SONiC Architecture and Deployment Deep Dive

Jiri Chaloupka
Principal Technical Marketing Engineer

BRKMSI-2004

Cisco Webex App

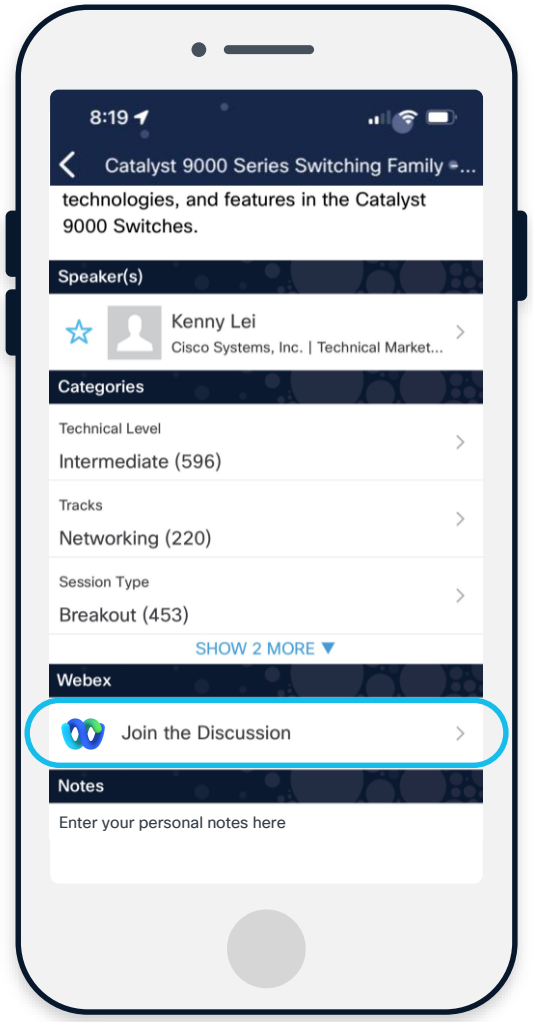
Questions?

Use Cisco Webex App to chat
with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until June 13, 2025.



Agenda

01 SONiC Highl Level Architecure &
SONiC on Cisco 8000

02 SONiC Internals / Workflow
- Life of Port
- Life of New IP prefix

03 Initial Loading

04 Essential CLI

05 Configuration Changes & Upgrade

06 SONiC Usecases
- IP Fabric
- L2VPN All-Active EVPN with VXLAN Data Plane
- L3VPN All-Active EVPN with VXLAN Data Plane
- AI Usecases

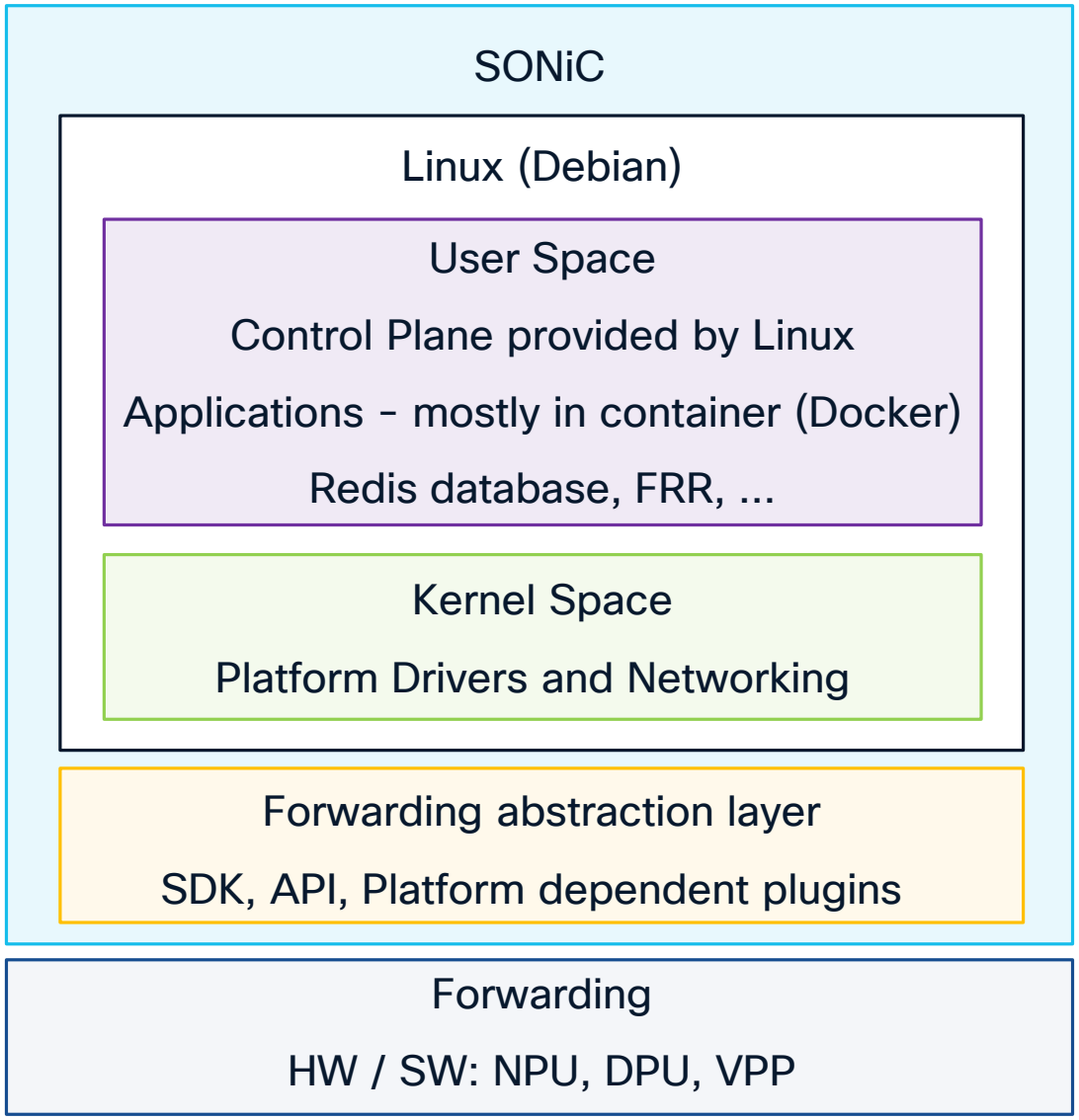
07 Summary

The SONiC Network Operating System

- Open-source network operating system based on Debian Linux.
- Initially created by Microsoft and Open-sourced in 2016.
- It became part of the Linux Foundation in 2022 which focuses on the software components of SONiC
- Open Compute Platform (OCP) helps in aligning hardware and specifications like the Switch Abstraction Interface (SAI).
- https://github.com/sonic-net/SONiC/wiki
- https://sonicfoundation.dev/

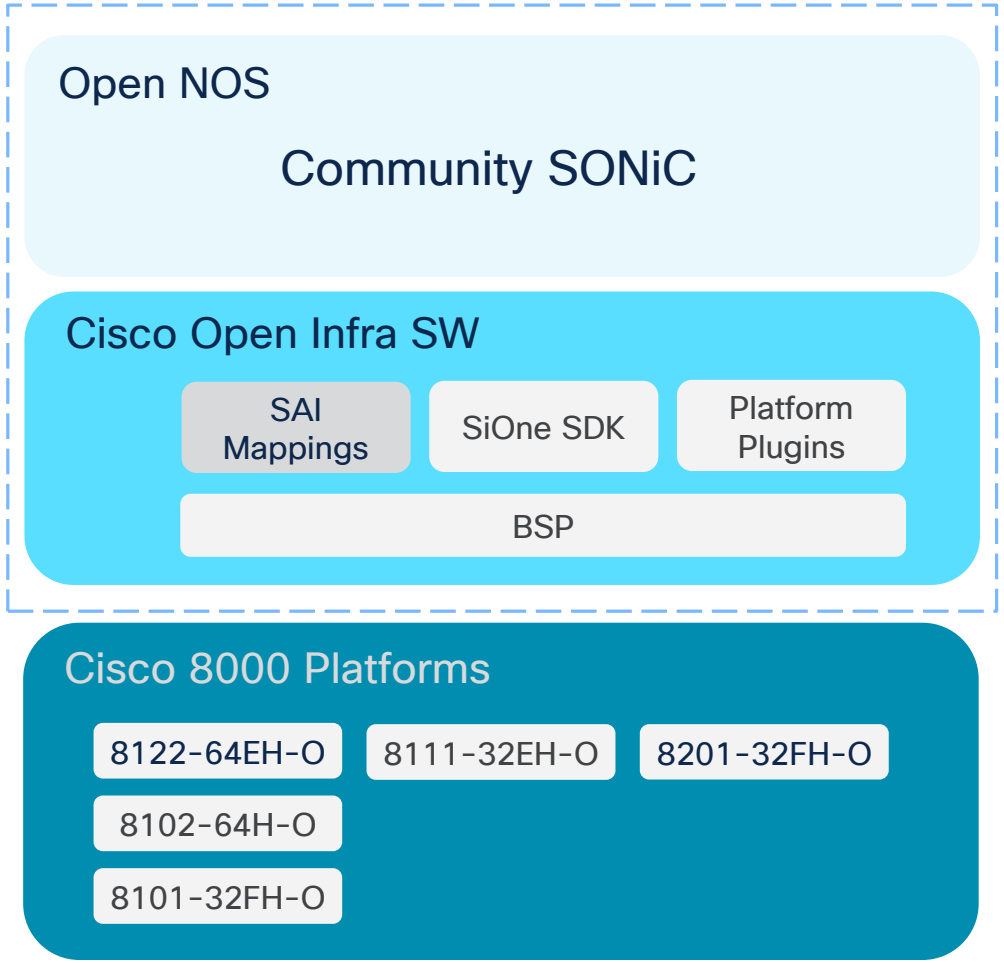
SONiC High Level Architecture & SONiC on Cisco 8000

SONiC High Level Architecture View



SONiC on Cisco 8000 #1

Full Stack Support – Cisco Validated SONiC

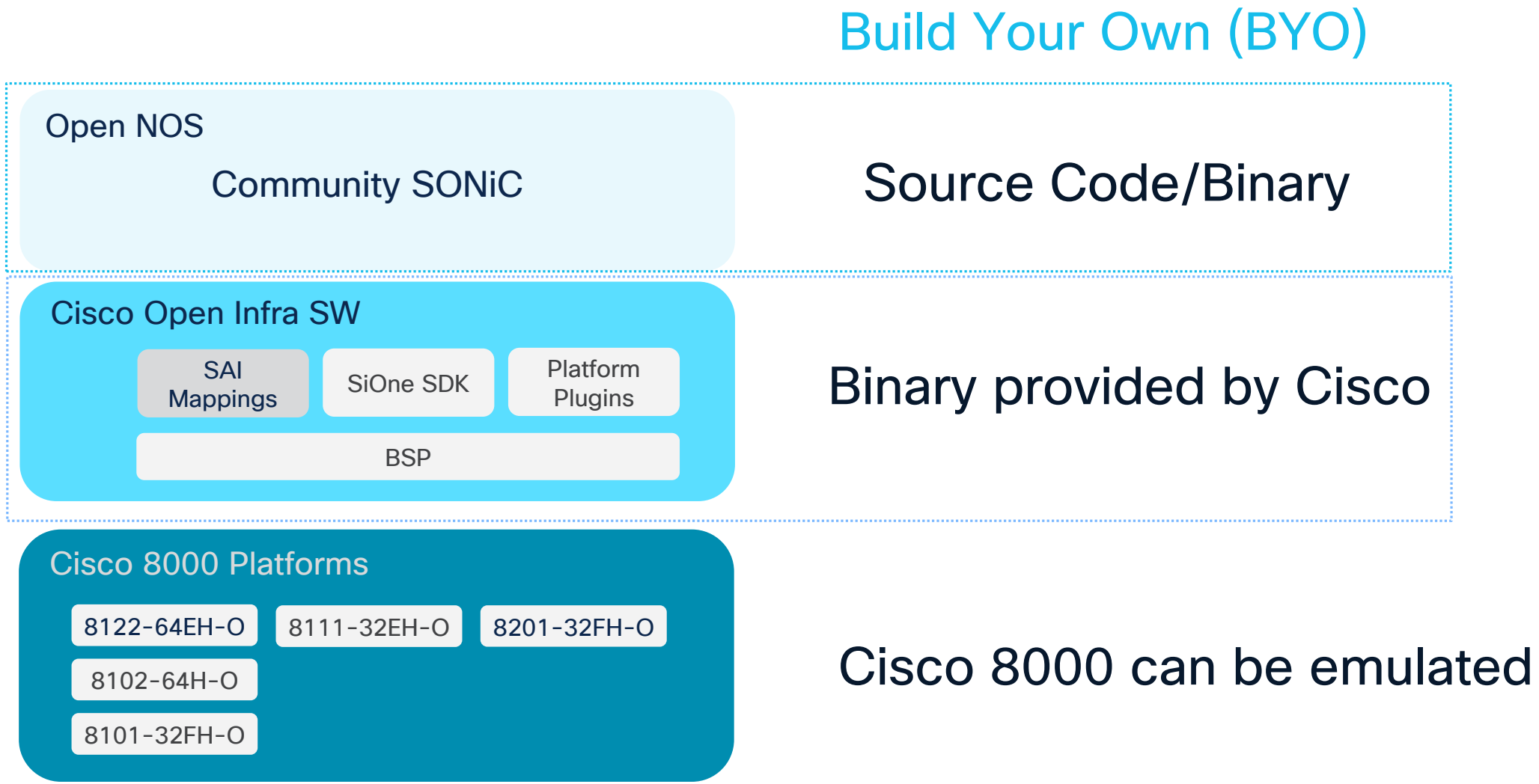


Bootable Image provided by Cisco

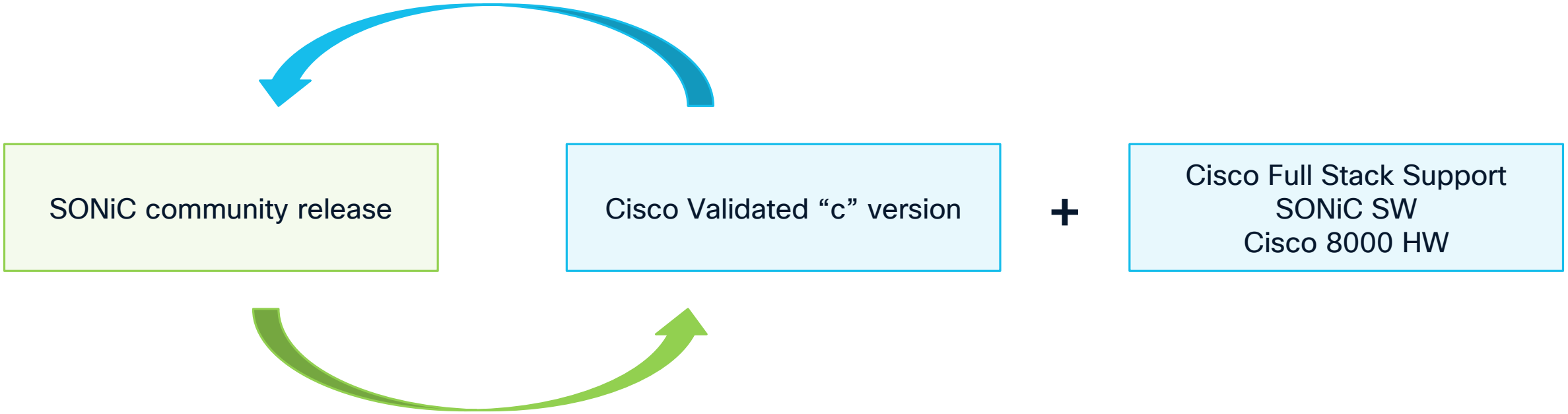
Cisco 8000 can be emulated

SONiC on Cisco 8000 #2

Build Your Own (BYO)



