CUMULUS

OSAD München

Simple Open Networking
&
Automation

15-OCT-2019

Dipl.-Ing. Andreas la Quiante
alq@cumulusnetworks.com

Pre-Sales Systems Engineer | Cumulus Networks
CNX$_{1999}$, …, CCIE , … CCONP$_{2019::9}$

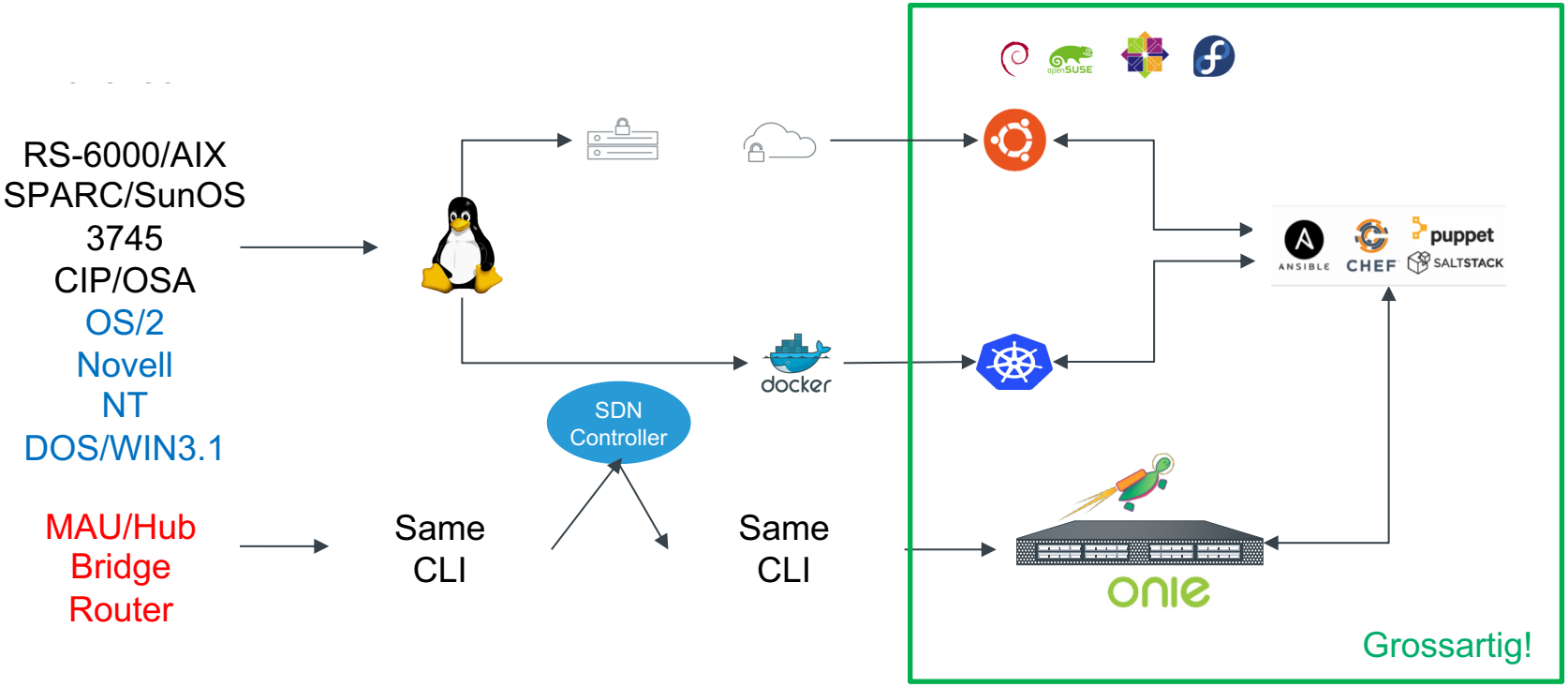
Version 007

Agenda / Content:

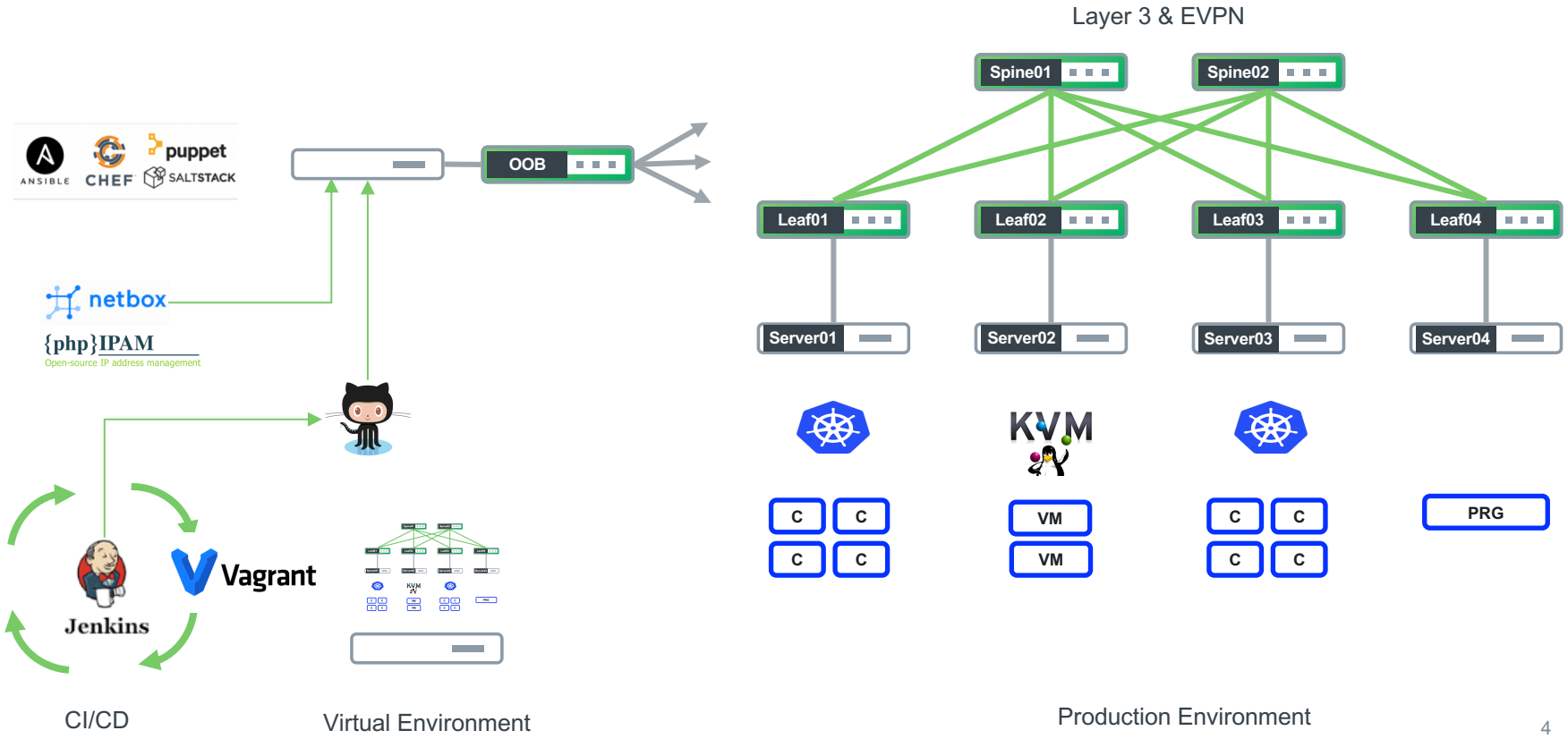
- History
- Motivation
- The Scope

- Linux Networking
- Open Networking
- Container Networking
- Automation/Orchestration

