our own stash of configs and associated diagrams



Architecture

Faucet Design and Architecture

Faucet enables practical SDN for the masses (see http://queue.acm.org/detail.cfm?id=3015763).

- Drop in/replacement for non-SDN L2/L3 IPv4/IPv6 switch/router (easy migration)
- Packet forwarding/flooding/multicasting done entirely by switch hardware (controller only notified on topology change)
- BGP and static routing (other routing protocols provided by NFV)
- Multi vendor/platform support using OpenFlow 1.3 multi table
- Multi switch, vendor neutral "stacking" (Faucet distributed switching, loop free topology without spanning tree)
- ACLs, as well as allow/drop, allow packets to be copied/rewritten for external NFV applications
- Monitored with Prometheus
- Small code base with high code test coverage and automated testing both hardware and software

See unit and integration tests for working configuration examples.

Faucet Openflow Switch Pipeline

This summarizes the global FAUCET pipeline; however, certain tables may be omitted if the functionality is not required. For example, if routing is not configured, neither FIB table nor the VIP table will be provisioned.

Usually the OpenFlow table IDs will be allocated sequentially for the tables actually used, so tables should be referenced by their name rather than the table ID in this diagram.

See also canonical pipeline definitions in `faucet_pipeline.py`.

Sandia National Laboratories

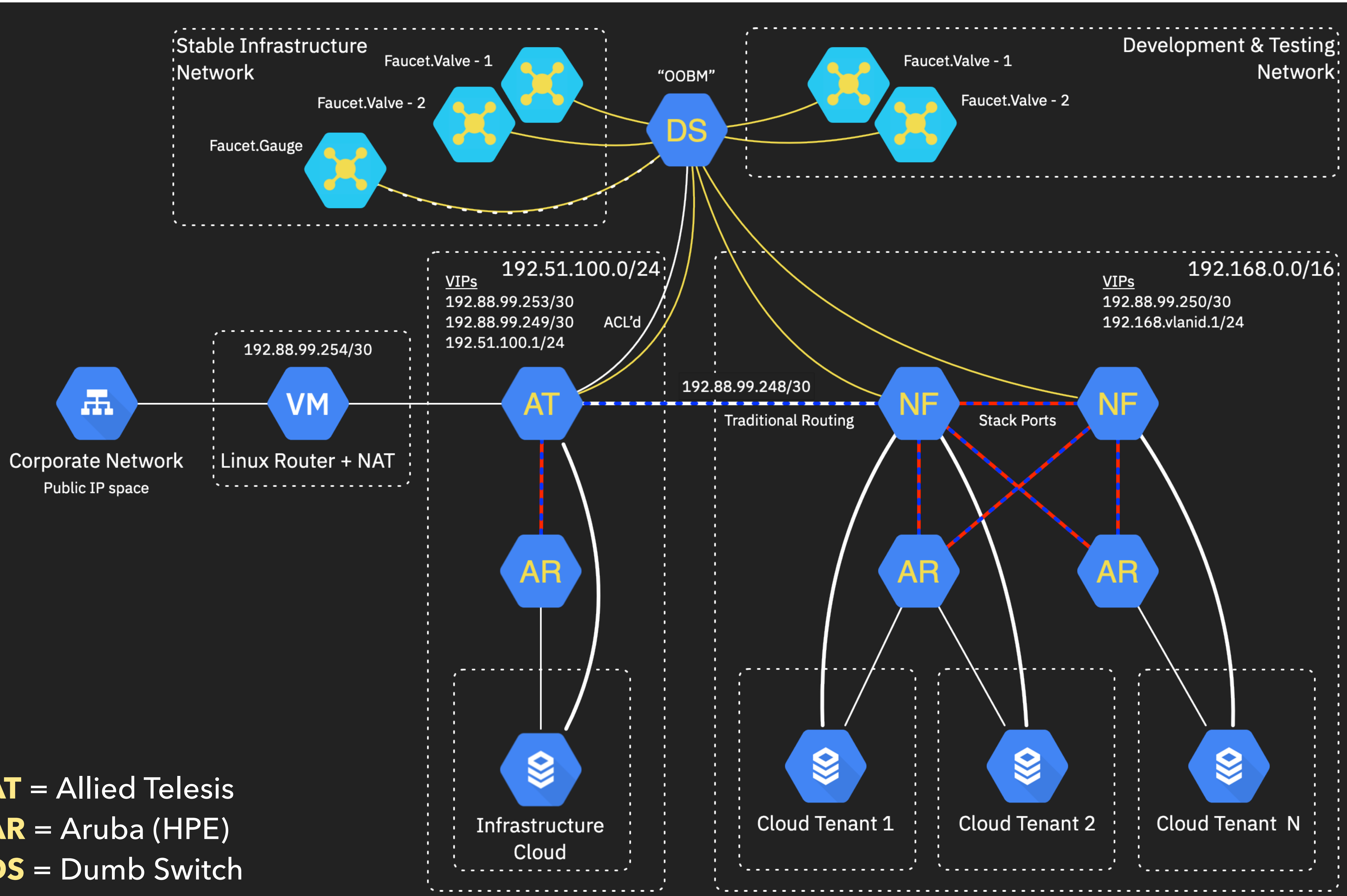
LESSONS LEARNED (3 OF 3)

- **Collect logs** - as much as you can
  - Save faucet.log files on your controller(s)
  - Collect PCAP on your OpenFlow channel (assumes non-TLS)
  - debug openflow on switches to a syslog server
- **Follow a software development mindset**
  - Run tests against configs with OVS, **then with hardware**
  - Script/automate as much as possible
- **Not all switching hardware is created equally**
  - Brad quote from ONS2019: “I learned a lot more about vendor hardware architectures than I expected to.”