Cisco SONiC Release View



Cisco & SONiC – Strong Participation

Chassis *Smartswitch*

Routing *Platform*

SONiC-VPP *Test*

Contributions to multiple working groups

THE LINUX FOUNDATION

SONiC

CISCO

Linux Foundation Board Technical Steering Committee

Principal Engineers

Development Engineers

Architects

Test Engineers

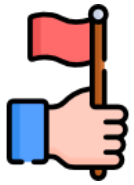
Dedicated SONiC Team

Source code sharing

Fully Integrated image

Option to build your own

Enable BYO innovation



Architecture Innovations

System Integrations

Industry thought leadership

Hyperscalers

Enterprises

DPU/GPU Vendors

Strategic customer partnerships

cisco

Cisco Solution for DC/AI/WAN

Domain Specific OS/Controller

Cisco 8000

Tier 2 web AI aaS | Hyperscalers | Service Providers WAN / SDN

Nexus 9000 with Nexus Dashboard

Enterprise Public sector Commercial | Service providers | Tier 2 web AI aaS

Nexus Hyperfabric

Enterprise Public sector Commercial

Customizable solution

Cisco IOS XR
Cisco Validated SONiC
BYO SONiC

Network solution

Private cloud / DC

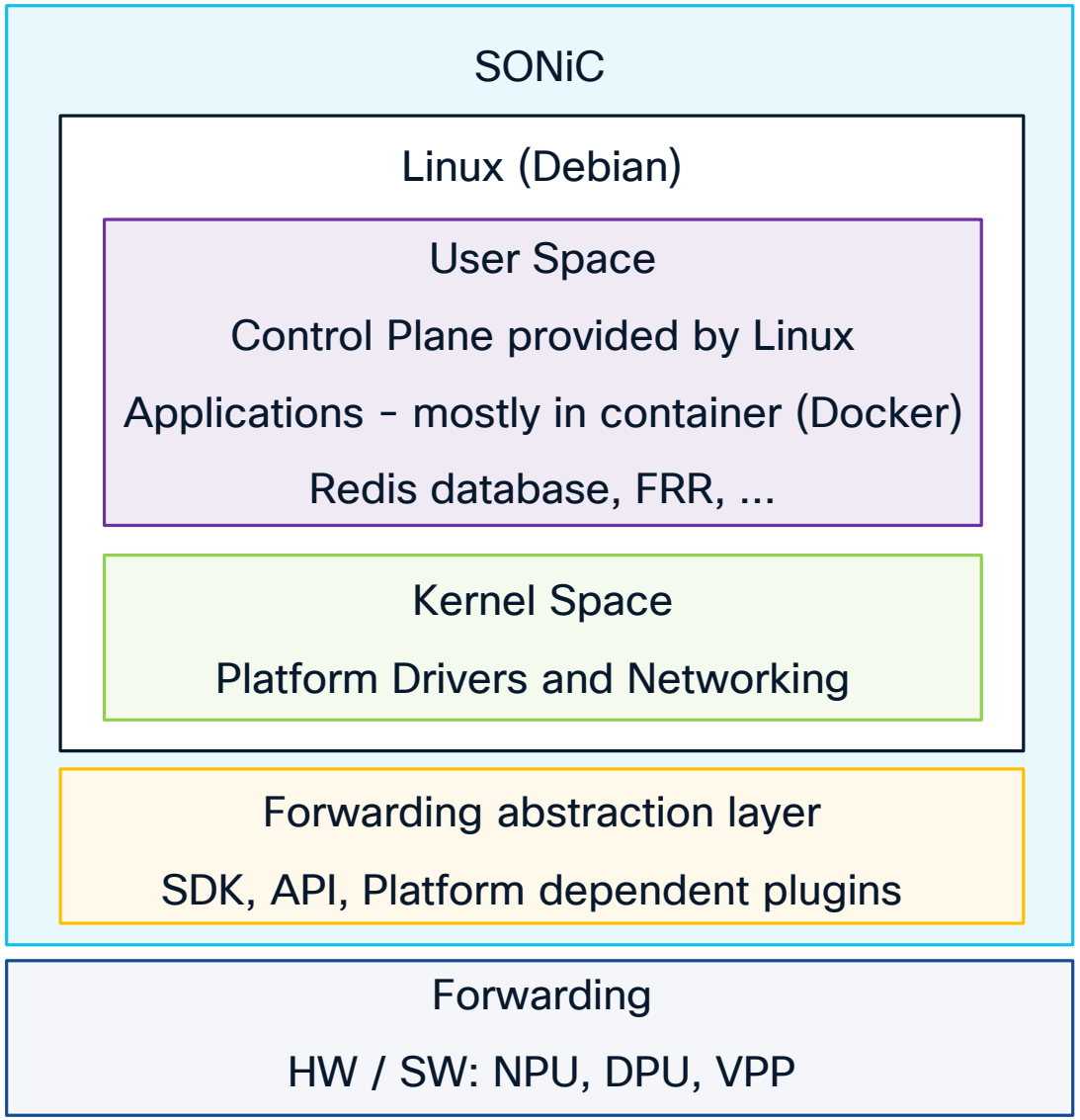
SaaS solution

Public cloud managed, full stack

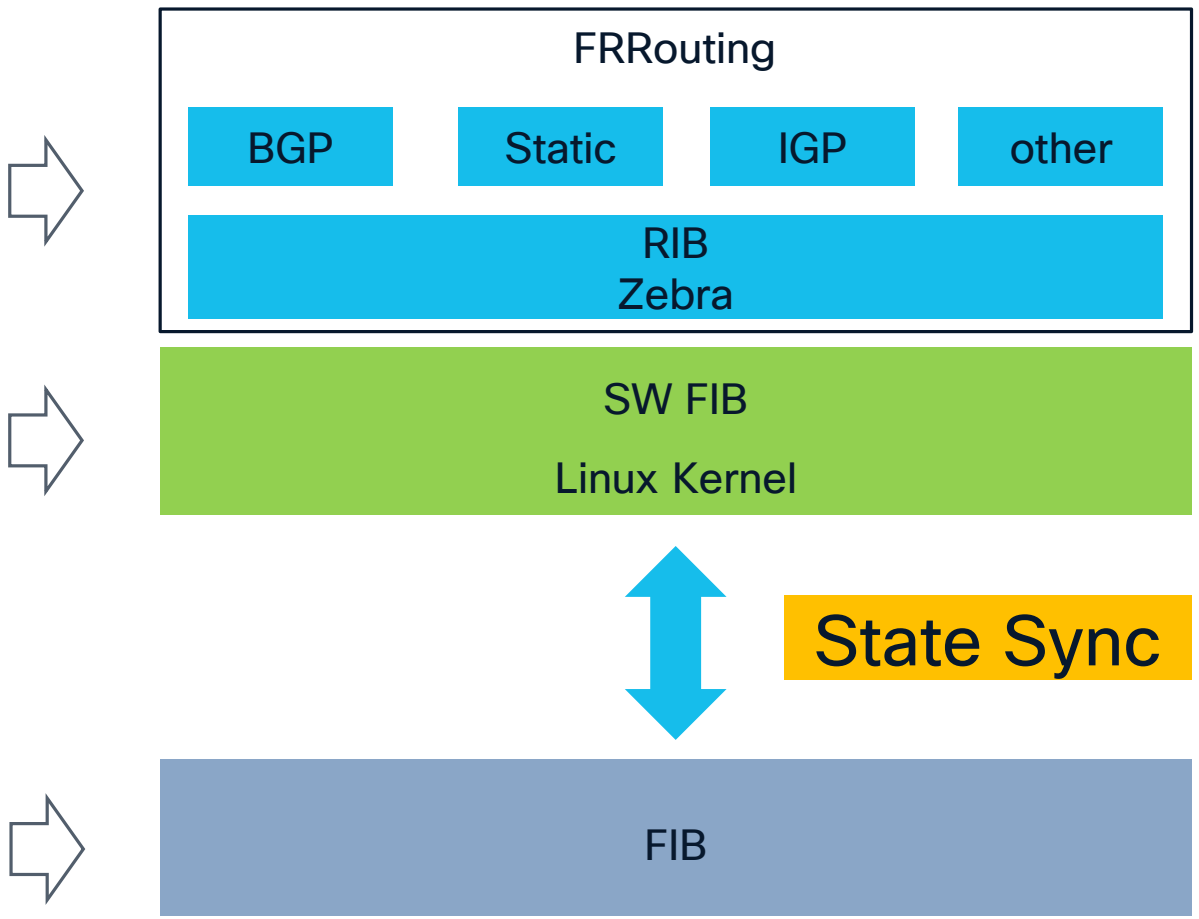
Common Components based on Cisco Silicon ONE

Silicon | SDK | Hardware | Optics

SONiC High Level Architecture View

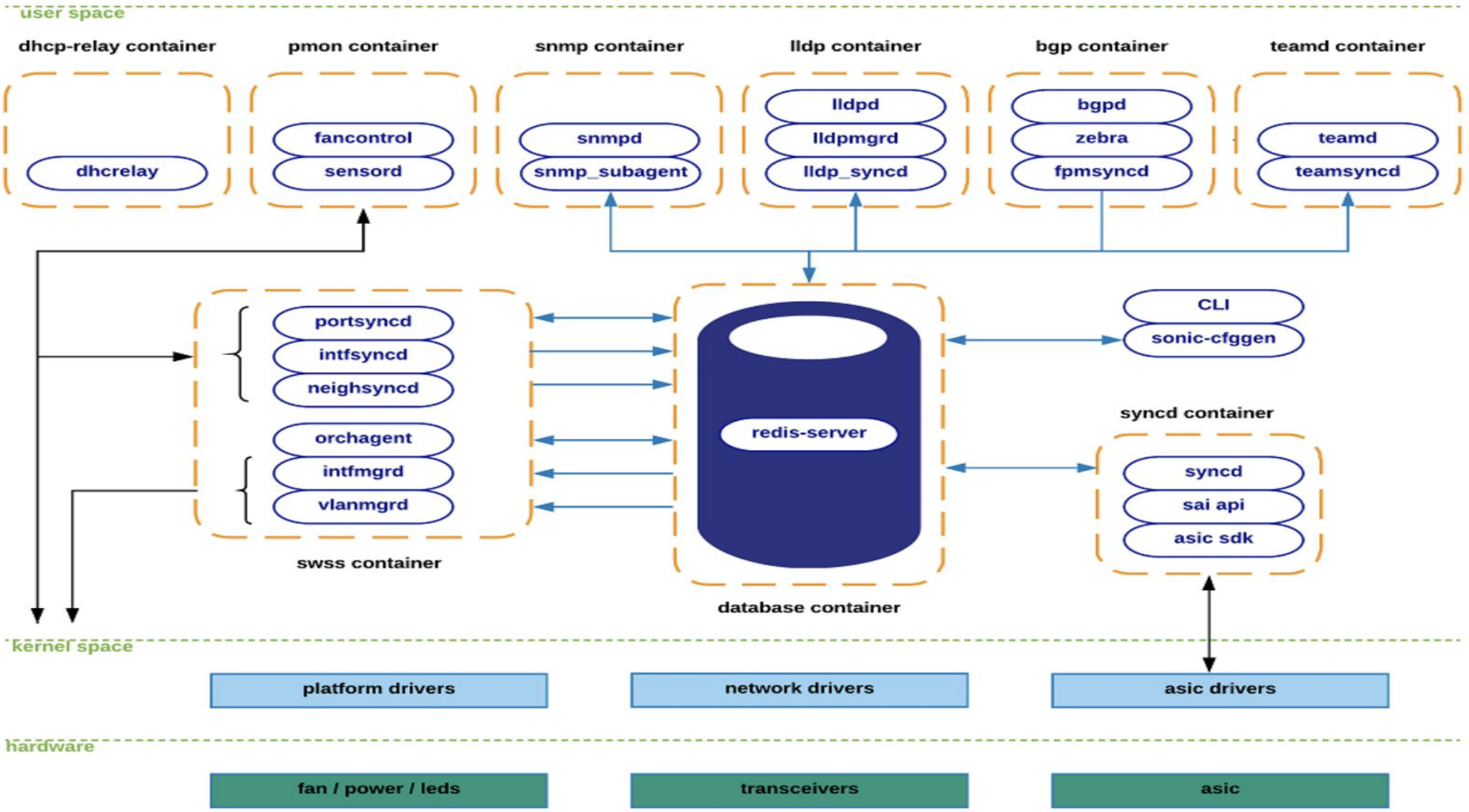


Control Plane / Data Plane View



SONiC Internals / Workflow

SONiC Architecture



Source: https://github.com/sonic-net/SONiC/wiki/Architecture

SONiC Architecture – Database Container

- Redis-database engine is a key-value database
- Databases held within this engine are accessible to SONiC applications through a UNIX socket

APPL_DB: Stores the state/information generated by all application containers

- routes, next-hops, neighbors, etc.
- south-bound interface for all applications

CONFIG_DB: Stores the configuration created by SONiC applications

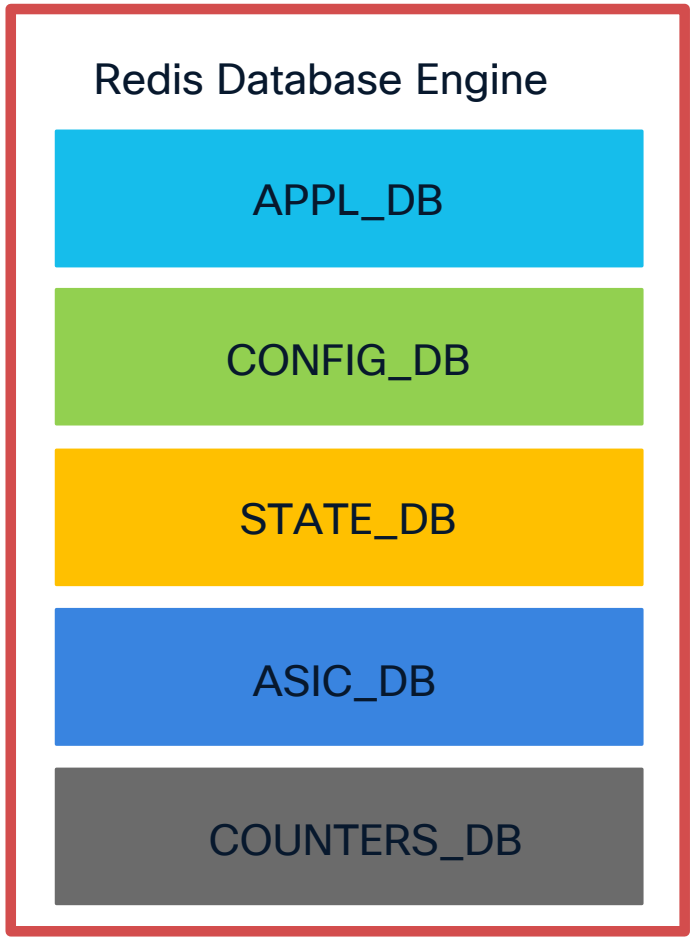
- port configurations, interfaces, vlans, etc.

STATE_DB: Stores states to resolve cross application or module dependencies

- Example: vlanmgrd needs visibility about port presence

ASIC_DB: Stores state to drive asic's configuration and operation

COUNTERS_DB: Stores counters/statistics associated to each port in the system



Source: https://github.com/sonic-net/SONiC/wiki/Architecture

SONiC Architecture – Switch State Service (SwSS) container

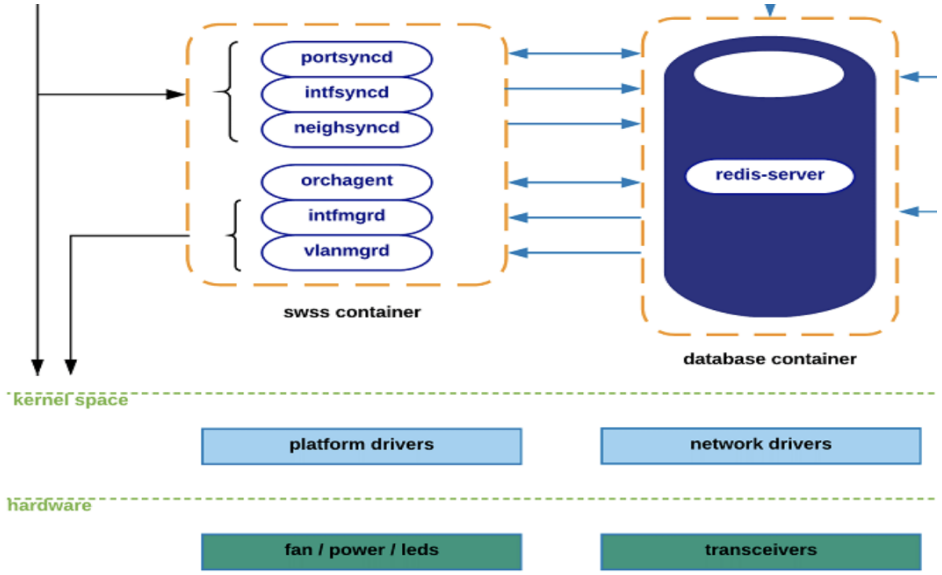
- Cross module communication and arbitration

Processes listening netlink (Linux Kernel) events and producing state / information into APPL_DB / STATE_DB

Portsyncd: port-related netlink events. During boot-up, portsyncd obtains physical-port information
pushs all the collected information into APPL_DB (port-speeds, mtu) and state into STATE_DB

Intfsyncd: interface-related netlink events
pushs all the collected information into APPL_DB (IP address)

Neighsyncd: neighbor-related netlink events triggered by newly discovered neighbors as a result of ARP processing.
pushs all the collected information into APPL_DB (Adjacency, MAC)



Source: https://github.com/sonic-net/SONiC/wiki/Architecture

SONiC Architecture – Switch State Service (SwSS) container

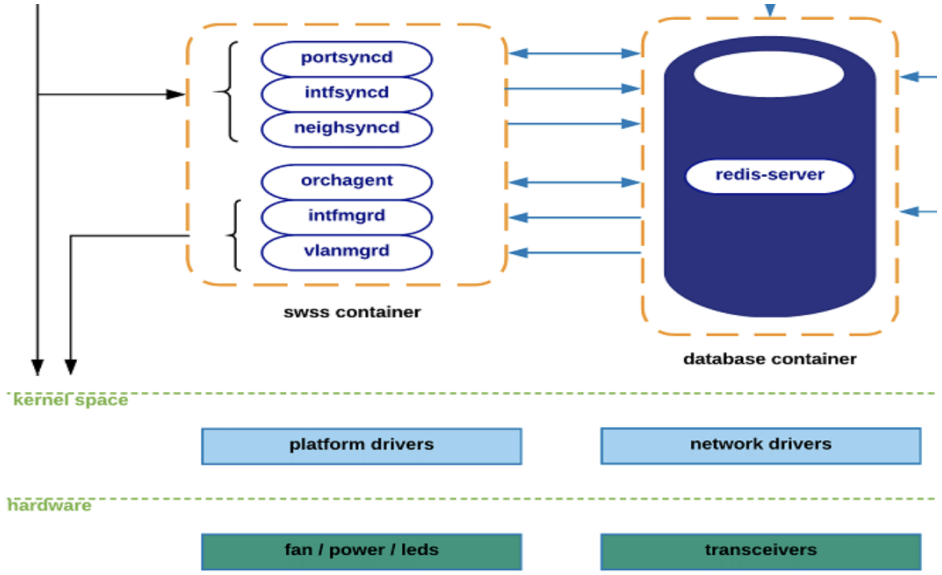
- Cross module communication and arbitration

Processes responsible for distributing state/information

Orchagent: The most critical component in the SwSS subsystem.
Process data from APPL_DB and push new data accrodingly into ASIC_DB

IntfMgrd: Process data from APPL_DB, CONFIG_DB, STATE_DB and configure interfaces in the Linux Kernel

VlanMgrd: Process data from APPL_DB, CONFIG_DB, STATE_DB and configure vlan-interfaces in the Linux Kernel



Source: https://github.com/sonic-net/SONiC/wiki/Architecture

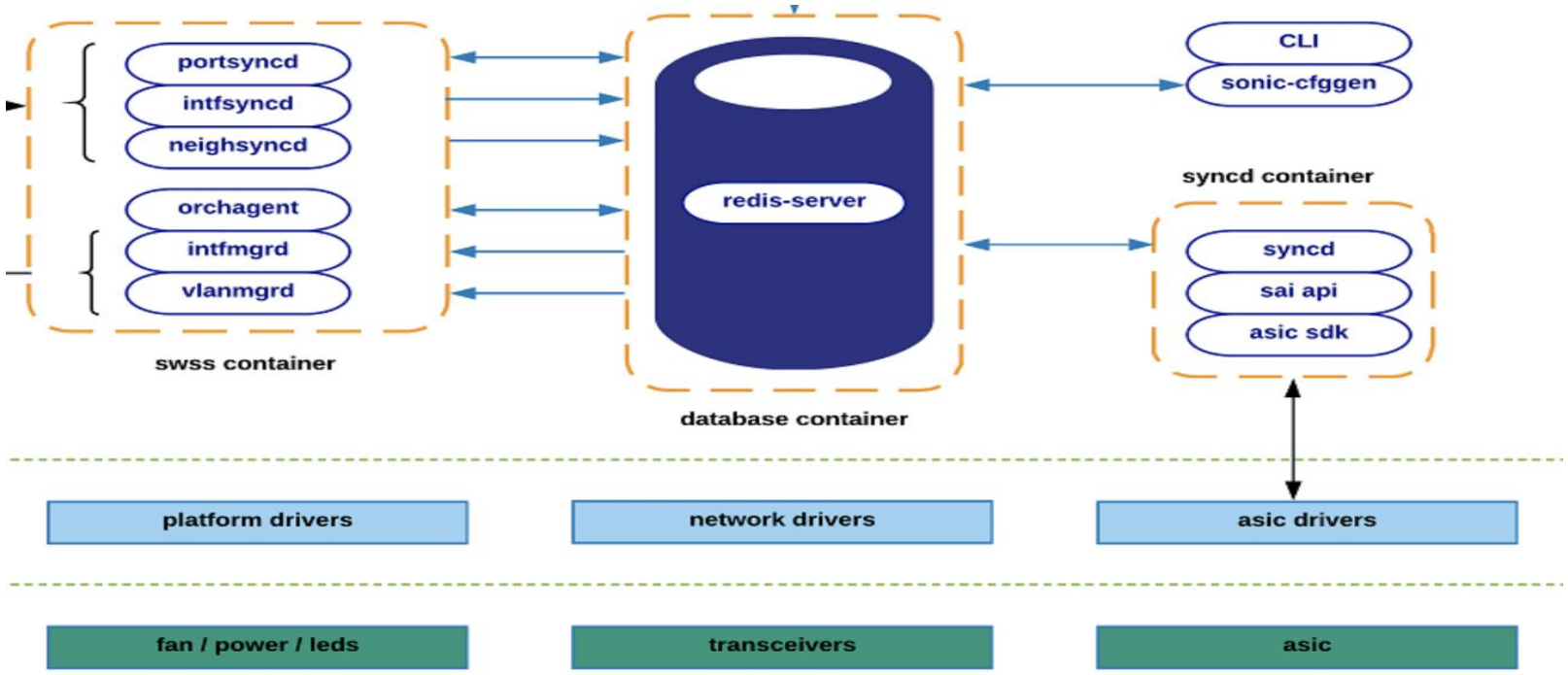
SONiC Architecture – Syncd container

- Synchronization between Switch State Service (SwSS) and actual forwarder (NPU, DPU, VPP, etc)