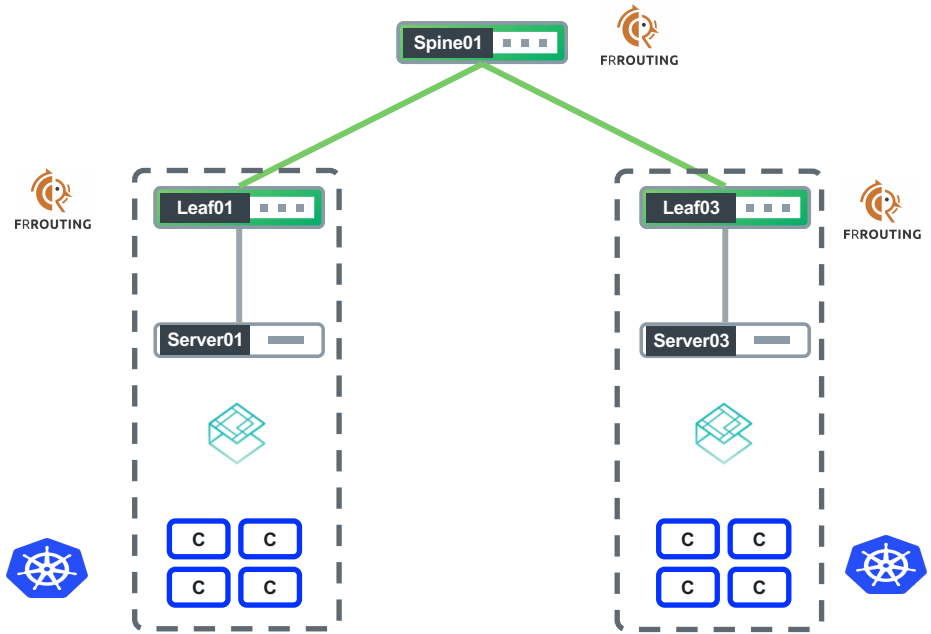
a quick private history lesson

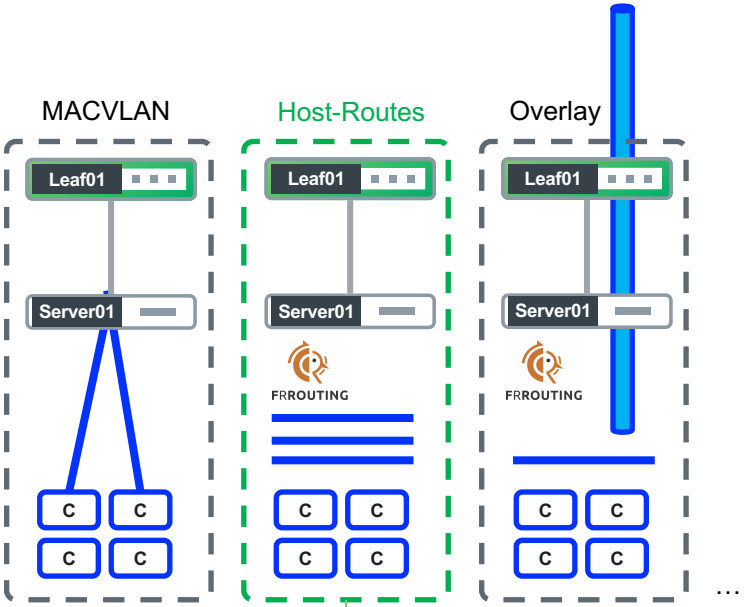


Motivation


Layer 3 & EVPN
Spine01
Spine02
Leaf01
Leaf02
Leaf03
Leaf04
Server01
Server02
Server03
Server04
OOB
ANSIBLE
CHEF
puppet
SALTSTACK
netbox
{php}IPAM
Open-source IP address management
Jenkins
Vagrant
KVM
C
VM
PRG
CI/CD
Virtual Environment
Production Environment

