deploying high capacity IP fabrics
How can I build a network of 100 switches in 5 minutes?
Bringing the Linux revolution to Networking!

Traditional Network Gear
Networking Gear and Networking OS locked onto one vendor

New Paradigm
Disaggregate the Networking Gear from the OS
Bring Linux OS to networking: Cumulus Linux
Industry standard switch hardware

- 1RU height
- Redundant power supply units and fan trays
- Directional airflow (cable side inlet) or (PSU side inlet)

Click here for more information on Cumulus Networks hardware support: http://cumulusnetworks.com/support/hcl/
Cumulus™ Linux® IS Linux

- Native Linux experience
- Linux applications now work on networking gear

**User Space**
- Routing Suite
- Overlay Network
- Routing Protocols
- Automation
- Monitoring
- 3rd Party/Customer Applications

**Switch HAL**
- switchd
- Switch Driver
- Switch Silicon
  - Front Panel Ports
- Ethernet Interfaces
- Bridge Table
- ARP Table
- Routing Table

**Linux Kernel**
- CPU, RAM, Flash, etc

**Diagram Notes:**
- "The registered trademark Linux® is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a world-wide basis."
Networking hardware needs an installer for flexibility and economies of scale
  • (i)PXE equivalent for networking HW

ONIE – installer environment for open networking hardware
  • small, Linux OS pre-installed on switches
  • environment for network OS installers
    network operating system neutral
  • stays resident for un-install, re-install, etc

Discovery and execution
  • v6ND… DHCPv6…DHCPv4

Modern awesomeness
  • http, ssh, syslog, etc

https://github.com/onie
First boot – Current state

ODM
- ONIE pre installed by ODM
- Ship hardware thru supply chain

Customer
- Unpack and rack switch
- Connect to management network
- Power up switch
- ONIE discovers Cumulus Linux
- Cumulus Linux first boot

What next?
Zero-touch install

Right-size and provision devices

- credentials
- users
- logging & monitoring
- operational packages
- configuration

1- Discover installer on management interface:
   USB, URL from DHCP, URL from DNS-SD, ....

2- Download, run installer, and reboot
dhcp configuration

ddns-update-style none;

default-lease-time 600;
max-lease-time 7200;

authoritative;

log-facility local7;

option www-server code 72 = ip-address;

subnet 10.99.0.0 netmask 255.255.255.0 {
  range 10.99.0.10 10.99.0.250;
  option routers 10.99.0.1;
  option domain-name-servers 10.99.0.1;
  option domain-name "provisioning.local";
  option www-server 10.99.0.1;
}
/bin/bash

# update and install puppet
apt-get -q -y update
apt-get -q -y upgrade
apt-get -qq install puppet

# set the hostname
SWHOSTNAME=demo-`facter macaddress | sed "s/://g" -c < /dev/urandom tr -dc a-z0-9 | head -c4`
echo $SWHOSTNAME >/etc/hostname
echo "127.0.0.1 $SWHOSTNAME" >> /etc/hosts
hostname $SWHOSTNAME

invoke-rc.d hostname.sh start

# config puppet
sed -i /etc/default/puppet -e 's/START=no/START=yes/'
service puppet restart

# CUMULUS-AUTOPROVISIONING

exit 0
import "classes".pp

# unassigned nodes get this
node default {
  if $operatingsystem == "CumulusLinux" {
    include switchbase
  }
}

class switchbase {
  include users
  include sudo
  include motd
  include licenseinstall
}

class users {
  add_user { nat:
    email => "nat@cumulusnetworks.com", uid => 5001, shell => "/bin/bash", groups => [],
    password => "$6$v3GTFaOh$8oyYyqafHMGtNi9ZQzg9OQGJe2mZyp1WY22c/04l3GJeVxGmxpAgOIHUKGJF0otU1O9b409W3WURYa."
  }

  add_ssh_key { nat:
    type => "ssh-dss", key=> "...snip..."
  }
}

class sudo {
  package { sudo:
    ensure => latest
  }

  file { "/etc/sudoers":
    owner => root,
    group => root,
    mode => 440,
    source => "puppet:///files/sudoers",
    require => Package["sudo"]
  }
}

class motd {
  file { "/etc/motd":
    owner => root,
    group => root,
    source => "puppet:///files/motd"
  }
}

class licenseinstall {
  file { "/etc/cumulus/mylicense.txt":
    owner => root,
    group => root,
    source => "puppet:///files/mylicense.txt"
  }

  exec { "/usr/cumulus/bin/cl-license -i /etc/cumulus/mylicense.txt":
    subscribe => File["/etc/cumulus/mylicense.txt"],
    refreshonly => true
  }
}
Prescriptive Topology Module

- **Goal:** Operational simplicity, reduced Operator errors
  - Verify connectivity per cabling plan
  - Bring up routing adjacency only if cabling test passes
  - Selective actions on link up

- **How?** Network topology specified via DOT language and distributed to all nodes
  - Each node determines its relevant information
  - Use LLDP to verify connectivity
  - Logs errors
  - Daemon executes a set of scripts on topology pass and a different set of scripts on topology fail

```
Graph G {
  S1:p1 – M1:p3;
  S1:p2 – M2:p3;
  S1:p3 – M3:p3;
  S1:p4 – M4:p3;
  S2:p1 – M1:p4;
  S2:p2 – M2:p4;
  S2:p3 – M3:p4;
  S2:p4 – M4:p4;
  M1:p1 – T1:p1;
  M1:p2 – T2:p2;
  M4:p2 – T4:p2;
}
```
digraph G {
  //spine0's connections
  spine0:swp1 -> leaf0:swp1;
  spine0:swp2 -> leaf1:swp1;

  //spine1's connections
  spine1:swp1 -> leaf0:swp2;
  spine1:swp2 -> leaf1:swp2;

  //leaf0's connections
  leaf0:swp1 -> spine0:swp1;
  leaf0:swp2 -> spine1:swp1;

  //leaf1's connections
  leaf1:swp1 -> spine0:swp2;
  leaf1:swp2 -> spine1:swp2;
}

Demo

Topology
Monitoring & Troubleshooting

• Native Linux troubleshooting & monitoring utilities
  • SNMP, syslog, log files, traceroute, tcpdump…

• Wire-rate tools such as SPAN/ERSPAN

• Hardware counters
  • ethtool stats, queue-depth, buffer utilization

• Packages tested for monitoring
  • e.g. collectd, netsnmp

• Advanced troubleshooting with PTMd
Thank you!