Syncd: Process in charge of synchronization
Links with the ASIC SDK library during compilation
Subscribed to ASIC_DB to receive state from SWSS

SAI API: The Switch Abstraction Interface (SAI) defines the API to provide a vendor-independent way of controlling forwarder

ASIC SDK: Forwarder vendor specific dynamic-linked-library linked with syncd

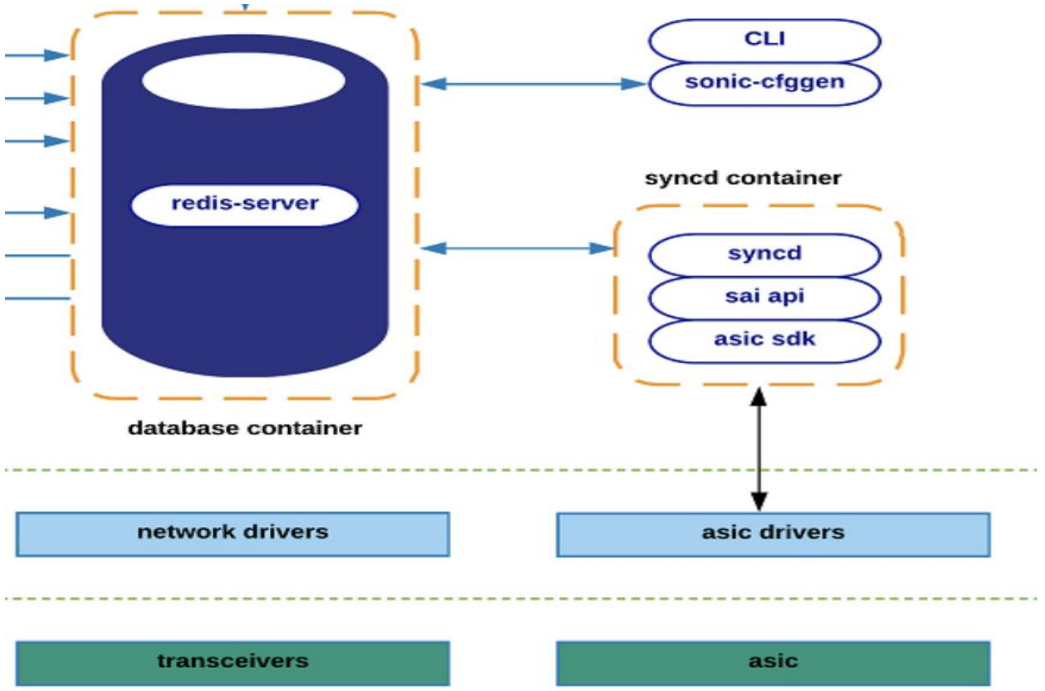


Source: https://github.com/sonic-net/SONiC/wiki/Architecture

SONiC Architecture – CLI / sonic-cfggen modul

- Provides CLI and configuration

CLI component relies on Python's Click library
Sonic-cfggen tool helps with configuration changes



Source: https://github.com/sonic-net/SONiC/wiki/Architecture

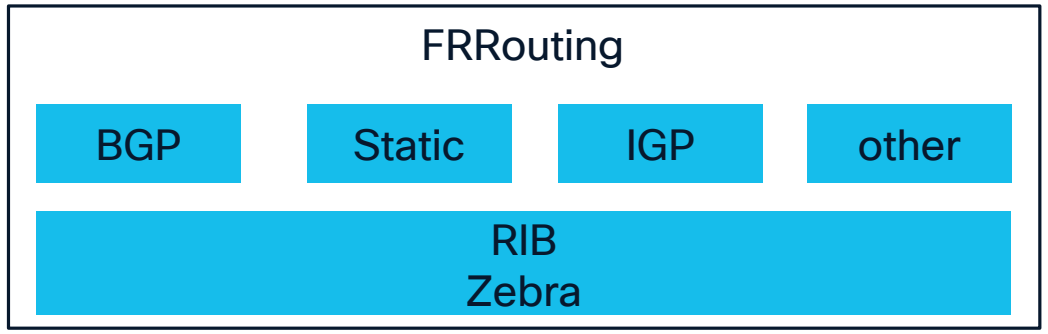
SONiC Architecture – BGP container

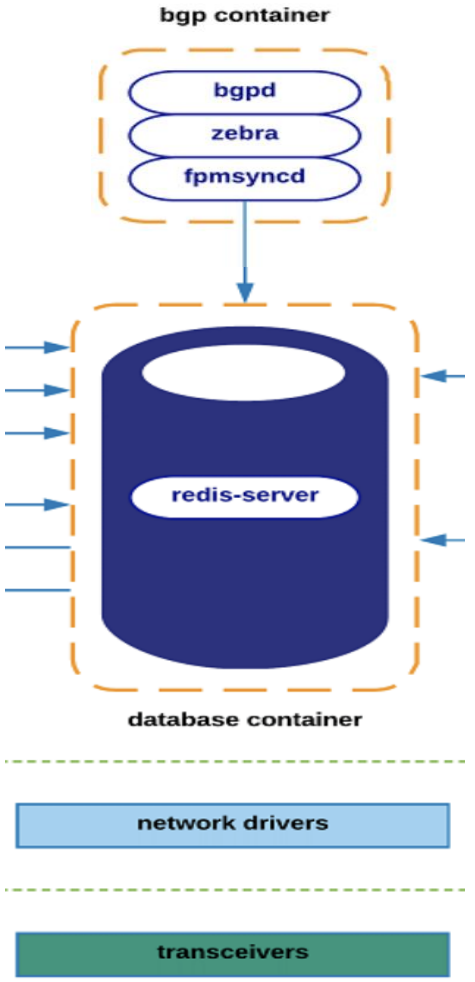
- Routing container instead (FRR)
- BGP, Static, ISIS, OSPF, RIP or even EIGRP ;)
- SONiC usecases relies mainly on BGP and Static

bgpd: BGP implementation
Cisco IOS CLI style
Regular Linux tcp sockets is used to establish BGP session
Push routing/forwarding state to zebra

zebra: RIB
Static Routing, route redistribution component
Updates SW FIB (Linux Kernel) through netlink

fpmsyncd: updates FIB state generated by zebra in APPL_DB (remote SwSS component)





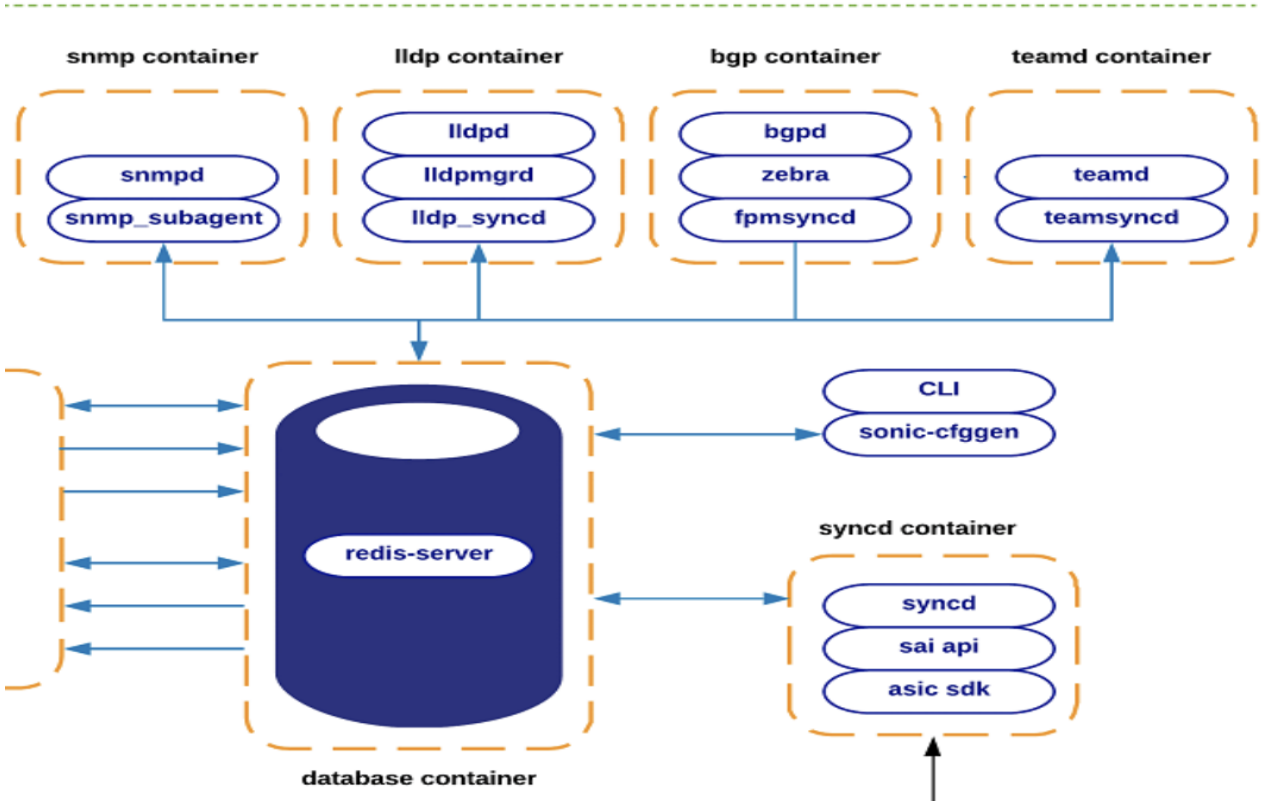
Source: https://github.com/sonic-net/SONiC/wiki/Architecture

SONiC Architecture - Snmp container

- Hosts snmp features

Snmpd: Actual snmp server responsible to process polling from external device

Snmp-agent: Provides data from Redis-engine databases to **Snmpd**



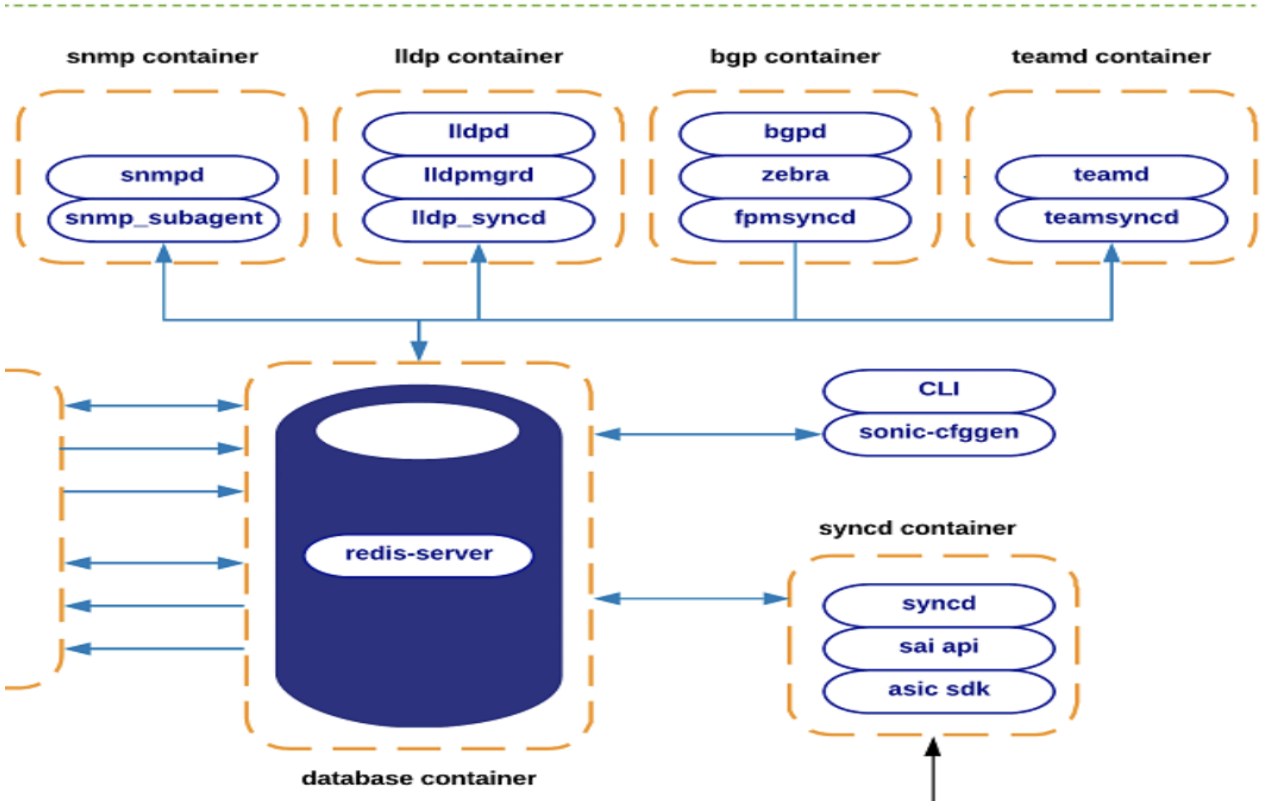
Source: https://github.com/sonic-net/SONiC/wiki/Architecture

SONiC Architecture - Lldp container

Lldp: Manage establishing lldp connections with external peers to advertise/receive system capabilities.

Lldp_syncd: Uploading lldp's discovered state to redis-engine

Lldpmgr: manage subscription to STATE_DB to update lldp in case of configuration/state changes

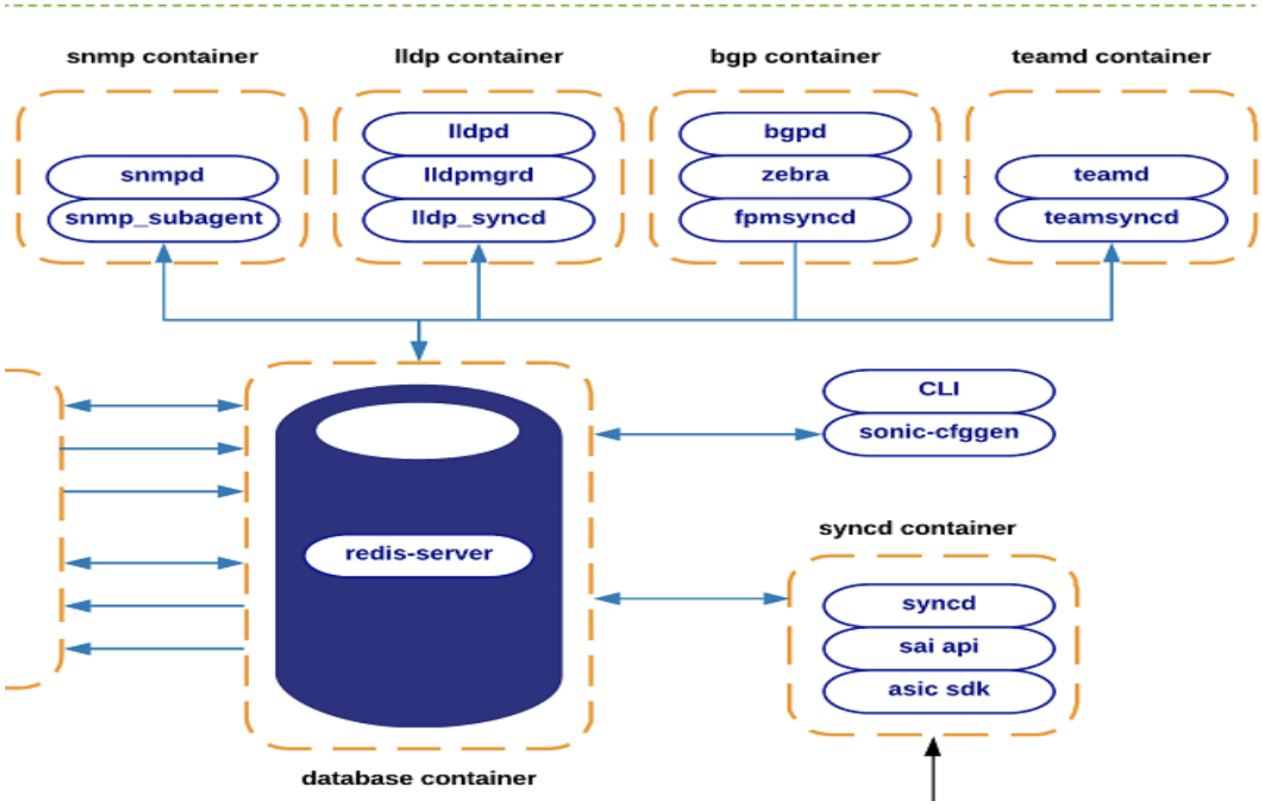


Source: https://github.com/sonic-net/SONiC/wiki/Architecture

SONiC Architecture – Teamd container

- Link Aggregation functionality (LAG) - Port-Channel

teamd: is a linux-based open-source implementation of LAG protocol
teamsyncd: process data from/to APPL_DB and STATE_DB