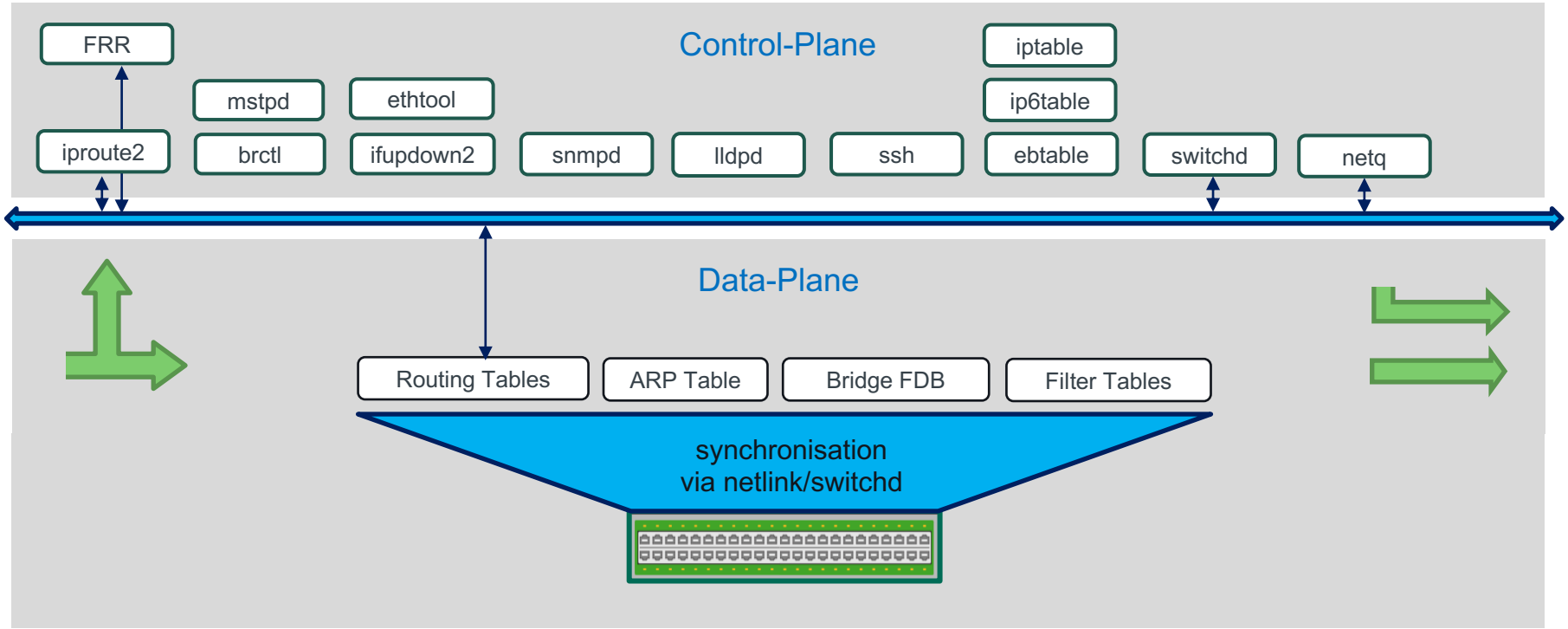
Motivation





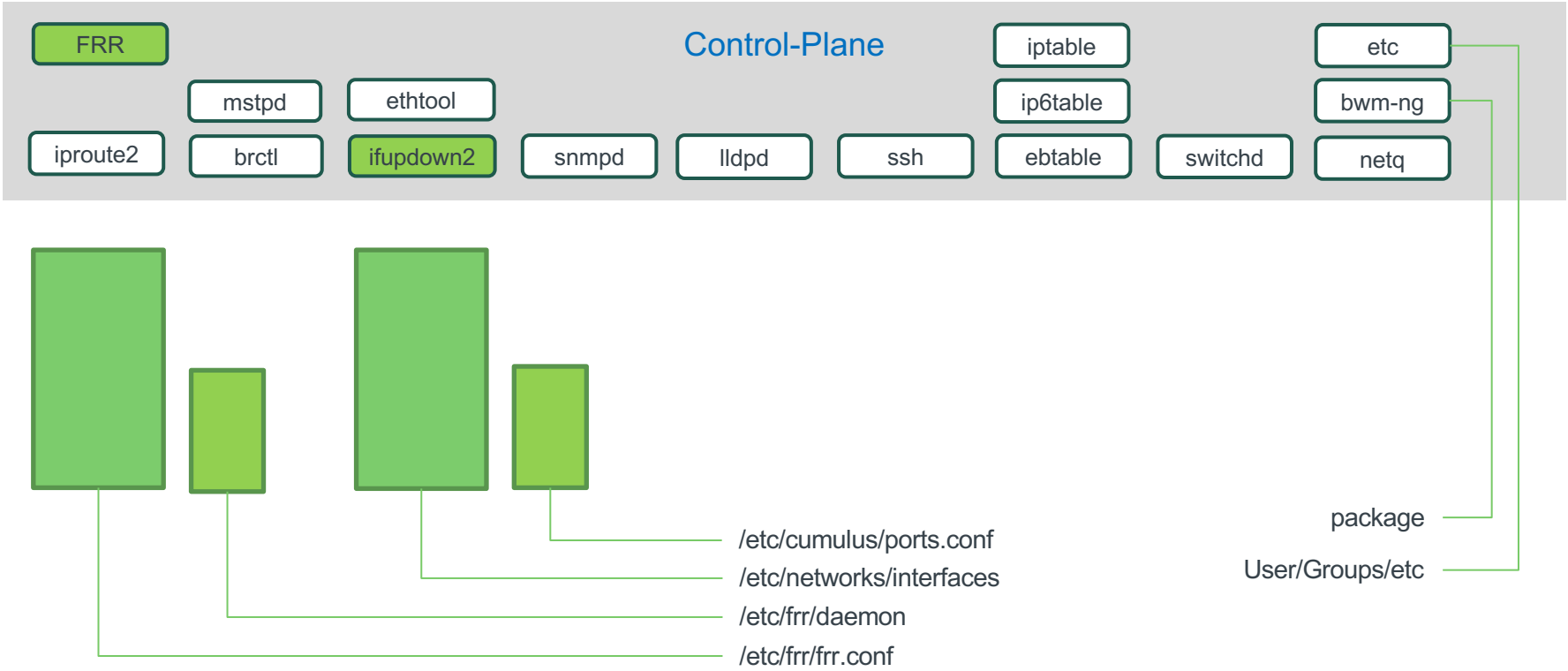
Let‘s have a look
at the OS architecture

Architecture (Linux / Cumulus Linux)



6

Automatisation / Orchestration



7

Side-Note

- Options: Network CLI

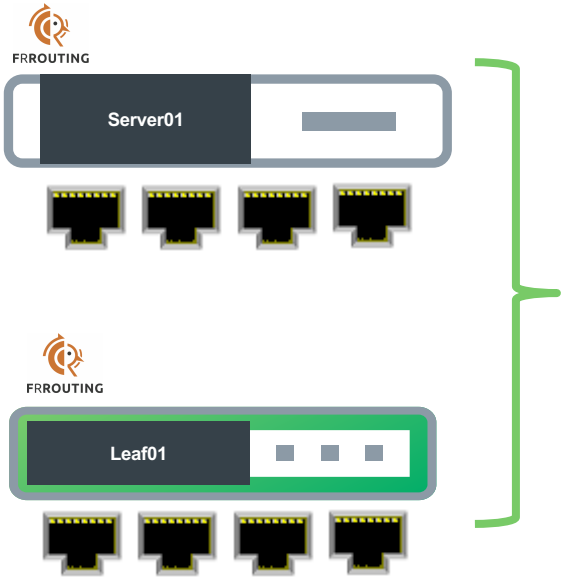
```
$ net add interface swp1 ip address 10.1.3.11/24
$ net pending
$ net commit
```

```
- hosts: leaf01
  tasks:
  - name: Phys.Schnittstelle
    nclu:
      commands:
        - add interface swp1 ip address 10.1.3.11/24
```

“What“ do we REALLY need to automate?