STATUS OF OCT 2019

- ~300 physical interfaces, 4 switches controlled via Faucet in a stacked, L3 routing architecture
- Network is quiet, efficient, reconfigurable. **Awesome!**
- Controller and switch management interfaces are [almost] completely out-of-band (~7 switch ports)
  - Had a few oddities when bridging legacy [dumb] switch into OF-controlled network
  - OOB controller network is Faucet routed, but heavily ACL'd - desire for Gauge/Prometheus/Grafana
- HW limitation errors fixed Aruba switches (thanks, Josh!) - See '**port_table_scale_factor**' feature
  - Built a second, smaller testbed for running new Faucet configs on Aruba hardware
- **Re-designing the architecture (again)**
  - Integrating EdgeCore/NoviFlow switches into the network
  - Adding additional controllers: redundancy & segmentation (based upon availability needs)
- Writing our own Ryu app for collecting OF messages and querying switch features
- Want to spend more time with Poseidon

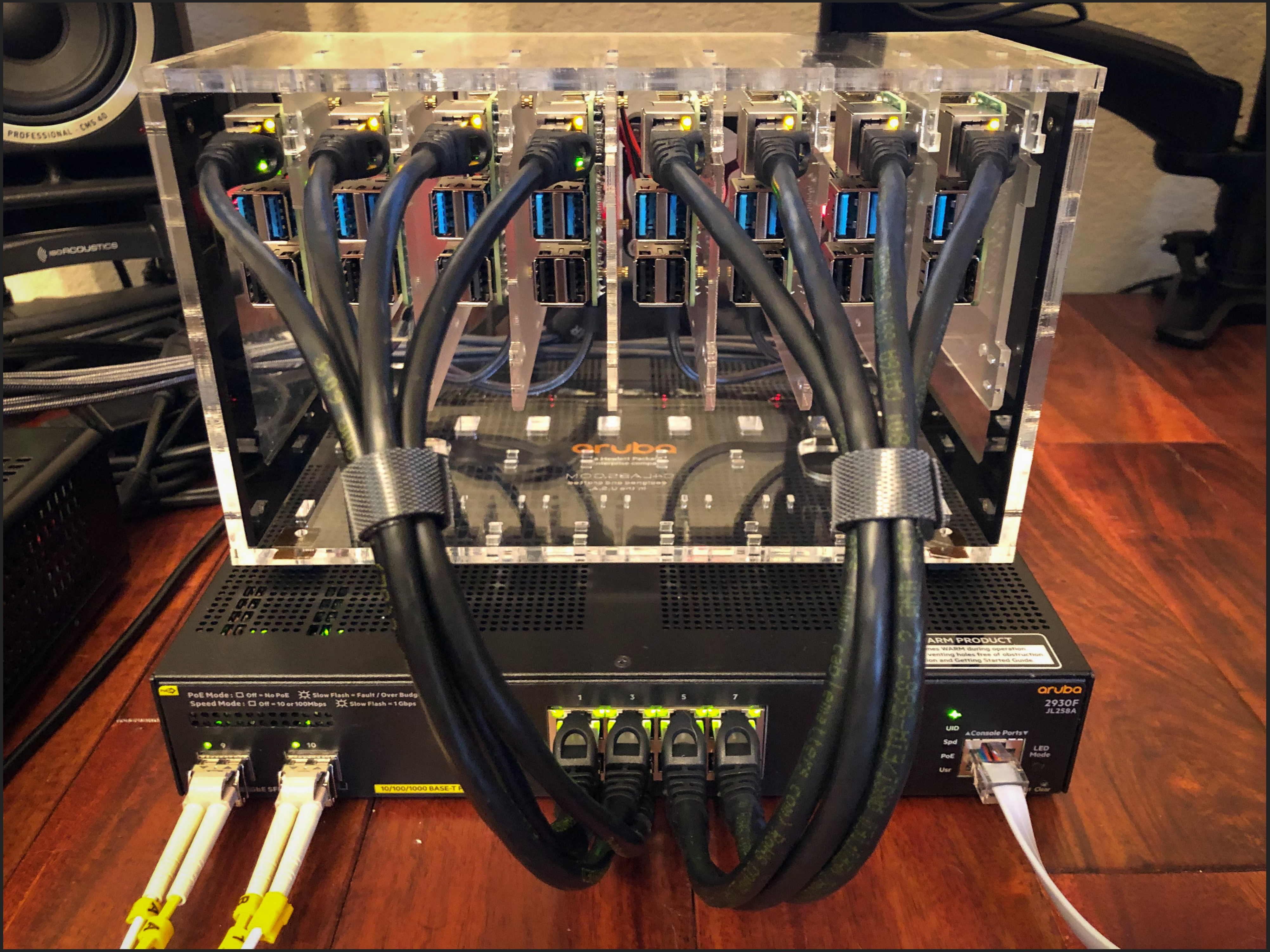
PATH FORWARD



AT = Allied Telesis
AR = Aruba (HPE)
DS = Dumb Switch
NF = NoviFlow

* 192.51.100.0/24 - IANA Reserved: Documentation (TEST-NET-2)

BONUS! - RASPBERRY PI 4 SDN CLUSTER



SPECIAL THANKS

- Nick Buraglio
- Brad Cowie
- Josh Bailey
- mab68 - don’t know who you are, but THANK YOU for your commits
- Rest of the FAUCET team
- Open vSwitch team
- SNL networking team - Rick Strong, David Burton, Will Stout