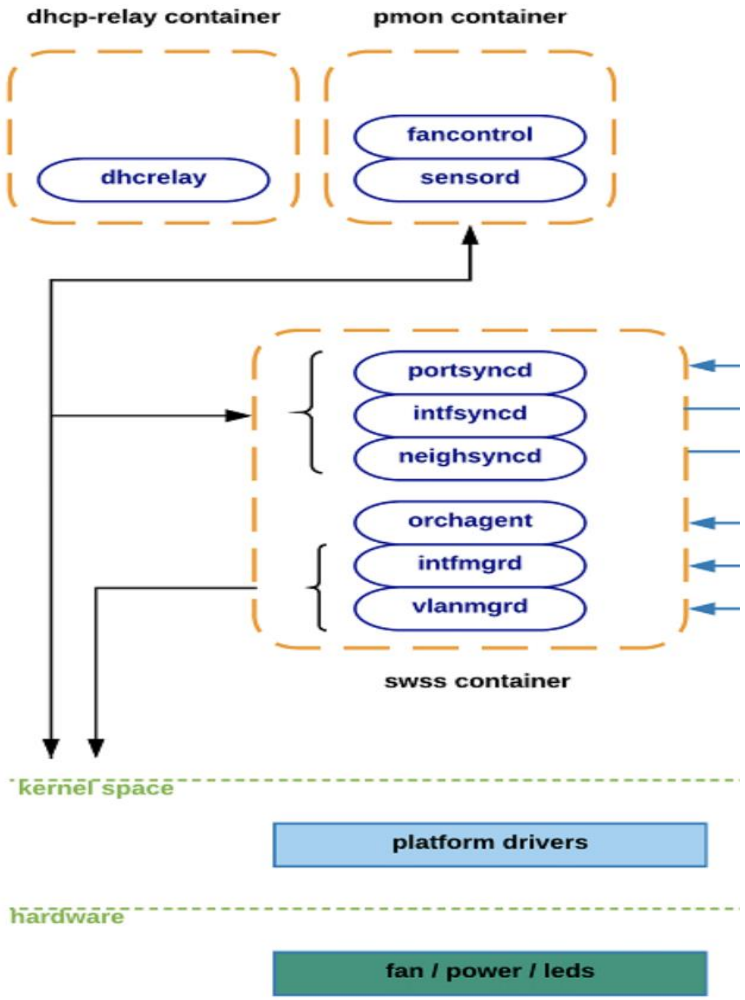
Source: https://github.com/sonic-net/SONiC/wiki/Architecture

SONiC Architecture – Pmon and Dhcp-relay container

Pmon container:

sensord: periodically log sensor from hardware
fancontrol: process fan-related state from platform drivers

Dhcp-relay container:



Source: https://github.com/sonic-net/SONiC/wiki/Architecture

Life of Port

SONiC Port Initialization

(0) portsyncd:
subscriber to CONFIG_DB
produce update into APPL_DB and STATE_DB + netlink

(1) portsyncd:
process port_config.ini associated to the hardware-profile (lines, interface name, speed, etc)
produce update into APPL_DB

(2) orchagent:
notified about APPL_DB changes
produce update into ASIC_DB

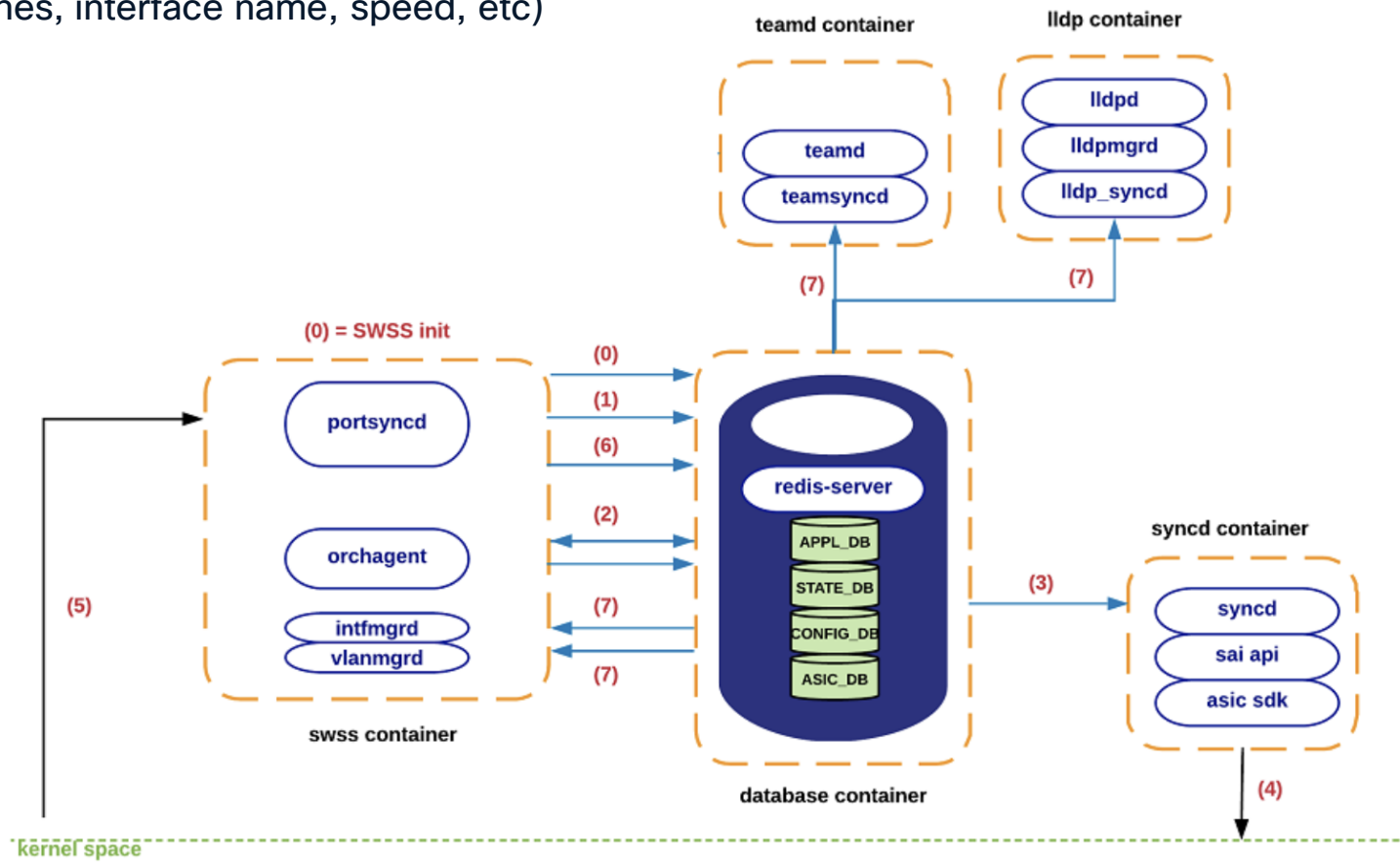
(3) syncd:
notified about ASIC_DB changes

(4) syncd:
update physical ports via SAI APIs + ASIC SDK

(5) portsyncd:
process update generated by Linux Kernel via netlink based on **step(4)** and declare initialization process completed

(6) portsyncd:
update STATE_DB (initialization process completed)

(7) Subscribers to STATE_DB are notified about changes



Source: https://github.com/sonic-net/SONiC/wiki/Architecture

Life of new IP prefix

SONiC BGP Route Update

(0) zebra (RIB) connects to fpmsyncd through a regular TCP socket.
Linux Kernel, APPL_DB and ASIC_DB are expected to be fully consistent/equivalent.

(1) BGP update arrives
Linux Kernel delivers update to bgpd process

(2) bgpd proces bgp-update and notifies zebra

(3) zebra validate reachabilit of new prefix (next-hop)
zebra generates a route-netlink message to inject this new st

(4) zebra send netlink-route message to fpmsyncd.

(5) fpmsyncd processes the netlink message and pushes this

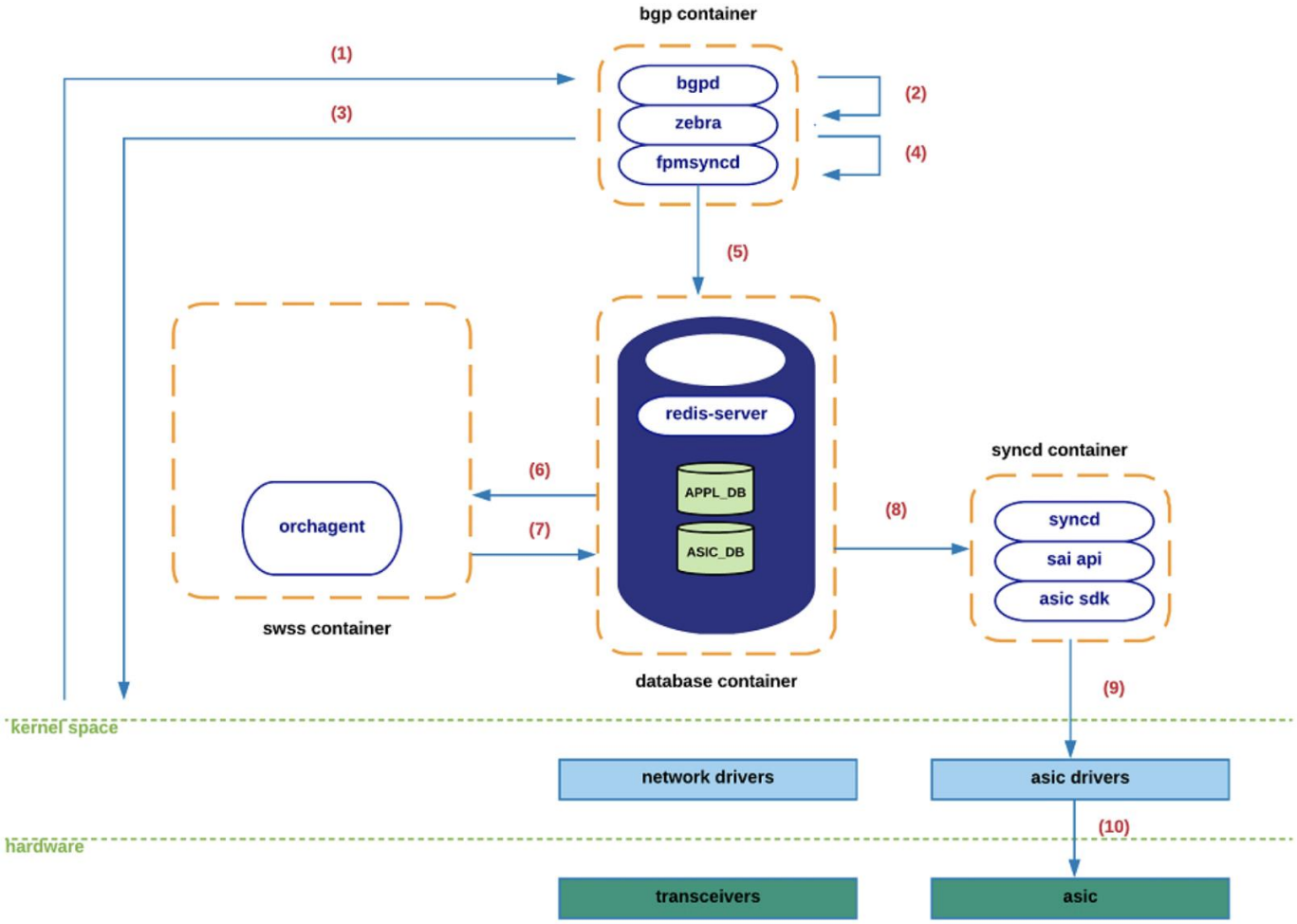
(6) orchagentd is notified about APPL_DB change

(7) orchagentd update ASIC_DB

(8) syncd is notified about ASIC_DB change

(9) Syncd program via SAI API

(10) New route is programed via SDK / ASIC driver

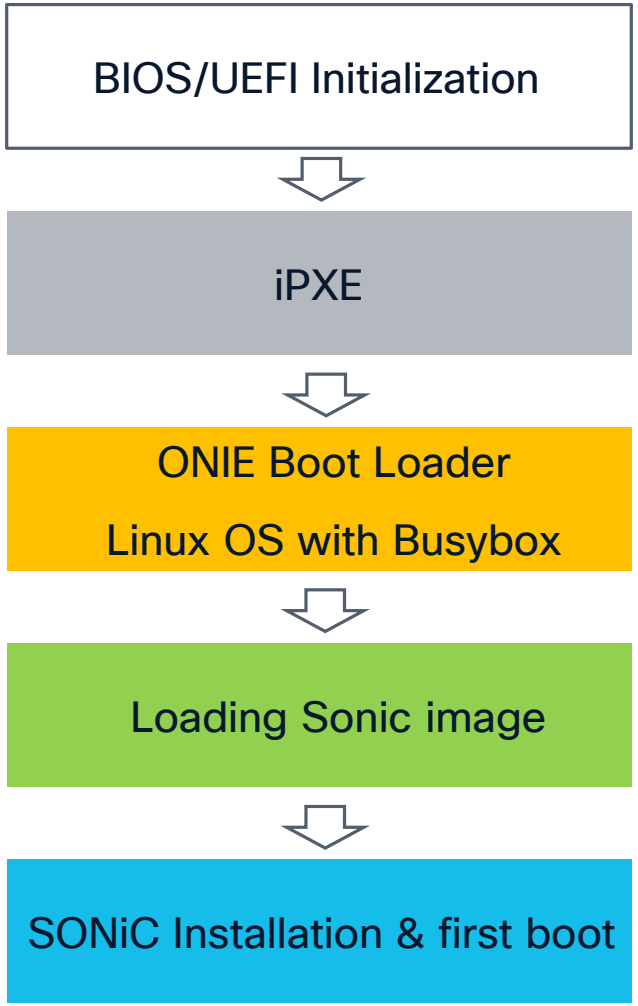


Source: https://github.com/sonic-net/SONiC/wiki/Architecture

Initial Loading & Hello World

SONiC Initial Loading & Installation on Cisco 8000

- SONiC uses Open Network Install Environment (ONIE) as bootloader to install SONiC image
- ONIE is open source Linux installation



SONiC / Cisco 8000 Access – Hello World ☺

- SSH or Console – Linux shell
- Username/Password: cisco/cisco123

```
user@frog:~$ ssh cisco@192.168.122.205
cisco@192.168.122.205's password:
Linux r5 6.1.0-11-2-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.38-4 (2023-08-08) x86_64
You are on
  ____   ___  _   _ _  ____
 / ___| / _ \| \ | (_)/ ___|
 \___ \| | | |  \| | | |
  ___) | |_| | |\  | | |___
 |____/ \___/|_| \_|_|\____|

-- Software for Open Networking in the Cloud --

Unauthorized access and/or use are prohibited.
All access and/or use are subject to monitoring.

Help:    https://sonic-net.github.io/SONiC/
cisco@r5:~$
```

Essential CLI

SONiC – Essential CLI

```
root@r5:/etc/sonic/frr# show ver

SONiC Software Version: SONiC.202405c.25293-dirty-20250521.233043
SONiC OS Version: 12
Distribution: Debian 12.11
Kernel: 6.1.0-11-2-amd64
Build commit: c41ced570
Build date: Thu May 22 11:24:38 UTC 2025
Built by: sonicci@sonic-ci-8-lnx

Platform: x86_64-8201_32fh_o-r0
HwSKU: Cisco-8201-32FH-O
ASIC: cisco-8000
ASIC Count: 1
Serial Number: CSNKCYFRMOM
Model Number: 8201-32FH-O
Hardware Revision: 0.33
```

- See more at https://developer.cisco.com/docs/sonic/

SONiC – Essential CLI

```
root@r5:/etc/sonic/frr# docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED      STATUS      PORTS     NAMES
8f9c1b736d94   docker-snmp:latest                   "/usr/local/bin/supe…"   9 days ago   Up 9 days             snmp
59fed8349aae   docker-platform-monitor:latest       "/usr/bin/docker_ini…"   9 days ago   Up 9 days             pmon
6e56931ceb77   docker-sonic-mgmt-framework:latest   "/usr/local/bin/supe…"   9 days ago   Up 9 days             mgmt-framk
9ef475d09c87   docker-lldp:latest                   "/usr/bin/docker-lld…"   9 days ago   Up 9 days             lldp
4389454f60fe   docker-sonic-gnmi:latest             "/usr/local/bin/supe…"   9 days ago   Up 9 days             gnmi
64ae0f73f39e   8ed070046cbf                         "/usr/bin/docker_ini…"   9 days ago   Up 9 days             dhcp_relay
f5b7e174cc4d   docker-router-advertiser:latest      "/usr/bin/docker-ini…"   9 days ago   Up 9 days             radv
c575f077c3a3   docker-stp:latest                    "/usr/local/bin/supe…"   9 days ago   Up 9 days             stp
10fdb9197fb6   docker-syncd-cisco:latest            "/usr/local/bin/supe…"   9 days ago   Up 9 days             syncd
e6793c59de9a   docker-fpm-frr:latest                "/usr/bin/docker_ini…"   9 days ago   Up 9 days             bgp
bfd4bf568270   docker-teamd:latest                  "/usr/local/bin/supe…"   9 days ago   Up 9 days             teamd
6af70aee8c3d   docker-orchagent:latest              "/usr/bin/docker-ini…"   9 days ago   Up 9 days             swss
18eaec24a306   docker-eventd:latest                 "/usr/local/bin/supe…"   9 days ago   Up 9 days             eventd
f816b4868a88   docker-database:latest               "/usr/local/bin/dock…"   9 days ago   Up 9 days             database
```

- See more at https://developer.cisco.com/docs/sonic/

SONiC – Essential CLI

```
root@r5:/etc/sonic# show interface status
   Interface                                  Lanes    Speed    MTU    FEC    Alias          Vlan    Oper    Admin     C
------------  ---------------------------------------  -------  -----  -----  -------  ------------  ------  -------  ----
   Ethernet0  2304,2305,2306,2307,2308,2309,2310,2311     400G   9100    N/A     etp0  PortChannel1      up       up  QSFA
   Ethernet8  2320,2321,2322,2323,2324,2325,2326,2327     400G   9100    N/A     etp1        routed      up       up  QSFA
  Ethernet16  2312,2313,2314,2315,2316,2317,2318,2319     400G   9100    N/A     etp2        routed      up       up  QSFA
  Ethernet24  2056,2057,2058,2059,2060,2061,2062,2063     400G   9100    N/A     etp3        routed      up       up  QSFA
.........
```

MGMT Interface IP configuration

```
config interface ip add eth0 192.168.122.205/24
```

- See more at https://developer.cisco.com/docs/sonic/

SONiC – Platform check

```
root@r8:/home/cisco# show platform summary
Platform: x86_64-8201_32fh_o-r0
HwSKU: Cisco-8201-32FH-O
ASIC: cisco-8000
ASIC Count: 1
Serial Number: CSNDS2HKXFT
Model Number: 8201-32FH-O
Hardware Revision: 0.33
root@r8:/home/cisco#

root@r8:/home/cisco# show platform npu resource
+-------------------------------------------------+-------------+----------+-------------+--------------+-------+--------+
|                     Resource                    | Granularity | Location | Max Entries | Used Entries | State | Thresho|
+-------------------------------------------------+-------------+----------+-------------+--------------+-------+--------+
|                   ACL_GROUP                     |    Device   |    -     |     255     |      0       | Green |        |
|                   ACL_GROUP                     |    Device   |    -     |     255     |      0       | Green |        |
|                   ACL_GROUP                     |    Device   |    -     |     255     |      0       | Green |        |
|                   AC_PROFILE                    |     None    |    -     |      15     |      2       | Green |        |
|            ADDITIONAL_LABELS_TABLE              |    Device   |    -     |    16384    |      0       | Green |        |
.........
```

- See more at https://developer.cisco.com/docs/sonic/

Configuration Changes & Upgrade

SONiC Configuration

- Redis engine **CONFIG_DB** - running configuration
- **/etc/sonic/config_db.json** – startup configuration
- Only root or “sudo” can change configuration
  - `config save`
  - `config reload`
  - `reset-factory`
  - `config load`

```
root@r5:/home/cisco# config save
Existing files will be overwritten, continue? [y/N]: y
Running command: /usr/local/bin/sonic-cfggen -d --print-data > /etc/sonic/config_db.json
```