- Use-Case: Server network-interfaces {place-holder} packages

- Use-Case: Switch network-interfaces routing-protocol(s) packages

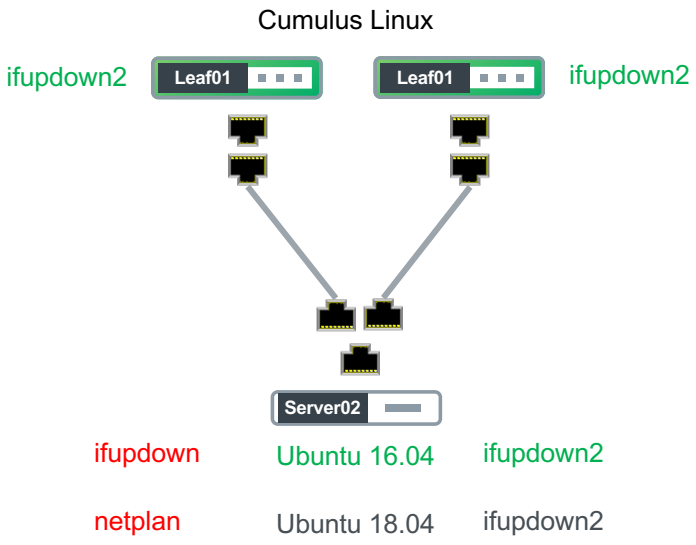



```
hosts: server-switch
   name: example
   become: yes
   tasks:
   -  name: Ifs
      copy:
        src: /…
        dst: /etc/network/interfaces
```

and a bit of magic

Interface Manager

- New implementation in Python
- Backward compatibility with debian ifupdown
- Pluggable architecture with python add-on modules
- Ordered Network Interface dependency relationship handling
- Incremental changes and query live configuration




```
# git clone git://github.com/CumulusNetworks/ifupdown2
# sudo apt-get install make
# cd ifupdown2 && git checkout master-next && make deb
# sudo dpkg -i ./ifupdown2_1.2.1_all.deb
```

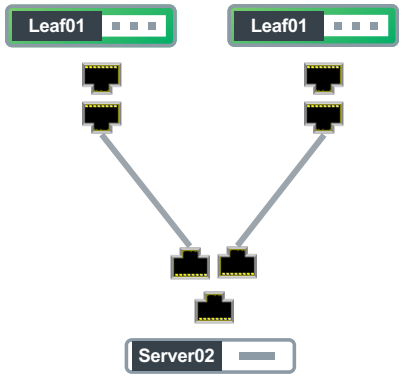
$ sudo ifquery

Dependencies: demo leaf01

```
cumulus@leaf01:mgmt-vrf:~$ ifquery -a -p list
lo : []
eth0 : []
mgmt : ['eth0']
swp1 : []
swp2 : []
TEST : ['swp2']
swp51 : []
bridge : ['swp1', 'vni-10010']
vni-10010 : []
vlan10 : ['bridge']
```

Interface Manager

```yaml
- hosts: spine01
  name: Konfiguration der IF
  become: yes
  tasks:
    - name: IFs for BGP unnumbered
      copy:
        src: /home/cumulus/on/on-001/on-spine01-001
        dest: /etc/network/interfaces
    # aktivieren
    - name: Durchstarten der Schnittstelle
      shell: /sbin/ifreload -a
```



Template & Interfaces

stolen from Eric

```yaml
---
- name: Load Interface Configuration
  copy: src=config/{{ansible_hostname}}/interfaces dest=/etc/network/

- name: Apply Interface Configuration Delta
  shell: nohup bash -c 'sleep 2 && /sbin/ifreload -a > /tmp/ifreload.out 2>&1' &
  async: 1
  poll: 0
  changed_when: true
```

stolen from Attilla

```
⌥⌘4                                ssh
  GNU nano 2.5.3              File: interfaces.j2

###############
# Ansible generated config, will be overwritten!
###############

###############
# General
###############
auto lo
iface lo inet loopback
   address {{node[inventory_hostname]["routing"]["lo"]}}
{% if node[inventory_hostname]["overlay"] is defined %}
{% if node[inventory_hostname]["overlay"]["anycast-ip"] is defined %}
   clagd-vxlan-anycast-ip {{ node[inventory_hostname]["overlay"]["anycast-ip"]|$
{% endif %}
{% endif %}

auto eth0
iface eth0 inet dhcp
  vrf mgmt

auto mgmt
iface mgmt
  address 127.0.0.1/8
  vrf-table auto

###############
# VRFs / L3VNI
###############

{% if node[inventory_hostname]["overlay"] is defined %}
{% set vrfs = [] %}
{% for irb in node[inventory_hostname]["overlay"]["irb"] -%}
{% if node[inventory_hostname]["overlay"]["irb"][irb]["vrf"] is defined %}
{% if [node[inventory_hostname]["overlay"]["irb"][irb]["vrf"]] not in vrfs %}
auto {{node[inventory_hostname]["overlay"]["irb"][irb]["vrf"]}}
iface {{node[inventory_hostname]["overlay"]["irb"][irb]["vrf"]}}
    vrf-table auto
```

Layer 2 and Layer 3 (a quick private history)

Once upon a time I had a CAT…


```
telnet 10.0.0.1 or
session 3

> enable
# show run
# conf t

router bgp 65000
router-id 1.1.1.1
```