- Configuration can be changed via python based click (SONiC) CLI or by modifying **/etc/sonic/config_db.json**
- Click CLI provides also show commands

- Watch offline **DEVNET-2990 presented by Suhaib Ahmad**
  - Configuration modes
  - How to use Ansible to manage SONiC

FRR Configuration options

- FRR configuration is stored in **/etc/sonic/config_db.json** by default
  - See more files in **/etc/sonic**
- Configuration done via FRR shell (vtysh) is lost after **`reboot`** or **`config reload`**
  - FRR configuration changes should be done via SONiC CLI or via **/etc/sonic/config_db.json** directly

- FRR configuration can be stored in separate config file(s)
  - Add into **/etc/sonic/config_db.json:**

```
"localhost": {
    "docker_routing_config_mode": "split-unified"
```

  - `echo service integrated-vtysh-config >/etc/sonic/frr/vtysh.conf`
  - `config load`
- FRR configuration is stored in: `/etc/sonic/frr/frr.conf`
  - FRR configuration must be saved in FRR shell (vtysh)

```
r5# copy running-config startup-config
Building Configuration...
Integrated configuration saved to /etc/frr/frr.conf
[OK]
```

SONiC – Upgrade

- iPXE -> ONIE – same as initial Installation

  or

- Copy new file to local directory (scp)
- sonic-installer install /home/images/sonic-cisco-8000.bin
- Load new image:
  - **reload (cold)** – Data Plane Impacted
  - **warm-reboot** – No Data Plane Impact
  - **fast-reboot** – Data Plane reset ~30s interrupt
  - **express-reboot** – Data Plane is not reset, NPU updated via DMA, DP interrupt ~1s

Port Speed change / Breakout

- Edit `/usr/share/sonic/device/<Platform>/<HwSKU>/platform_cisco_cfg.yaml`

```
port_mode:
    -
      ports: 0-31
      mode : 1x400G
      media: OPTIC
```

- Additional port speed / breakout can be added:
  - `/usr/share/sonic/device/<Platform>/asic_info.yaml`

```
brkoutmode: ['4x100G', '4x25G', '2x200G', '2x100G','1x10G','1x40G']
  intf:
    mode: 1x10G
      lanes: 1
      speed: 10000
    -
      mode: 1x40G
      lanes: 4
      speed: 10000
```

- Validate configuration changes: `platform_cisco_cfg.py`
- Delete `/etc/sonic/config_db.json` and `reload`

```
# port_mode: options (example)
platform_cisco_cfg.yaml
#    -
#       ports: 1,2,3
#       mode: 1x400G
#       media: OPTIC
#    -
#       ports: 4,5,6
#       mode: 1x200G
#    -
#       ports: 7,8,9
#       speed: 1x100G
#       media: OPTIC
#    -
#       ports: 10,11,12
#       mode: 2x200G
#       media: OPTIC
#    -
#       ports: 13,14,15
#       mode: 2x100G
#       media: OPTIC
#    -
#       ports: 16,17
#       mode: 4x100G
#       media: OPTIC
```

SONiC Usecases

SONiC Main Usecases

#1 IP Fabric – DC / AI Front-End

- Spine/Leaf CLOS
- IPv4 or IPv6 transport
- eBGP IPv4/IPv6 Underlay – Leaf-Spine hop-by-hop eBGP session
- No extra encapsulation (like VXLAN)
- No VRF
- No L2

#2 L3 VXLAN (L3VPN) or/and L2 VXLAN (L2VPN ELAN) – DC / AI Front-End

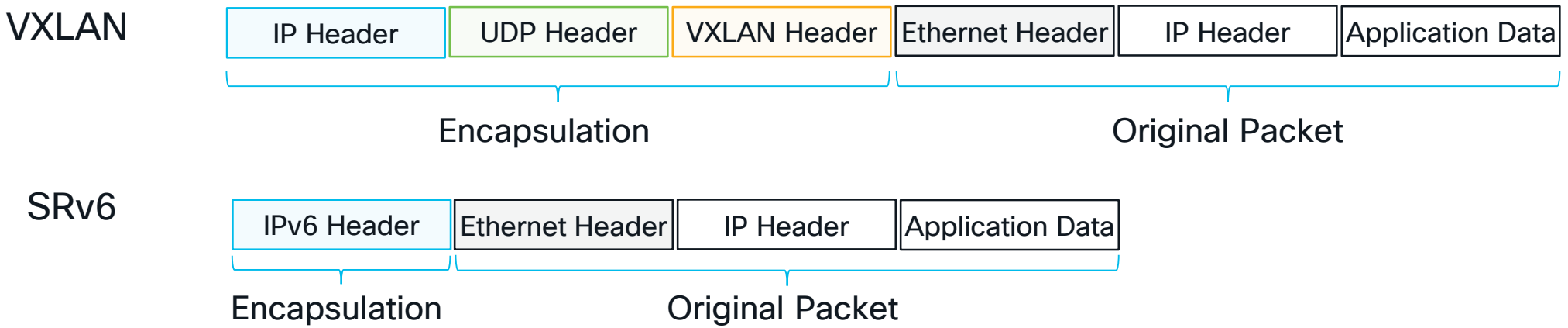
- Based on #1 IP Fabric
- Spine/Leaf CLOS
- IPv4 or IPv6 transport
- eBGP IPv4/IPv6 Underlay – Leaf-Spine hop-by-hop eBGP session
- eBGP EVPN Overlay - BGP Route Server to scale number of eBGP EVPN sessions
- L3VNI / L2VNI
- VRF / Bridge - multitenancy

#3 IP Fabric - AI Back-End

- Based on **#1 IP Fabric** – possible extension to **#2 L3 VXLAN** for multitenancy
- Transport ROCEv2 – RDMA over Ethernet over IP/UDP
- Focused on optimal loadbalancing
  - PFC/ECN
  - FlowLet Load Balancing
  - Packet Spraying
  - Static Balancing / Pinning – SRv6 Traffic Engineering
  - Packet trimming, Congestion Signaling (CSIG), ...

L2VPN Services Overlay Encapsulation

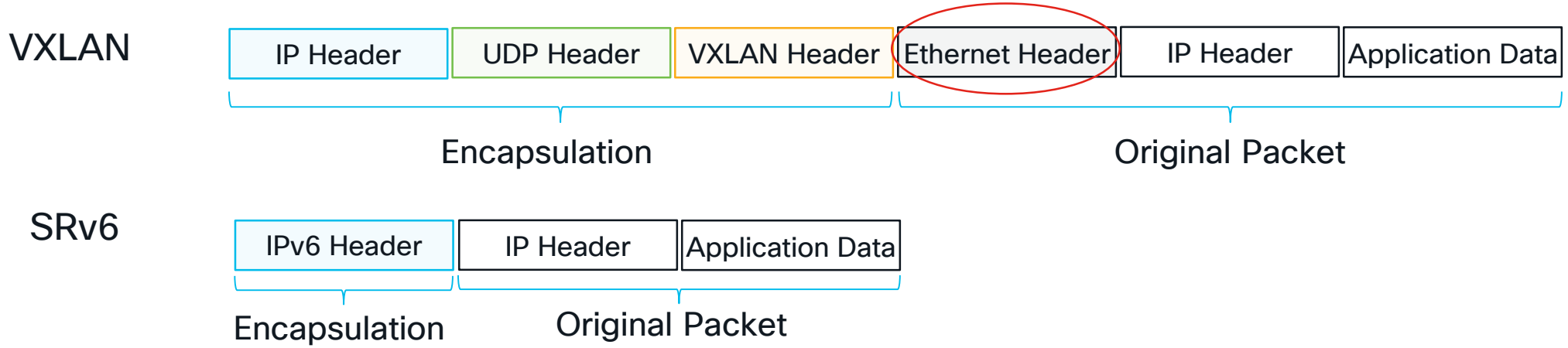
IP Data Plane



L3VPN Services Overlay Encapsulation

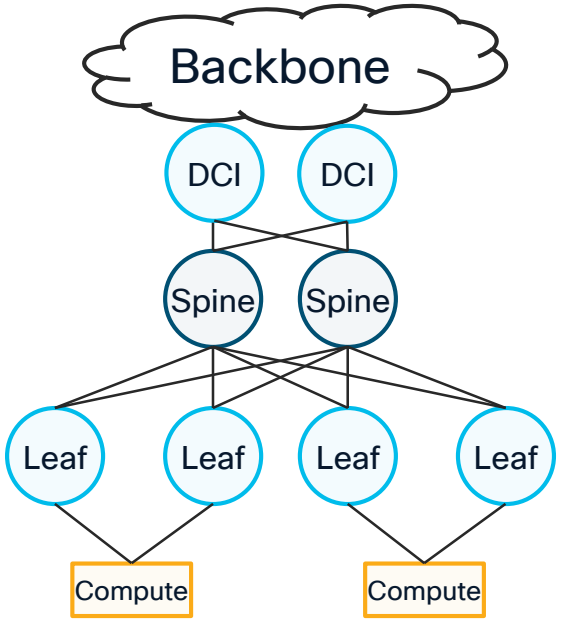
IP Data Plane

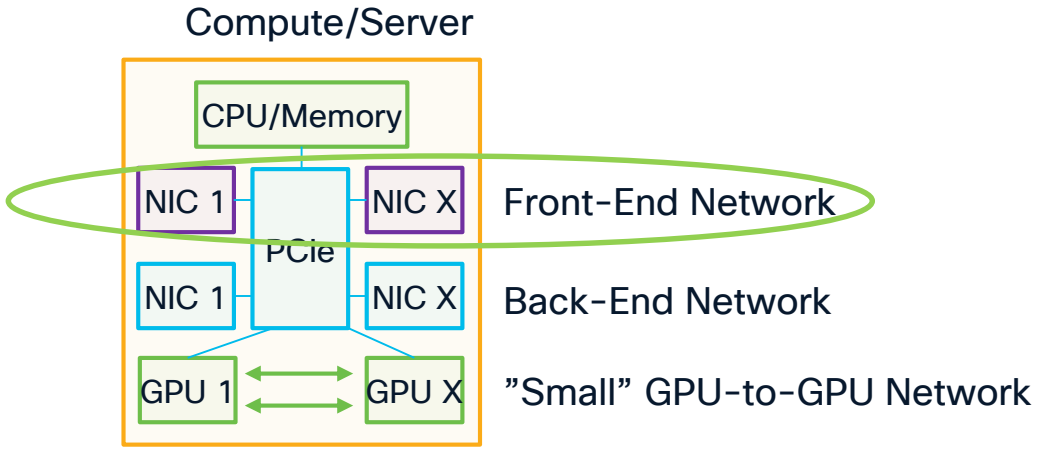
VXLAN RFC7348 - Requires Inner Ethernet
- Additional overhead for L3VPN / IP Forwarding



Front-End Network / DC

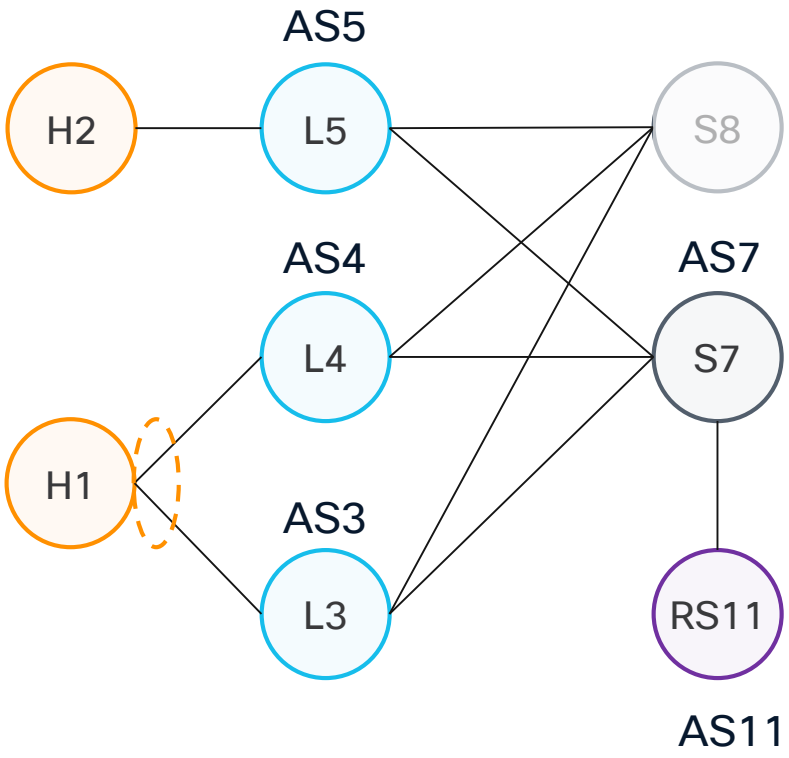
- Network connecting compute to outside world or to storage
- **Well-Known Data Center Standard design (CLOS):**
  - Spine/Leaf
  - IP Transport
  - Flow-Based ECMP
  - EVPN Overlay
    - High Availability
    - Multitenancy
    - L2/L3 Transparency





IP Fabric

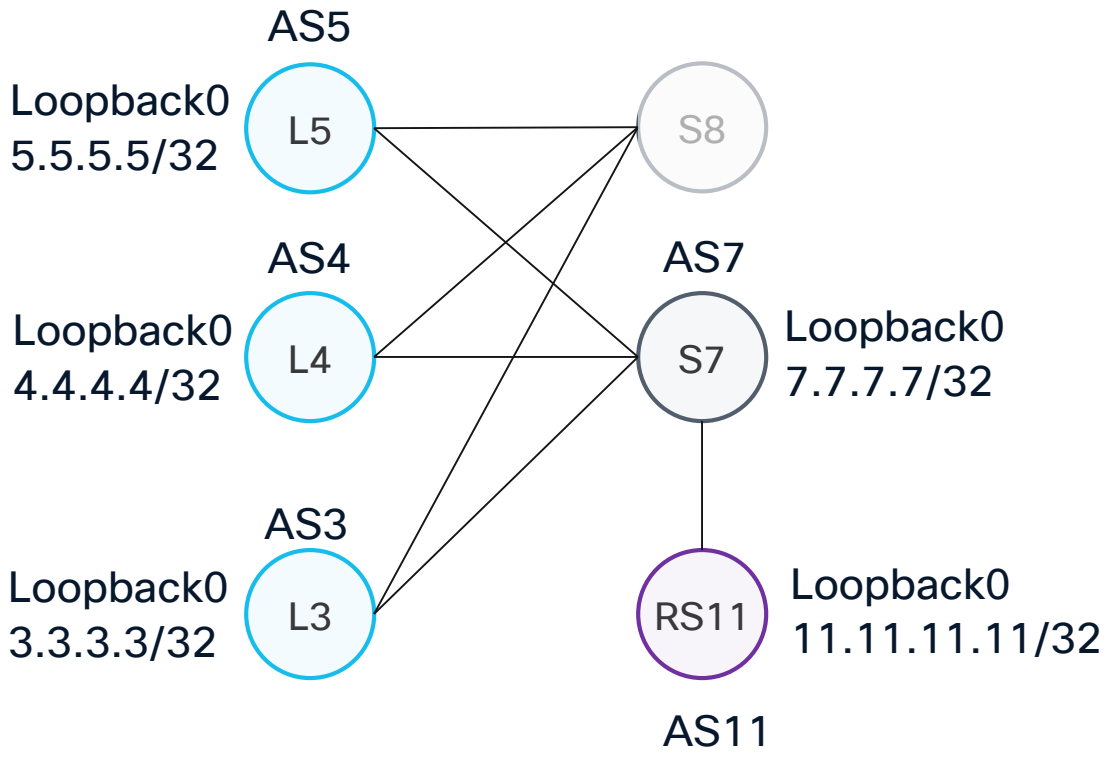
IP VXLAN with EVPN All-Active Multihoming - Testbed



IP / eBGP Configuration

L3 SONiC CLI

```
config interface ip add Loopback0 3.3.3.3/32
config interface ip add etp10 3.7.1.3/24
```



L3 FRR vtysh (Cisco IOS style CLI)

```
router bgp 3
 bgp router-id 3.3.3.3
 no bgp ebgp-requires-policy
 no bgp default ipv4-unicast
 bgp bestpath as-path multipath-relax
 neighbor pass peer-group
 neighbor 3.7.1.7 remote-as 7
 !
 address-family ipv4 unicast
  redistribute connected route-map loopback
  neighbor 3.7.1.7 activate
 exit-address-family
 !
ip prefix-list loopback seq 5 permit 0.0.0.0/0 ge 32
!
route-map loopback permit 10
 match ip address prefix-list loopback
Exit
```

IP / eBGP Configuration Validation

L3 SONiC CLI

show ip interface

```
Interface    Master    IPv4 address/mask    Admin/Oper    BGP Neighbor    Neighbor IP
------  --------  ------------------  ------------  --------------  -------------
etp10                3.7.1.3/24           up/up         N/A             N/A
Loopback0            3.3.3.3/32           up/up         N/A             N/A
docker0              240.127.1.1/24       up/down       N/A             N/A
eth0                 192.168.122.203/24   up/up         N/A             N/A
eth4                 192.168.123.61/24    up/up         N/A             N/A
lo                   127.0.0.1/16         up/up         N/A             N/A
```

show arp

```
Address         MacAddress          Iface   Vlan
-------------  -----------------  -----  ------
3.7.1.7        78:66:a8:cf:80:00  etp10  -
192.168.122.1  7e:df:61:31:ca:51  eth0   -
192.168.123.1  ae:08:c4:72:f4:8c  eth4   -
```

show ip route

```
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP,
       F - PBR, f - OpenFabric,
       > - selected route, * - FIB route, q - queued route, r - rejected route

K>*0.0.0.0/0 [0/0] via 192.168.123.1, eth4, 5d15h59m
C>*3.3.3.3/32 is directly connected, Loopback0, 5d15h59m
C>*3.7.1.0/24 is directly connected, etp10, 5d15h59m
B>*4.4.4.4/32 [20/0] via 3.7.1.7, etp10, 5d15h59m
B>*5.5.5.5/32 [20/0] via 3.7.1.7, etp10, 5d15h59m
B>*7.7.7.7/32 [20/0] via 3.7.1.7, etp10, 5d15h59m
B>*11.11.11.11/32 [20/0] via 3.7.1.7, etp10, 5d15h59m
C>*192.168.122.0/24 is directly connected, eth0, 5d15h59m
C>*192.168.123.0/24 is directly connected, eth4, 5d15h59m
```

IP / eBGP Configuration Validation

L3 Linux native tools to validate Linux Kernel state

ip route

```
default via 192.168.123.1 dev eth4
3.7.1.0/24 dev Ethernet80 proto kernel scope link src 3.7.1.3
4.4.4.4 via 3.7.1.7 dev Ethernet80 proto bgp src 3.3.3.3 metric 20
5.5.5.5 via 3.7.1.7 dev Ethernet80 proto bgp src 3.3.3.3 metric 20
7.7.7.7 via 3.7.1.7 dev Ethernet80 proto bgp src 3.3.3.3 metric 20
11.11.11.11 via 3.7.1.7 dev Ethernet80 proto bgp src 3.3.3.3 metric 20
192.168.122.0/24 dev eth0 proto kernel scope link src 192.168.122.203
192.168.123.0/24 dev eth4 proto kernel scope link src 192.168.123.61
240.127.1.0/24 dev docker0 proto kernel scope link src 240.127.1.1 linkdown
```

cat /proc/net/arp

IP address	HW type	Flags	HW address	Mask	Device
192.168.122.1	0x1	0x2	7e:df:61:31:ca:51	*	eth0
3.7.1.7	0x1	0x2	78:66:a8:cf:80:00	*	Ethernet80
192.168.123.1	0x1	0x2	ae:08:c4:72:f4:8c	*	eth4

netstat -r

Kernel IP routing table

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
default	192.168.123.1	0.0.0.0	UG	0	0	0	eth4
3.7.1.0	0.0.0.0	255.255.255.0	U	0	0	0	Ethernet80
4.4.4.4	3.7.1.7	255.255.255.255	UGH	0	0	0	Ethernet80
5.5.5.5	3.7.1.7	255.255.255.255	UGH	0	0	0	Ethernet80
7.7.7.7	3.7.1.7	255.255.255.255	UGH	0	0	0	Ethernet80
11.11.11.11	3.7.1.7	255.255.255.255	UGH	0	0	0	Ethernet80
192.168.122.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
192.168.123.0	0.0.0.0	255.255.255.0	U	0	0	0	eth4
240.127.1.0	0.0.0.0	255.255.255.0	U	0	0	0	docker0

IP / eBGP Configuration Validation

L3 FRR vtysh

```
show bgp ipv4 unicast summary
BGP router identifier 3.3.3.3, local AS number 3 vrf-id 0
……
Neighbor        V         AS   MsgRcvd   MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd   PfxSnt Desc
3.7.1.7         4          7      8169      8169        0    0    0 5d16h01m            4        5 N/A
```

```
show bgp ipv4 unicast
BGP table version is 7, local router ID is 3.3.3.3, vrf id 0
Default local pref 100, local AS 3
Status codes:  s suppressed, d damped, h history, * valid, > best, = multipath,
               i internal, r RIB-failure, S Stale, R Removed
Nexthop codes: @NNN nexthop's vrf id, < announce-nh-self
Origin codes:  i - IGP, e - EGP, ? - incomplete
RPKI validation codes: V valid, I invalid, N Not found

    Network          Next Hop            Metric LocPrf Weight Path
 *> 3.3.3.3/32       0.0.0.0                  0         32768 ?
 *> 4.4.4.4/32       3.7.1.7                                0 7 4 ?
 *> 5.5.5.5/32       3.7.1.7                                0 7 5 ?
 *> 7.7.7.7/32       3.7.1.7                  0             0 7 ?
 *> 11.11.11.11/32   3.7.1.7                                0 7 11 ?
```

```
show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric, J - Adjacency,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

K>* 0.0.0.0/0 [0/0] via 192.168.123.1, eth4, 5d16h01m
C>* 3.3.3.3/32 is directly connected, Loopback0, 5d16h01m
C>* 3.7.1.0/24 is directly connected, Ethernet80, 5d16h01m
B>* 4.4.4.4/32 [20/0] via 3.7.1.7, Ethernet80, rmapsrc 3.3.3.3, weight 1, 5d16h01m
B>* 5.5.5.5/32 [20/0] via 3.7.1.7, Ethernet80, rmapsrc 3.3.3.3, weight 1, 5d16h01m
B>* 7.7.7.7/32 [20/0] via 3.7.1.7, Ethernet80, rmapsrc 3.3.3.3, weight 1, 5d16h01m
B>* 11.11.11.11/32 [20/0] via 3.7.1.7, Ethernet80, rmapsrc 3.3.3.3, weight 1, 5d16h01m
C>* 192.168.122.0/24 is directly connected, eth0, 5d16h01m
C>* 192.168.123.0/24 is directly connected, eth4, 5d16h01m
```

All-Active EVPN with VXLAN Data Plane L2VPN

BGP EVPN ELAN Signaling
Route Types

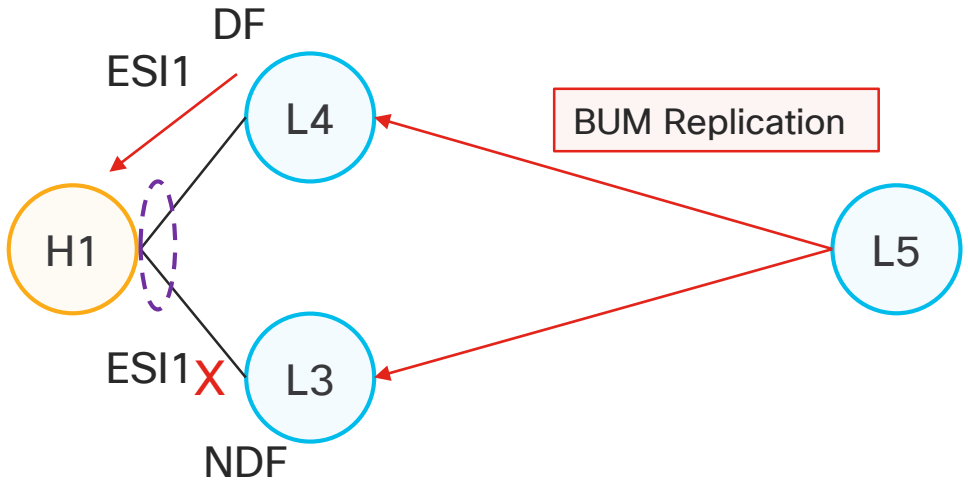
RT4 – Designated Forwarder (DF) Election
RT3 – BUM signaling
RT2 – MAC Advertisement
RT1 per ESI
- Multihoming type
  - **All-Active**
  - Single-Active
  - Port-Active
- Mass withdraw
- Peering PEs Split-Horizon / Local Bias

RT1 per EVI / ESI
- MAC address aliasing (Multipath)

BGP EVPN ELAN signaling mechanism is almost same for MPLS, VXLAN, SRv6

BGP EVPN ELAN Signaling
Route Type 4

RT4 – Designated Forwarder (DF) Election



Prevents duplicate copies of flooded traffic from remote Leaf being delivered to H1

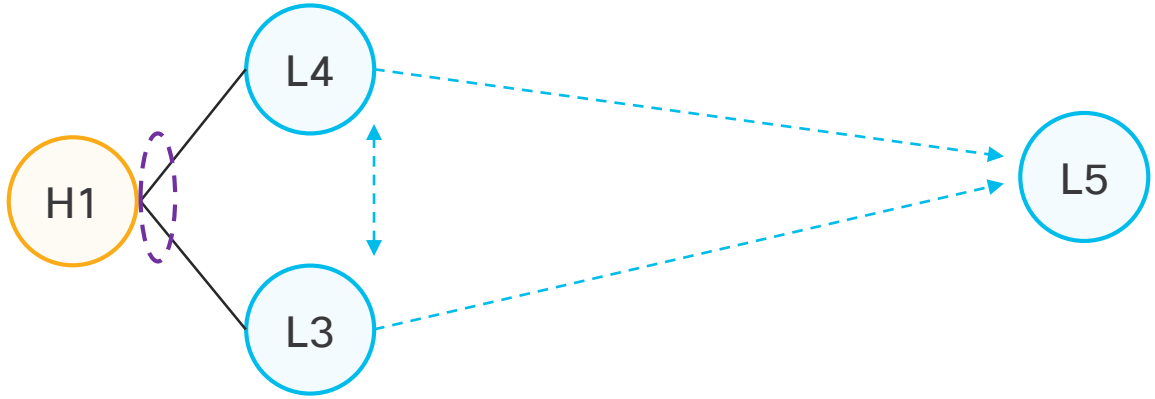
- Only DF elected Leaf forwards Broadcast, Unknown-Unicast or Multicast (BUM) to H1
- DF Election is performed for each Ethernet Segment Identifier (ESI)

Filtering based on DF election on Leaf applies to BUM traffic only

BGP EVPN ELAN Signaling
Route Type 3

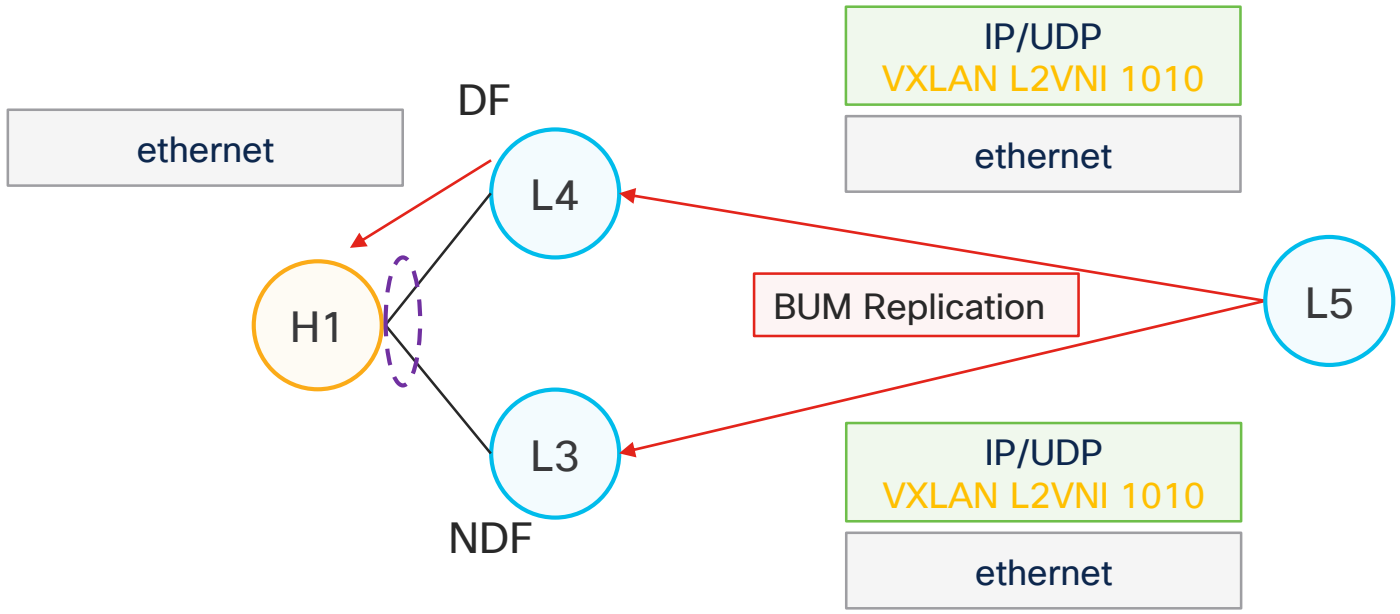
RT3 – signaling

L3/L4/L5 Orange BD
L2VNI: 1010, Replication Type (ingress Replication)



BGP EVPN ELAN Signaling
Route Type 3

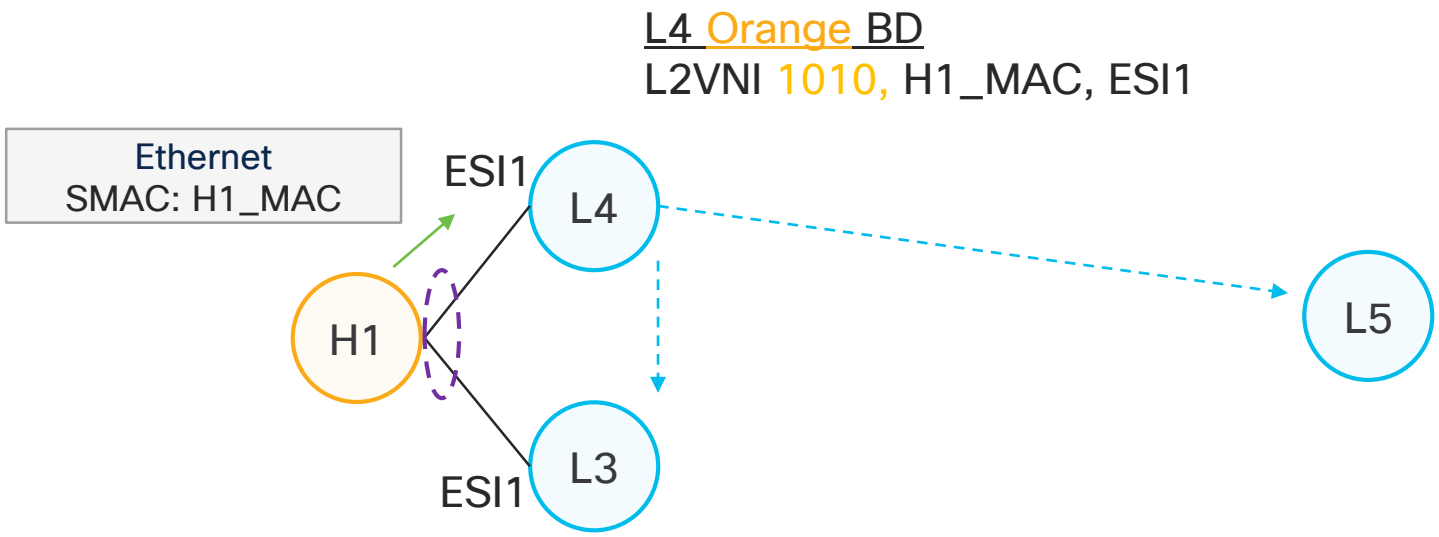
RT3 – signaling



Remote Leaf Ingress Replicates BUM traffic to all Leaves connected in Orange EVI
Filtering based on DF election on Leaf applies to BUM traffic only

BGP EVPN ELAN Signaling
Route Type 2

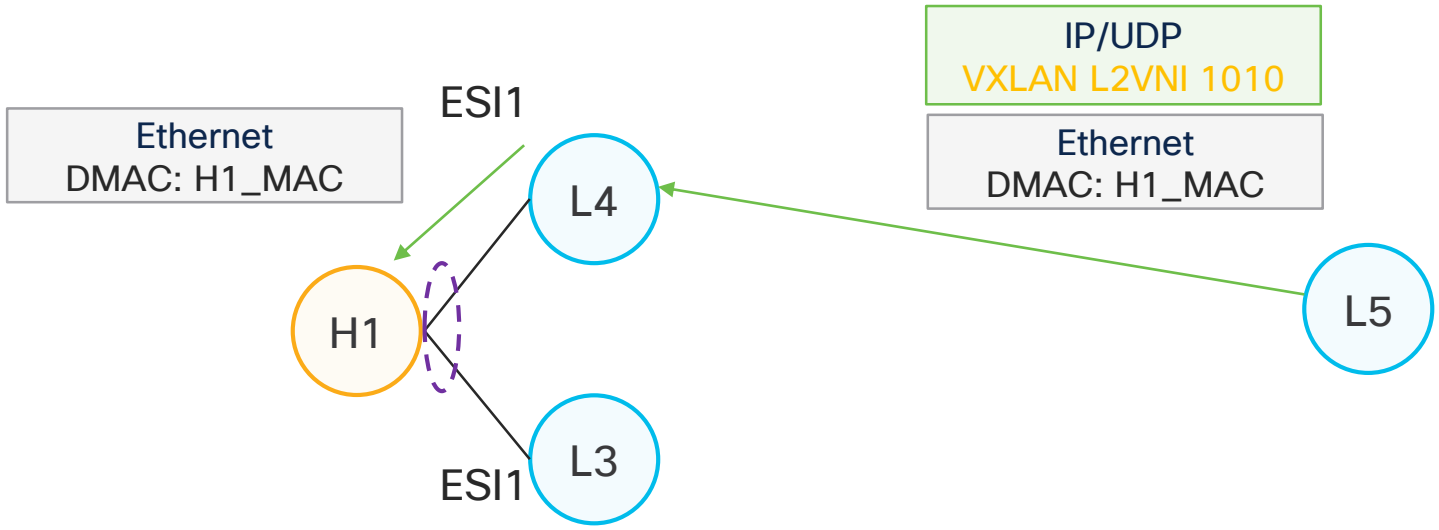
RT2 – MAC Advertisement



L4 learns H1 MAC via dataplane on local L2 attachment circuit (AC)
H1 MAC advertises together with:
ESI – remote Leaf can loadbalance via L3 and L4 (see BGP EVPN RT1 for more detail)

BGP EVPN ELAN Signaling
Route Type 2

RT2 – MAC Advertisement



BGP EVPN ELAN Signaling
Route Type 1

RT1 per ESI

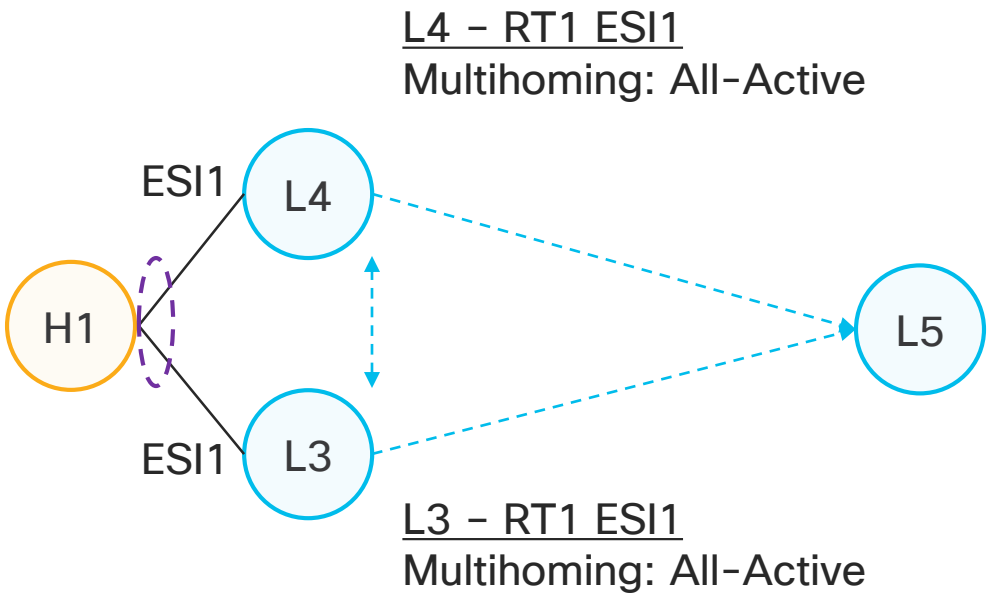
- Multihoming type
  - **All-Active**
  - Single-Active
  - Port-Active
- Mass withdraw

RT1 per EVI / ESI

- MAC address aliasing (Multipath)