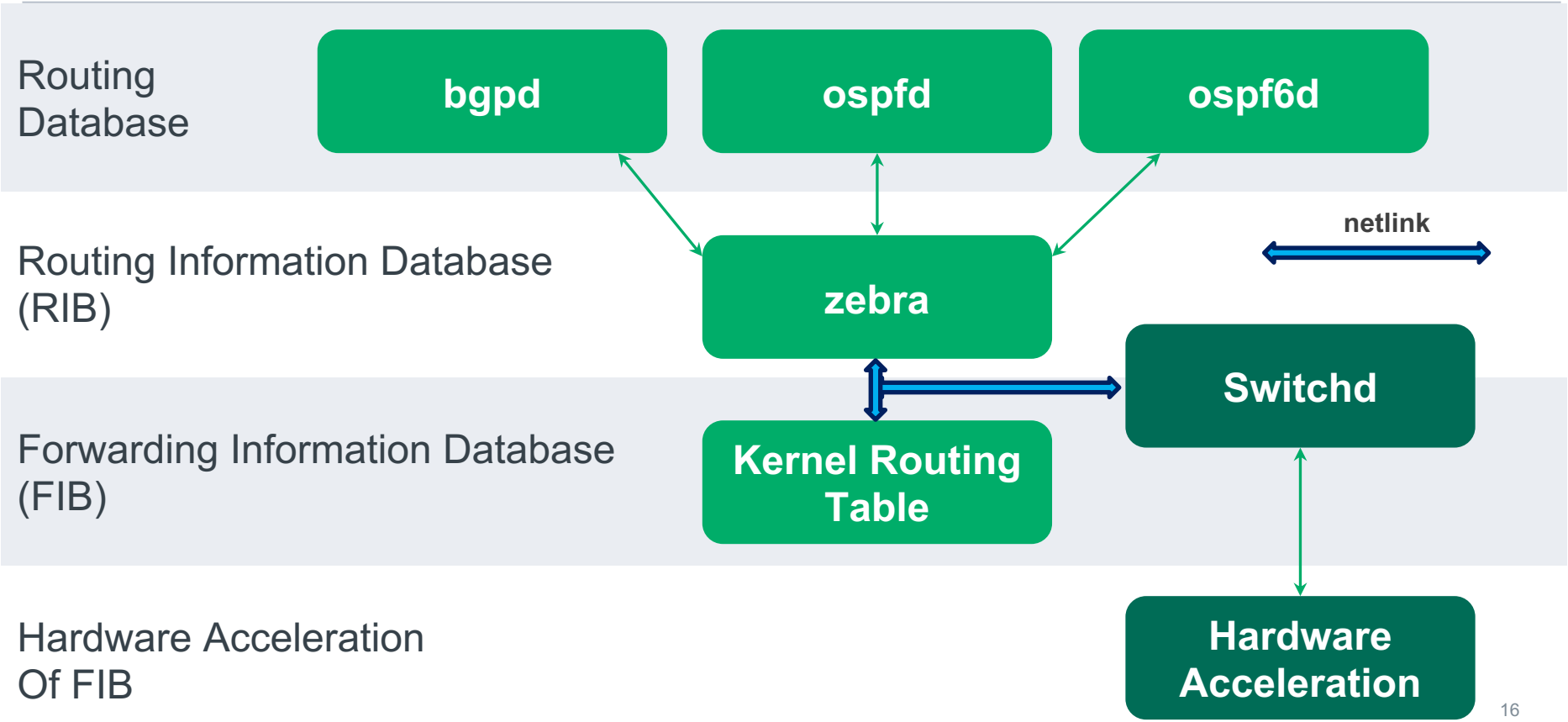
FRRouting Overview

FRROUTING

Open source routing software suite

- Linux Foundation project
- https://frrouting.org
- https://github.com/FRRouting/frr
- Presents an industry-standard CLI
- Robust protocol set

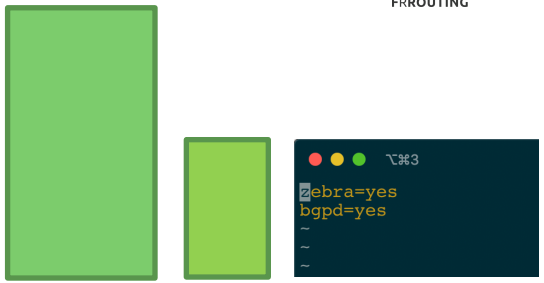
FRRouting Architecture



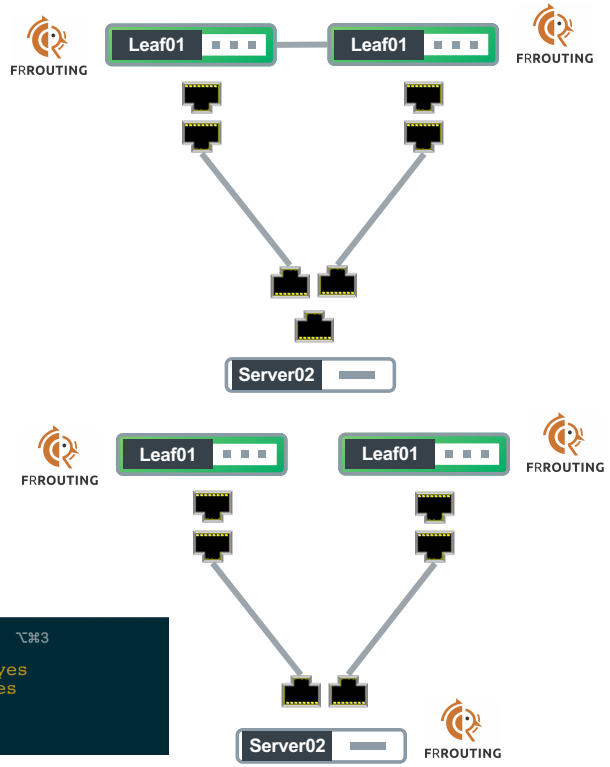
FRR

```yaml
- hosts: switches
  name: Configure /etc/frr/daemon
  become: yes
  tasks:
    - copy:
        src: /home/cumulus/on/on-001/daemons
        dest: /etc/frr/daemons
      notify:
        - frr_restart
```

```yaml
- copy:
    src: /home/cumulus/on/on-001/frr_spine01
    dest: /etc/frr/frr.conf
  notify:
    - frr_reload
```

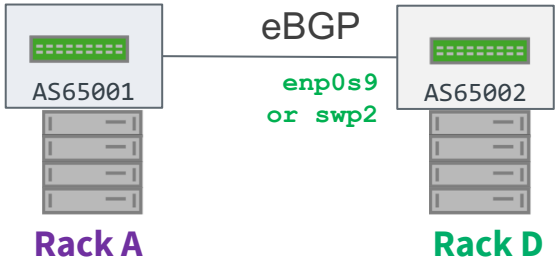


reload vs restart



BGP Configuration Evolution

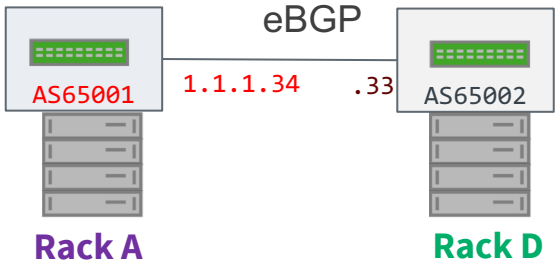
reference




```
router bgp 65002
 bgp log-neighbor-changes
 bgp router-id 10.0.0.17
 !
 neighbor swp2      remote-as external
```

Cumulus Linux (+ upstream FRR)

- Simple configuration
  Ideal for automating
- Enable BGP on an interface
- Removes IP addressing and AS numbers




```
router bgp 65002
 bgp log-neighbor-changes
 bgp router-id 10.0.0.17
 !
 neighbor 1.1.1.34      remote-as 65001
```