BGP EVPN ELAN Signaling
Route Type 1 per ESI

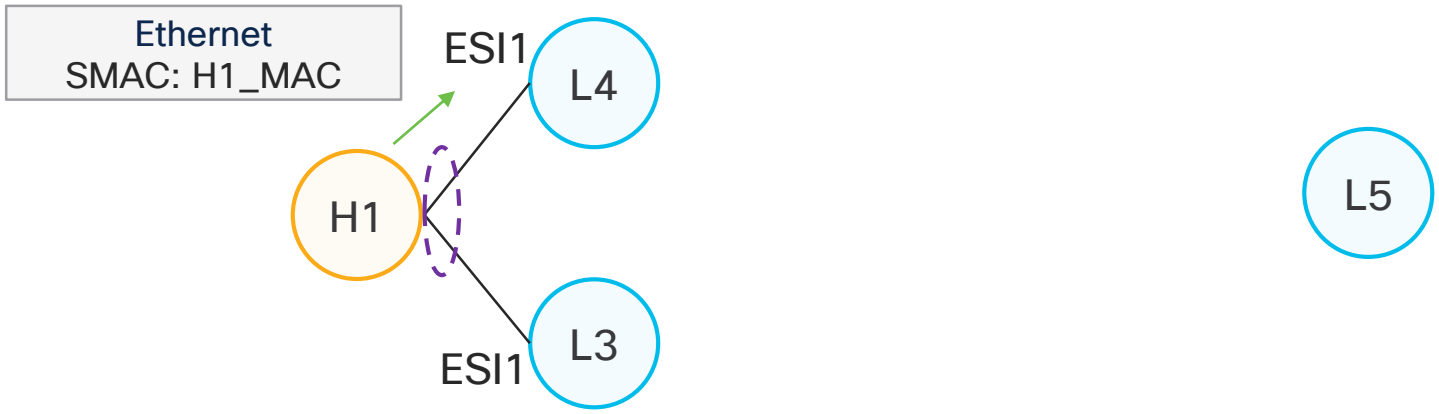
RT1 per ESI
- Multihoming type
  - All-Active



Each Leaf pair must identify multihomed ethernet segment by assigning Ethernet Segment Identifier (ESI)
ESI value is signaled together with
Multihoming type - Remote nodes(L5) know how to loadbalance unicast traffic

BGP EVPN ELAN Signaling
Route Type 1

RT1 per EVI / ESI – L2VNI signaling



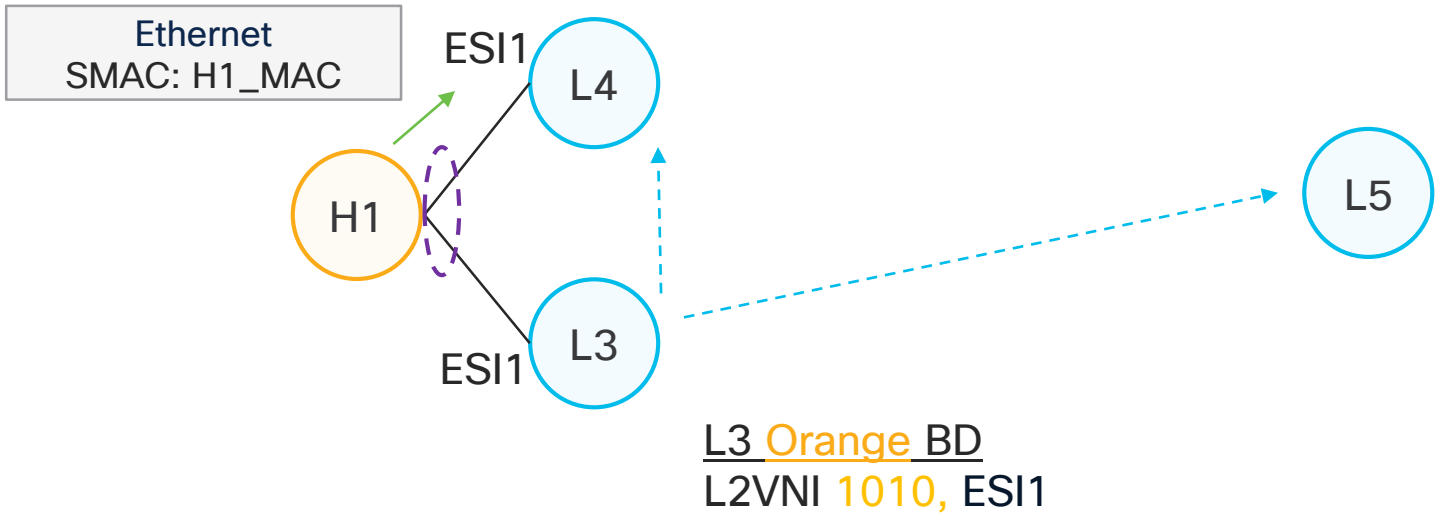
L5 Forwarding Information

H1 MAC -> ESI1-> L4
-> L3 UNRESOLVED
ESI1 is All-Active
per flow loadbalancing to L3 and L4

L3 L2VNI?

BGP EVPN ELAN Signaling
Route Type 1

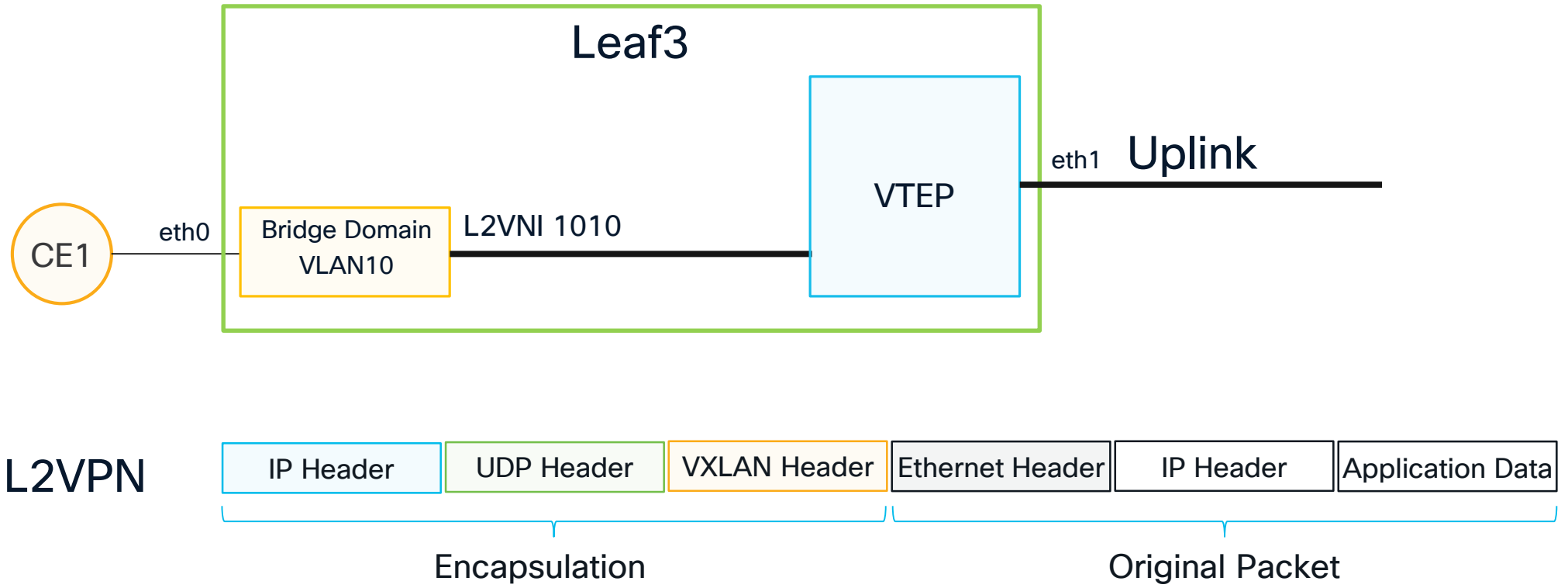
RT1 per EVI / ESI – L2VNI signaling



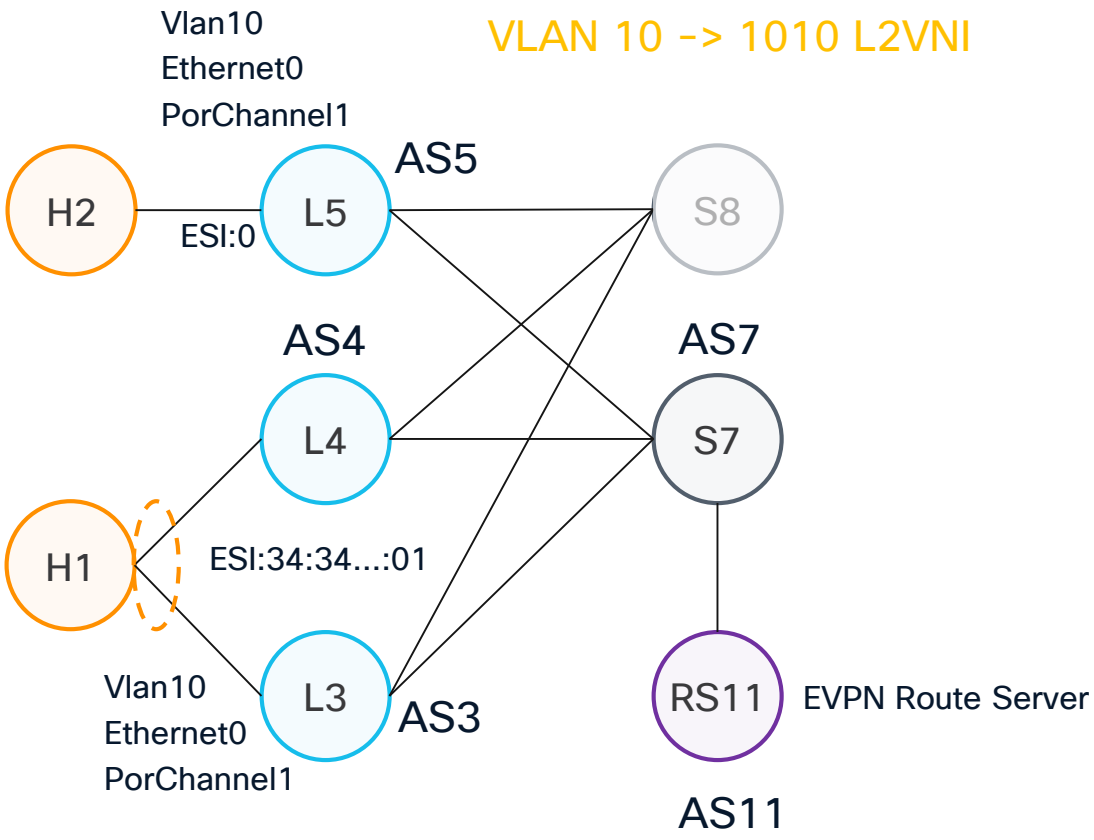
L5 Forwarding Information

H1 MAC -> ESI1-> L4
-> L3
ESI1 is All-Active
per flow loadbalancing to L3 and L4

VXLAN Data Plane Processing – not only in SONiC



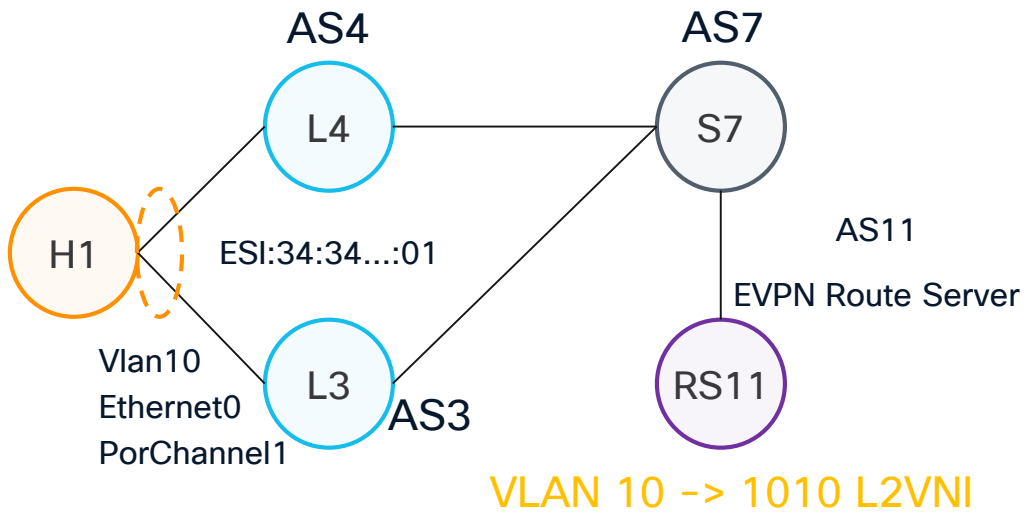
IP VXLAN with EVPN All-Active Multihoming - Testbed



L2VPN: IP VXLAN / EVPN Control Plane – Configuration

L3 SONiC CLI

```
config portchannel add PortChannel1
config portchannel member add PortChannel1 Ethernet0
config vlan add 10
config vlan member add 10 PortChannel1

config vxlan add VXLAN 3.3.3.3
config vxlan evpn_nvo add NVO VXLAN
config vxlan map add VXLAN 10 1010

config interface evpn-esi add PortChannel1 00:34:34:34:34:34:34:34:34:01
[config interface evpn-df-pref PortChannel1 100]
config interface sys-mac add PortChannel1 00:bb:bb:bb:bb:bb
```



L3 FRR vtysh

```
router bgp 3
 bgp router-id 3.3.3.3
 no bgp ebgp-requires-policy
 no bgp default ipv4-unicast
 bgp bestpath as-path multipath-relax
 neighbor evpn peer-group
 neighbor evpn ebgp-multihop 10
 neighbor evpn disable-connected-check
 neighbor evpn update-source Loopback0
 neighbor 11.11.11.11 remote-as 11
 neighbor 11.11.11.11 peer-group evpn
 neighbor 3.7.1.7 remote-as 7
 !
 address-family ipv4 unicast
  …… [previous config]
 !
 address-family l2vpn evpn
  neighbor evpn activate
  advertise-all-vni
  vni 1010
   route-target import 1010:1010
   route-target export 1010:1010
  exit-vni
 exit-address-family
evpn mh redirect-off
```

```
interface PortChannel1
 evpn mh es-id 00:34:34:34:34:34:34:34:34:01
exit
```

L2VPN: IP VXLAN / EVPN Control Plane – Configuration

Route Server FRR vtysh

```
router bgp 11
 bgp router-id 11.11.11.11
 no bgp ebgp-requires-policy
 no bgp default ipv4-unicast
 bgp bestpath as-path multipath-relax
 neighbor evpn peer-group
 neighbor evpn ebgp-multihop 10
 neighbor evpn disable-connected-check
 neighbor evpn update-source Loopback0
 neighbor pass peer-group
 neighbor 7.11.1.7 remote-as 7
 neighbor 3.3.3.3 remote-as 3
 neighbor 4.4.4.4 remote-as 4
 neighbor 5.5.5.5 remote-as 5
 neighbor 6.6.6.6 remote-as 6
 neighbor 3.3.3.3 peer-group evpn
 neighbor 4.4.4.4 peer-group evpn
 neighbor 5.5.5.5 peer-group evpn
 neighbor 6.6.6.6 peer-group evpn
 !
 address-family l2vpn evpn
  neighbor evpn activate
  neighbor evpn attribute-unchanged as-path next-hop med
 exit-address-family
Exit
…… [ipv4 address family,etc…]
```

L2VPN: IP VXLAN / EVPN Control Plane – Validation

Remote R5 SONiC CLI

```
/sbin/bridge fdb | grep "vlan 10"
78:2f:db:6c:90:00 dev PortChannel1 vlan 10 extern_learn master Bridge
78:b4:2c:55:d0:00 dev VXLAN-10 vlan 10 extern_learn master Bridge

/sbin/bridge fdb | grep VXLAN-10
78:b4:2c:55:d0:00 dev VXLAN-10 vlan 10 extern_learn master Bridge
78:b4:2c:55:d0:00 dev VXLAN-10 nhid 536870913 self extern_learn

ip nexthop
id 268435458 via 4.4.4.4 scope link fdb
id 268435459 via 3.3.3.3 scope link fdb
id 536870913 group 268435459/268435458 fdb

show evpn es detail
ESI: 00:34:34:34:34:34:34:34:01
 Type: Remote
 Interface: -
 Ready for BGP: no
 VNI Count: 0
 MAC Count: 1
 DF preference: 0
 Nexthop group: 536870913
 VTEPs:
     3.3.3.3 nh: 268435459
     4.4.4.4 nh: 268435458
```

L2VPN: IP VXLAN / EVPN Control Plane – Validation

Remote R5 SONiC CLI

show vxlan tunnel

```
vxlan tunnel name    source ip    destination ip    tunnel map name    tunnel map mapping(vni -> vlan)
-------------------  -----------  ----------------  -----------------  --------------------------------
VXLAN                5.5.5.5                        map_1010_Vlan10    1010 -> Vlan10
```

show vxlan vlanvnimap

```
+---------+-------+
| VLAN    |   VNI |
+=========+=======+
| Vlan10  |  1010 |
+---------+-------+
```

show vxlan remotevtep

```
+---------+---------+-------------------+--------------+
| SIP     | DIP     | Creation Source   | OperStatus   |
+=========+=========+===================+==============+
| 5.5.5.5 | 3.3.3.3 | EVPN              | oper_up      |
+---------+---------+-------------------+--------------+
| 5.5.5.5 | 4.4.4.4 | EVPN              | oper_up      |
+---------+---------+-------------------+--------------+
```

show vxlan remotemac all

```
+---------+-------------------+----------------+-------+---------+
| VLAN    | MAC               | RemoteTunnel   |   VNI | Type    |
+=========+===================+================+=======+=========+
| Vlan10  | 78:b4:2c:55:d0:00 | 3.3.3.3        |  1010 | dynamic |
|         |                   | 4.4.4.4        |       |         |
+---------+-------------------+----------------+-------+---------+
```

show vxlan remotevni all

```
+--------+--------------+-------+
| VLAN   | RemoteVTEP   |   VNI |
+========+==============+=======+
| Vlan10 | 3.3.3.3      |  1010 |
+--------+--------------+-------+
| Vlan10 | 4.4.4.4      |  1010 |
+--------+--------------+-------+
```

L2VPN: IP VXLAN / EVPN Control Plane – Validation

Remote R5 FRR vtysh

```
show bgp l2vpn evpn
……
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 3.3.3.3:2
 *> [1]:[0]:[00:34:34:34:34:34:34:34:34:01]:[32]:[0.0.0.0]:[0]
                    3.3.3.3                                0 3 i
                    RT:1010:1010 ET:8
 *> [2]:[0]:[48]:[78:b4:2c:55:d0:00]
                    3.3.3.3                                0 3 i
                    ESI:00:34:34:34:34:34:34:34:34:01
                    RT:1010:1010 ET:8
 *> [3]:[0]:[32]:[3.3.3.3]
                    3.3.3.3                                0 3 i
                    RT:1010:1010 ET:8
Route Distinguisher: 3.3.3.3:3
 *> [1]:[4294967295]:[00:34:34:34:34:34:34:34:34:01]:[32]:[0.0.0.0]:[0]
                    3.3.3.3                                0 3 i
                    RT:1010:1010 ET:8 ESI-label-Rt:AA
 *> [4]:[00:34:34:34:34:34:34:34:34:01]:[32]:[3.3.3.3]
                    3.3.3.3                                0 3 i
                    ET:8 ES-Import-Rt:34:34:34:34:34:34 DF: (alg: 2, pref: 32767)
```