&

```
interface eth1
  address 1.1.1.33 255.255.255.252
```

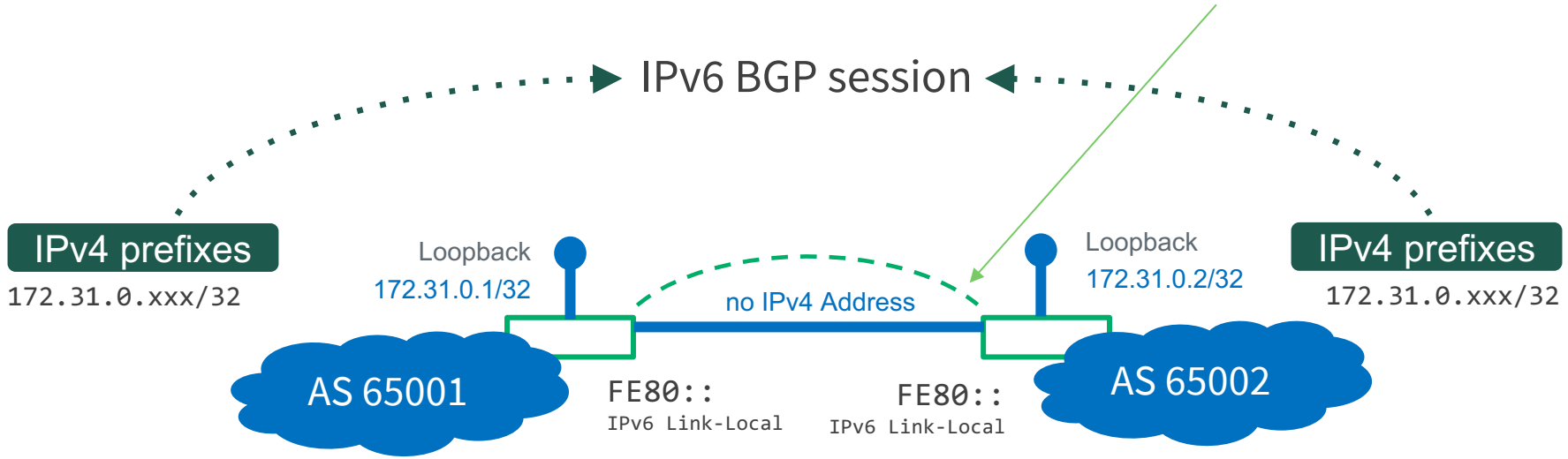
Traditional/Incumbent Vendor

- Complex configuration
  IP addresses
  AS numbers
- Slow timers
- Unique info for each node
- “external” information to this nodes config needed

BGP Unnumbered

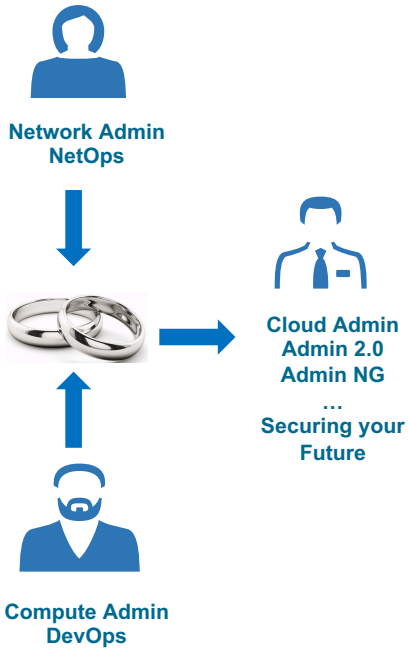
reference

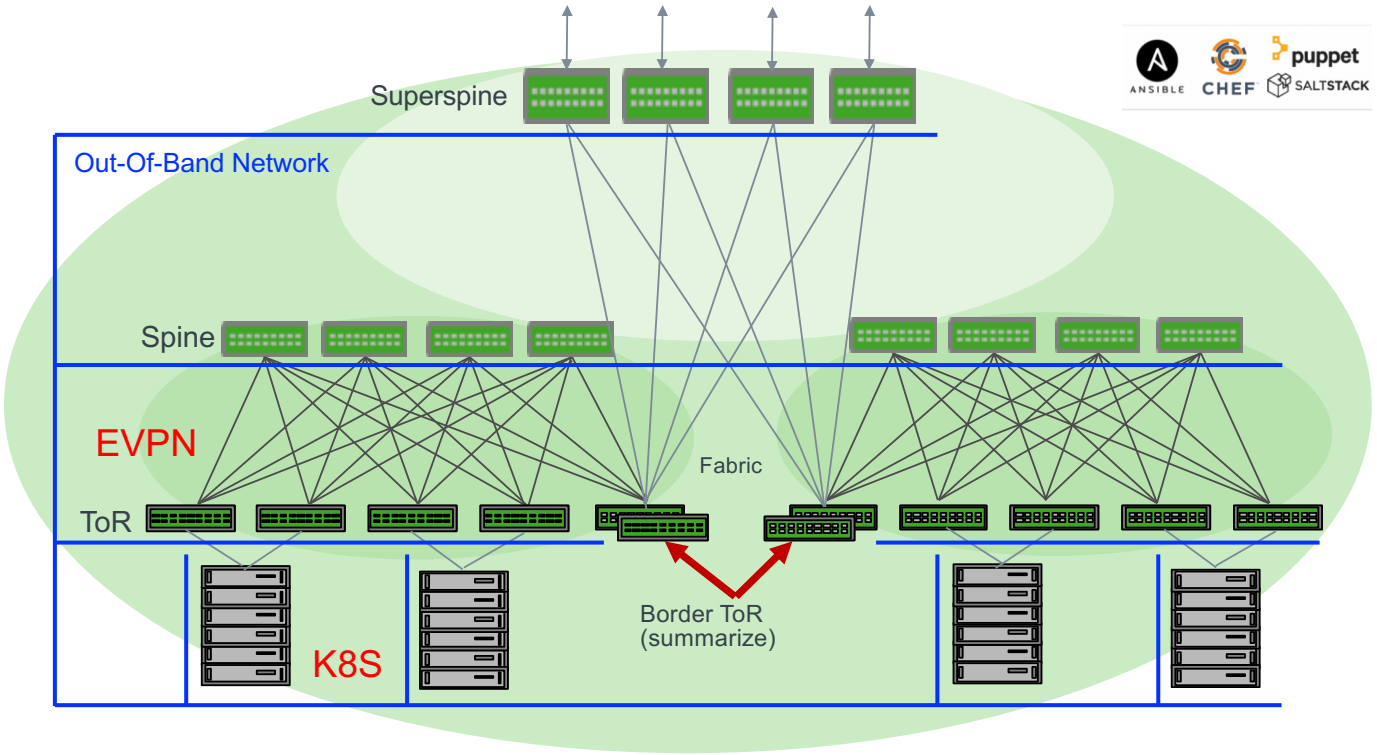
```
neighbor enp0s9 interface remote-as external
```



- IPv6 link local address for BGP sessions (not really unnumbered)
- RFC 5549 advertises IPv4 addresses over IPv6 session
- IPv6 router advertisement to learn neighbor's link local address

Real World Data-Center Fabric





Live and/or ”outside…”

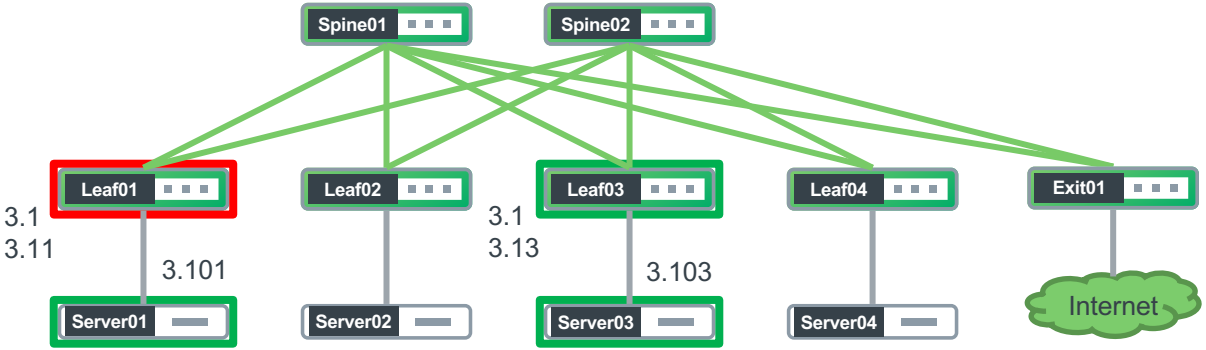
21

Example-01: EVPN decentralized symmetric routing

/on3

ansible-playbook clean-leaf01.yml
ip route
net show route

ansible-playbook evpn-leaf01.yml
ip route
net show route
net show route vrf vrf1



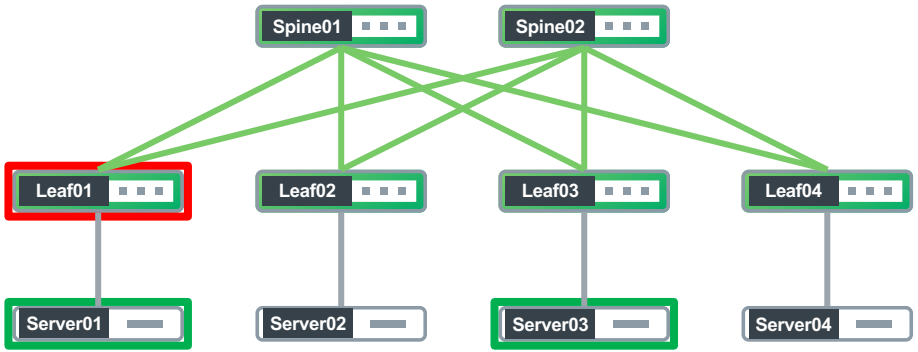
ping 10.1.3.1
ping 10.1.3.11
ping 10.1.3.13
ping 10.1.3.103

22

Example-01: EVPN decentralized symmetric routing

```
---
- hosts: leaf01
  name: clean it up
  become: yes
  gather_facts: no
  tasks:
    - name: clean e/n/i
      copy:
        src: /home/cumulus/on3/leaf01-clean
        dest: /etc/network/interfaces
    - name: activate
      shell: /sbin/ifreload -a
```

```
---
- hosts: leaf01
  name: evpn leaf01
  become: yes
  gather_facts: no
  tasks:
    - name: configure evpn e/n/i
      copy:
        src: /home/cumulus/on3/leaf01-evpn
        dest: /etc/network/interfaces
    - name: activate
      shell: /sbin/ifreload -a
```

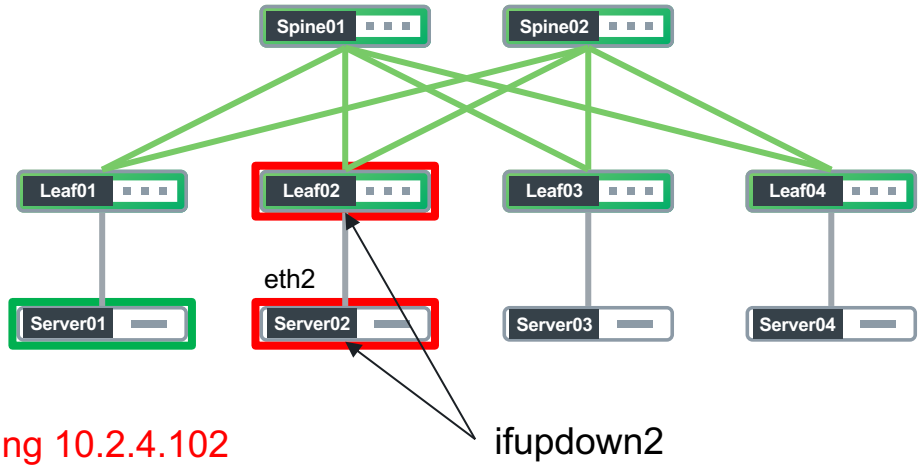


23

Example-02: EVPN decentralized multi-tenant symmetric routing

ansible server02 -a "sudo ifdown eth2“
ansible leaf02 –a „sudo ifdown swp2“

ansible server02 –a ip route
ansible leaf02 –a ip route
ansible leaf02 –a ip route show vrf vrf1



ansible-playbook leaf02-server02.yml
ip route
net show route
net show route vrf vrf1

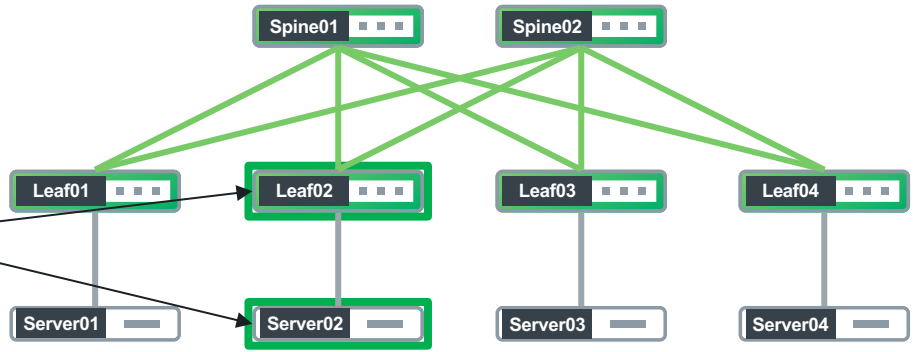
ping 10.2.4.102

ping 10.2.4.102

24

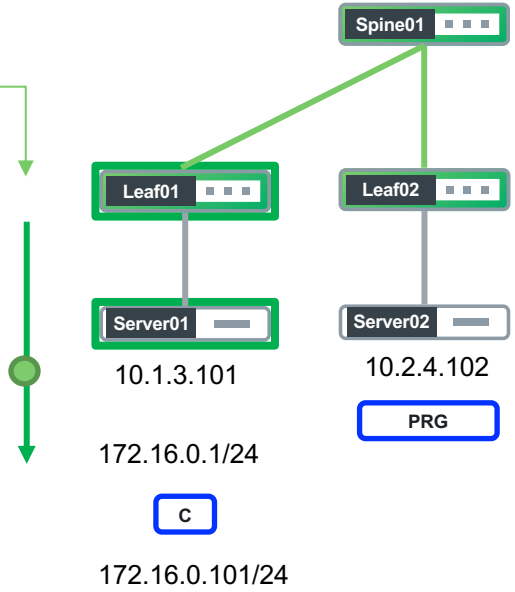
Example-01: EVPN decentralized symmetric routing

```yaml
---
- hosts: server02
  name: set it up
  become: yes
  become_method: sudo
  gather_facts: no
  tasks:
    - name: setup e/n/i
      copy:
        src: /home/cumulus/on3/server02
        dest: /etc/network/interfaces
    - name: activate
      shell: /sbin/ifreload -a

- hosts: leaf02
  name: set ip up
  become: yes
  gather_facts: no
  tasks:
    - name: setup e/n/i
      copy:
        src: /home/cumulus/on3/leaf02-evpn
        dest: /etc/network/interfaces
    - name: activate
      shell: /sbin/ifreload -a

    - name: frr
      copy:
        src: /home/cumulus/on3/leaf02-frr.conf
        dest: /etc/frr/frr.conf

    - name: frr_restart
      service:
        name=frr
        state=restarted
        enabled=yes
```



Example-03: NS (Server-01) + Leaf01 (Static + Redistribution)

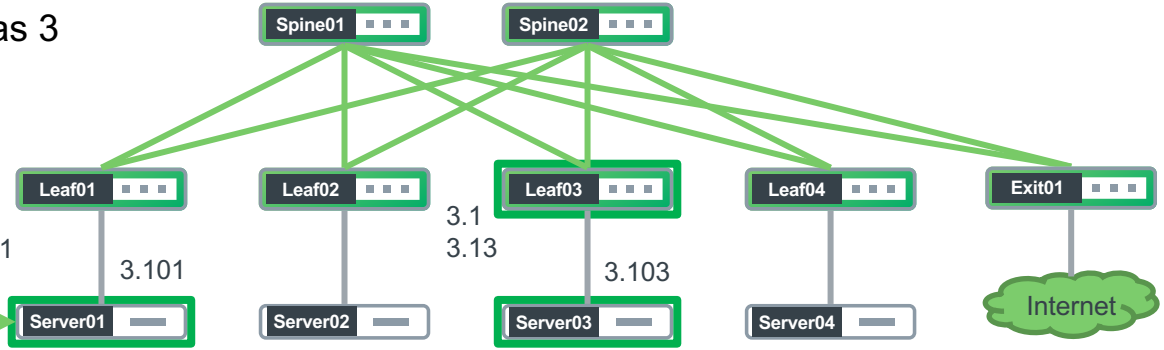
```
net show route vrf vrf1 | grep S
S>* 172.16.0.0/24 [1/0] via 10.1.3.101, vlan13, 1d02h13m
```

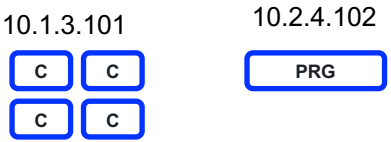


26

Example-04: K8S (Server-01) + Leaf01 (Static + Redistribution)

kubectl run nginx --image nginx --replicas 3





ip netns

sudo ip netns exec cni-78e7f747-1bfa-8431-3b8d-3a06bf0e178e ip route
sudo ip netns exec cni-78e7f747-1bfa-8431-3b8d-3a06bf0e178e ping 10.1.3.1/11/13/103
sudo ip netns exec cni-78e7f747-1bfa-8431-3b8d-3a06bf0e178e ping 10.2.4.102
sudo ip netns exec cni-78e7f747-1bfa-8431-3b8d-3a06bf0e178e traceroute 10.2.4.102

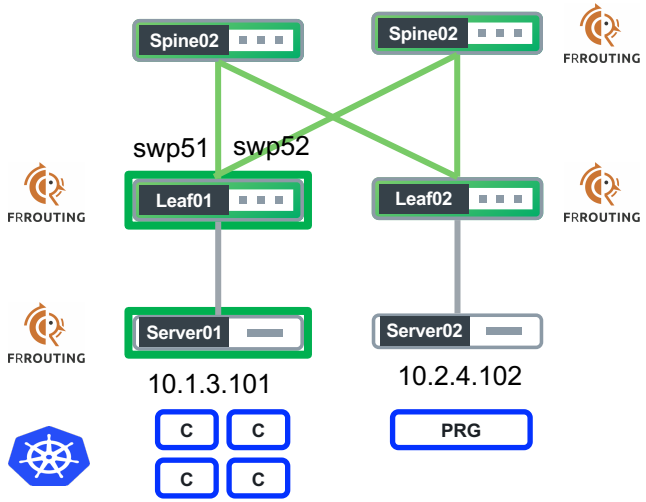
27

Example-05: FRR (Server-01)

```
router bgp 65011
 bgp router-id 10.0.0.11
 bgp bestpath as-path multipath-relax
 neighbor peerlink.4094 interface remote-as internal
 neighbor swp51 interface remote-as external
 neighbor swp52 interface remote-as external
 !
 address-family ipv4 unicast
  redistribute connected route-map LOOPBACK_ROUTES
 exit-address-family
 !
 address-family l2vpn evpn
  neighbor peerlink.4094 activate
  neighbor swp51 activate
  neighbor swp52 activate
  advertise-all-vni
 exit-address-family
!
router bgp 65011 vrf vrf1
 neighbor 10.1.3.101 remote-as 65666
 !
 address-family ipv4 unicast
  redistribute connected
 exit-address-family
!
route-map LOOPBACK_ROUTES permit 10
 match interface lo
```

```
auto swp51
iface swp51
  mtu 9216
  alias to Spine01

auto swp52
iface swp52
  mtu 9216
  alias to Spine02
```

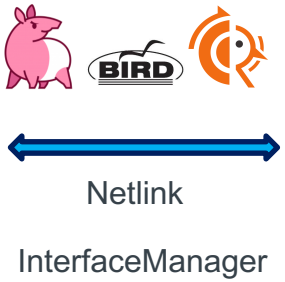
```
Hello, this is FRRouting (version 6.0.2).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

server01# sh run
Building configuration...

Current configuration:
!
frr version 6.0.2
frr defaults traditional
hostname server01
no ipv6 forwarding
no service integrated-vtysh-config
!
router bgp 65666
 bgp router-id 10.1.3.101
 neighbor 10.1.3.11 remote-as 65011
 !
 address-family ipv4 unicast
  redistribute connected
 exit-address-family
!
line vty
!
bfd
!
end
server01#
```

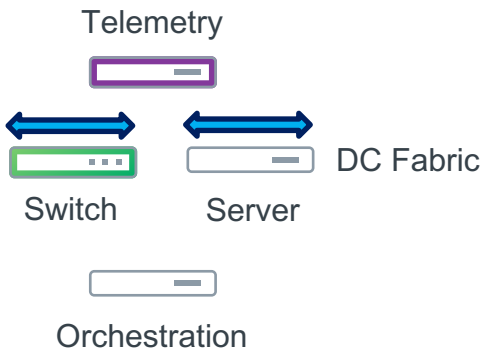


BGP unnumbered
Works well to the server,
For illustrative/education purposes
the legacy option with remote IPs and remote ASN
(== the more complicated variant) is shown

Open Networking

- Late 2012, brainstormed initial features
- Early 2013, evangelized with HW partners
- May 2013, first public demo at OCP MIT Workshop
- Summer 2013, first products available from multiple vendors
- Summer 2013, project incubated by OCP
- June 2014, project fully adopted by OCP





29

Open Networking

One tool-set to rule them all

One skill-set to rule them all

Simple:
Server == Network

30

CUMULUS

Thank you!

Visit us at cumulusnetworks.com or follow us @cumulusnetworks