L2VPN: IP VXLAN / EVPN Control Plane – Validation

Remote R5 FRR vtysh

```
how bgp l2vpn evpn
……
Route Distinguisher: 5.5.5.5:4
 *> [2]:[0]:[48]:[78:2f:db:6c:90:00]
                    5.5.5.5                              32768 i
                    ET:8 RT:1010:1010
 *> [3]:[0]:[32]:[5.5.5.5]
                    5.5.5.5                              32768 i
                    ET:8 RT:1010:1010
```

```
show evpn es
Type: B bypass, L local, R remote, N non-DF
ESI                             Type ES-IF                 VTEPs
00:34:34:34:34:34:34:34:34:01  R    -                     3.3.3.3,4.4.4.4

show evpn mac vni all

VNI 1010 #MACs (local and remote) 2

Flags: N=sync-neighs, I=local-inactive, P=peer-active, X=peer-proxy, L=local
MAC               Type   Flags Intf/Remote ES/VTEP            VLAN  Seq #'s
78:2f:db:6c:90:00 local  L      PortChannel1                   10    0/0
78:b4:2c:55:d0:00 remote        00:34:34:34:34:34:34:34:34:01        0/0
```

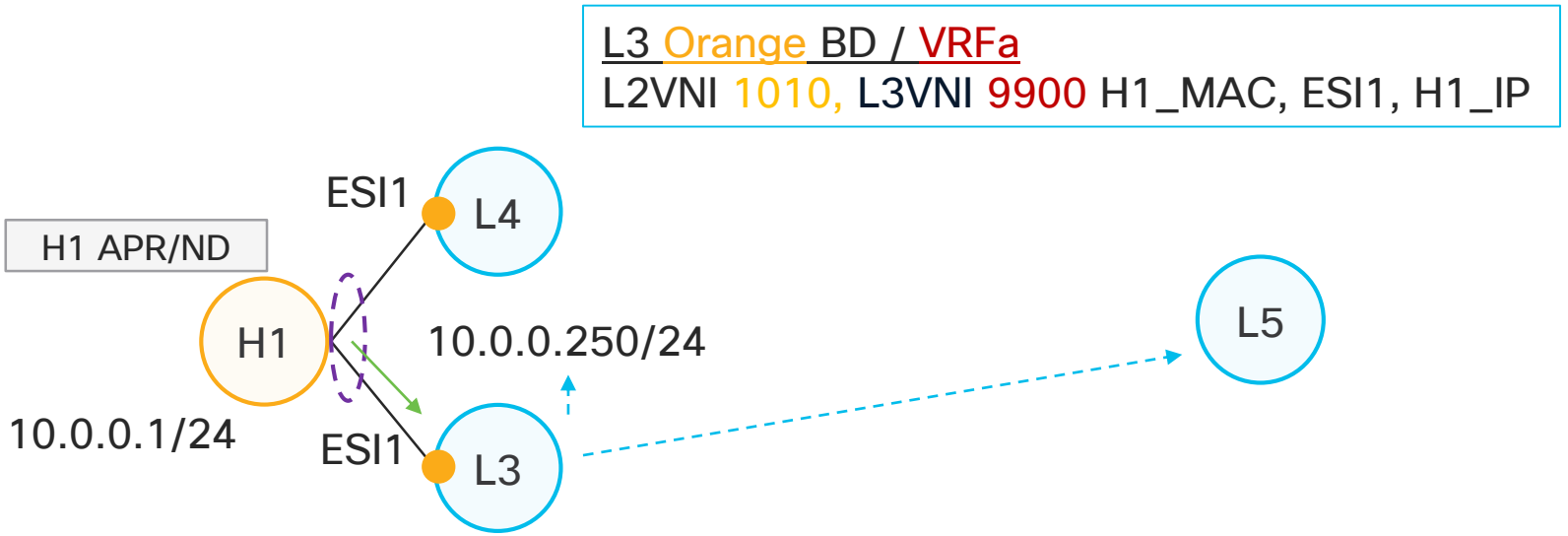
L2VPN: IP VXLAN / EVPN Control Plane – Counters

Remote R5 SONiC CLI

```
counterpoll tunnel enable
counterpoll tunnel interval 2000

show vxlan counters
       IFACE    RX_PKTS    RX_BYTES    RX_PPS    TX_PKTS    TX_BYTES    TX_PPS
------------  ---------  ----------  --------  ---------  ----------  --------
EVPN_3.3.3.3       2377      335580    0.00/s          4         570    0.00/s
EVPN_4.4.4.4       2362      333104    0.00/s         10        1104    0.00/s
       VXLAN          0           0    0.00/s          0           0    0.00/s

counterpoll show
Type                        Interval (in ms)    Status
--------------------------  ------------------  --------
QUEUE_STAT                  default (10000)     enable
PORT_STAT                   default (1000)      enable
PORT_BUFFER_DROP            default (60000)     enable
RIF_STAT                    default (1000)      enable
QUEUE_WATERMARK_STAT        default (60000)     enable
PG_WATERMARK_STAT           default (60000)     enable
PG_DROP_STAT                default (10000)     enable
BUFFER_POOL_WATERMARK_STAT  default (60000)     enable
ACL                         10000               disable
TUNNEL_STAT                 2000                enable
```

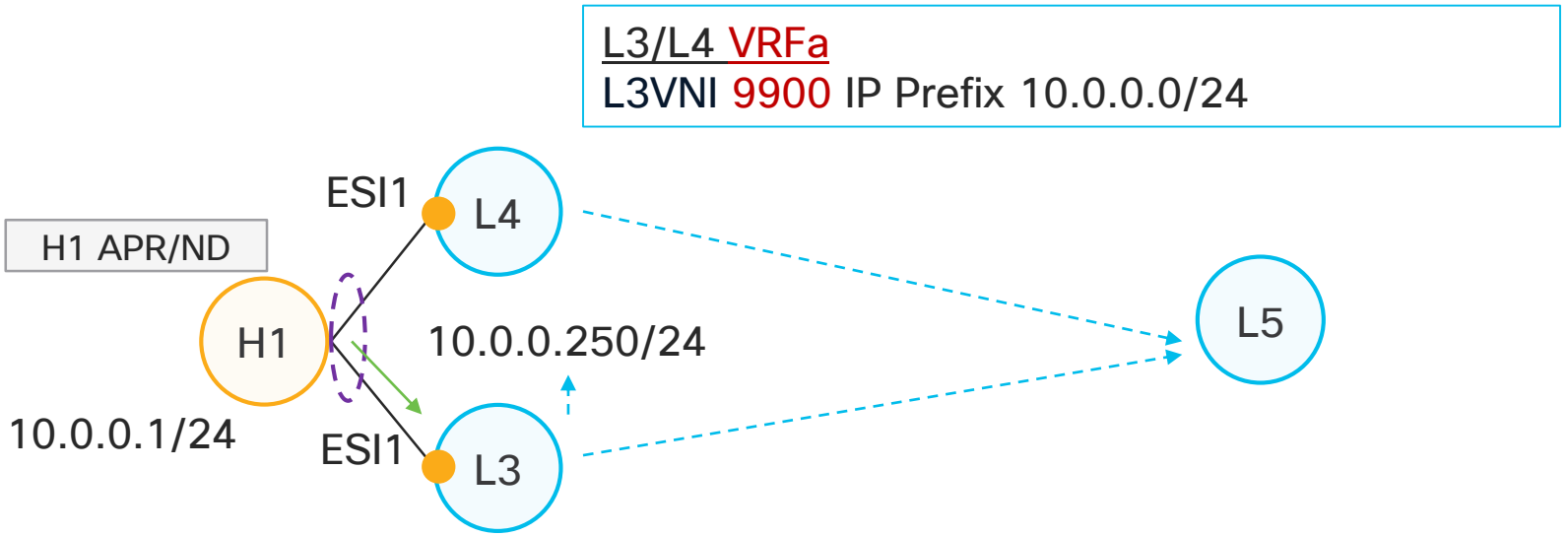
All-Active EVPN with VXLAN Data Plane L3VPN Distributed Anycast GW

EVPN Distributed Anycast Gateway
Route Type 2 (MAC+IP)

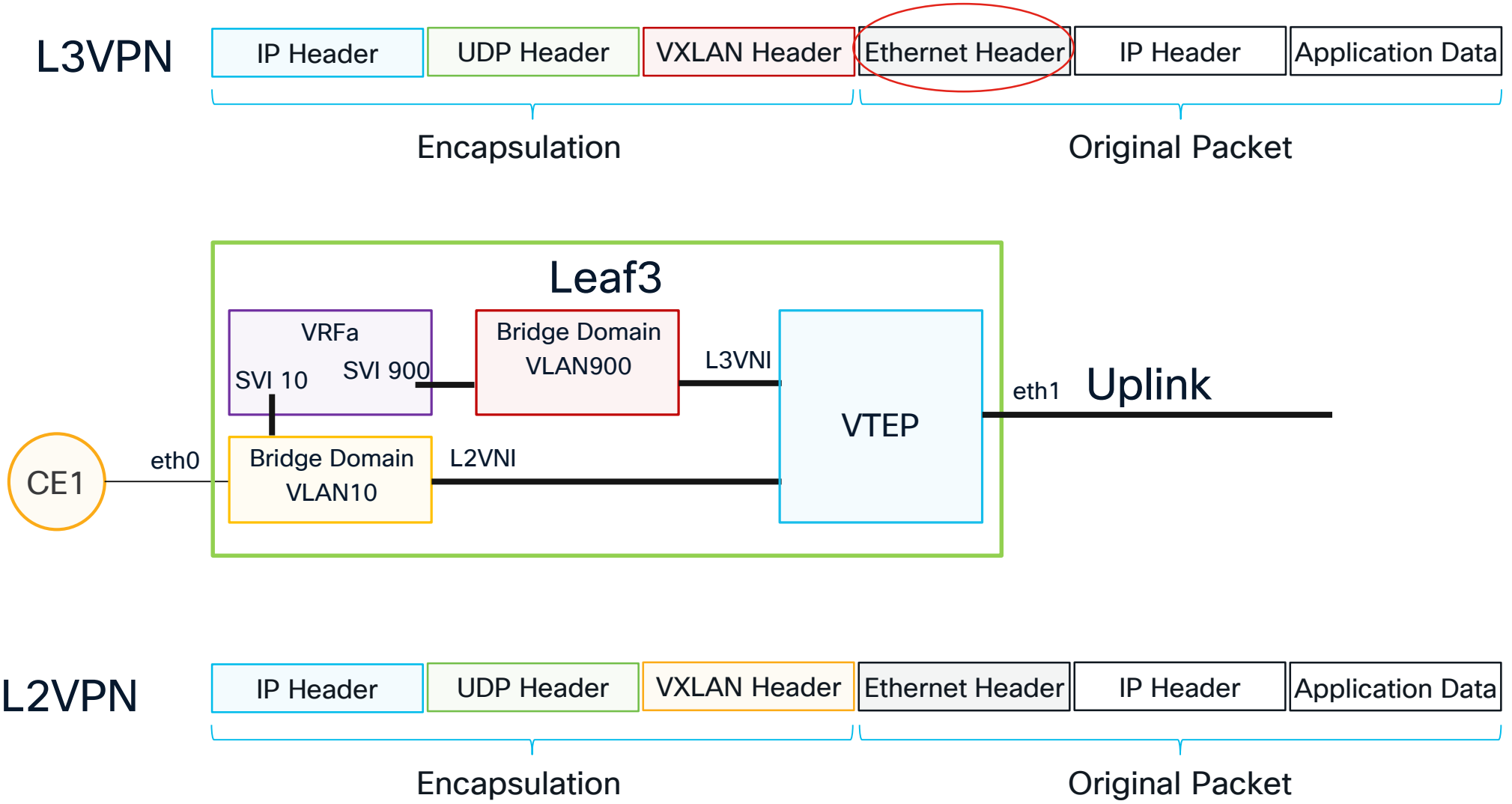


L3 learns H1 MAC + IP via ARP/ND – MAC is learned via Data plane
L3 advertises RT2 H1 MAC + IP:
- L4 synchronize adjacency table (based on ESI)
- L5 programs H1_IP (hostroute) into FIB for optimal forwarding

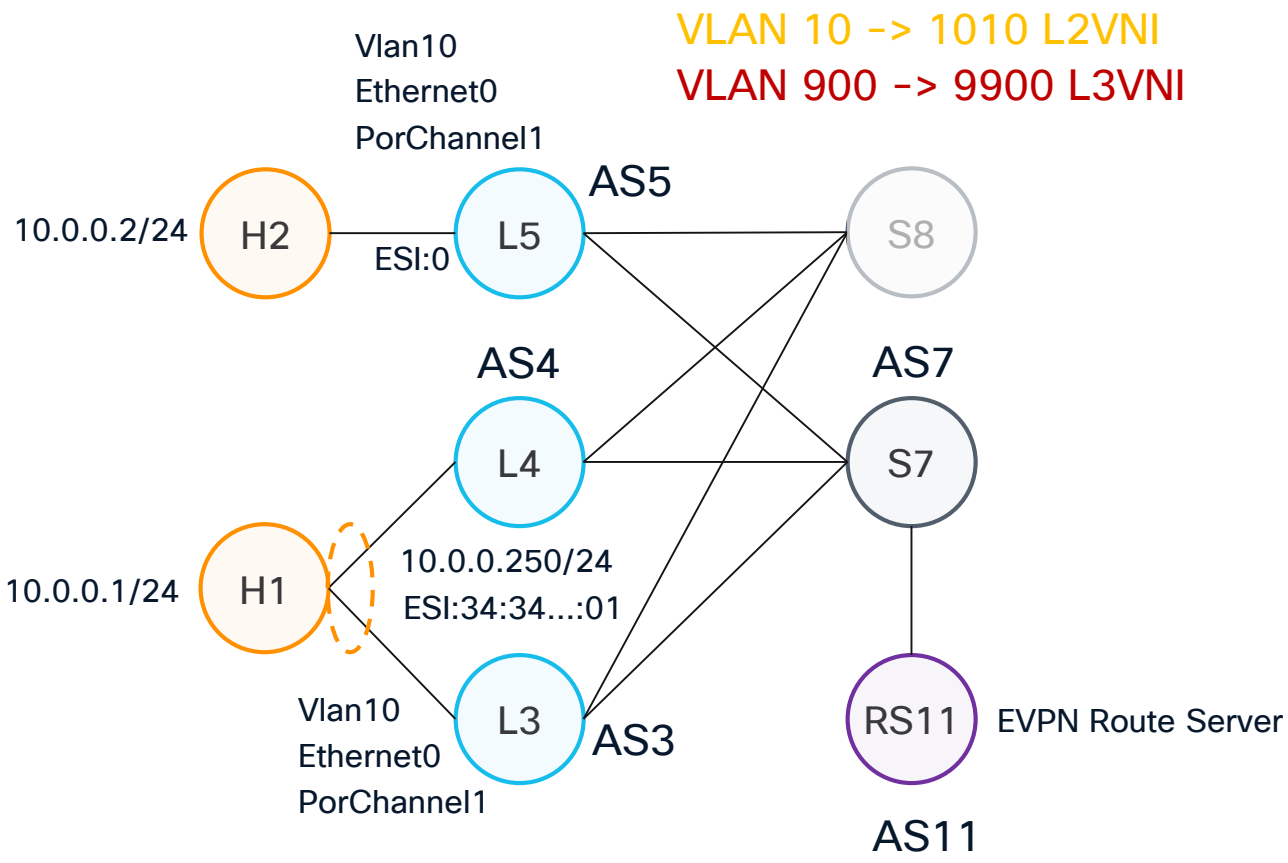
EVPN Distributed Anycast Gateway
Route Type 5 IP Prefix



VXLAN Data Plane Processing – not only in SONiC



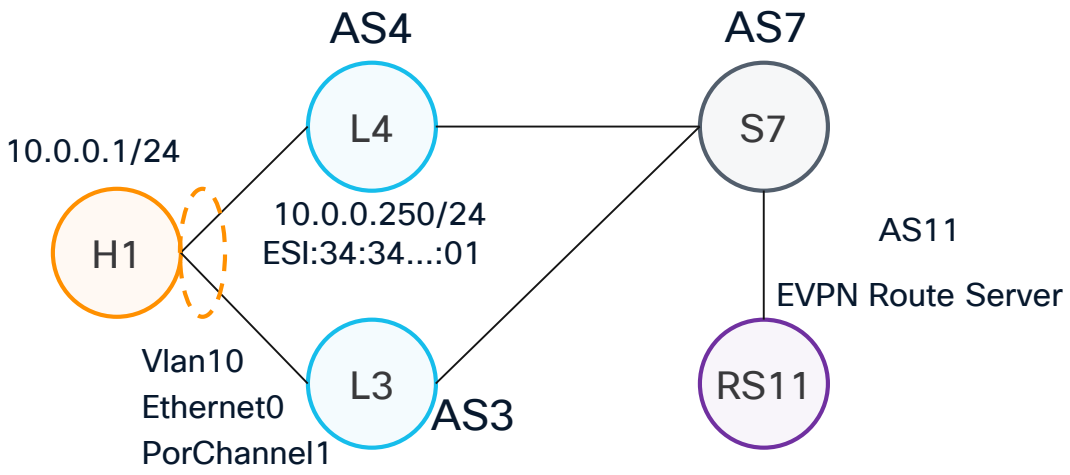
IP VXLAN with EVPN All-Active Multihoming - Testbed



L3VPN: IP VXLAN / EVPN Control Plane – Configuration

L3 SONiC CLI

```
config vlan add 900
config vrf add Vrfa
config interface vrf bind Vlan10 Vrfa
config interface vrf bind Vlan900 Vrfa

config vxlan map add VXLAN 900 9900
config vrf add_vrf_vni_map Vrfa 9900

config interface ip add Vlan10 10.0.0.250/24
config static-anycast-gateway mac_address add 0a:aa:aa:aa:aa:aa
config vlan static-anycast-gateway enable 10
```



L3 FRR vtysh

```
router bgp 3
……..
 address-family l2vpn evpn
  neighbor evpn activate
  advertise ipv4 unicast
  advertise-all-vni
  vni 1010
   route-target import 1010:1010
   route-target export 1010:1010
  exit-vni
  no use-es-l3nhg
 exit-address-family
exit

vrf Vrfa
 vni 9900
exit-vrf
```

```
router bgp 3 vrf Vrfa
 bgp router-id 3.3.3.3
 bgp bestpath as-path multipath-relax
 !
 address-family ipv4 unicast
  redistribute connected
  rt vpn both 9900:9900
 exit-address-family
 !
 address-family l2vpn evpn
  no use-es-l3nhg
  advertise ipv4 unicast
  route-target import 9900:9900
  route-target export 9900:9900
 exit-address-family
Exit
```

L3VPN: IP VXLAN / EVPN Control Plane – Validation

Remote R5 SONiC CLI

```
show vxlan tunnel
vxlan tunnel name    source ip    destination ip    tunnel map name    tunnel map mapping(vni -> vlan)
-------------------  -----------  ----------------  -----------------  --------------------------------
VXLAN                5.5.5.5                        map_1010_Vlan10    1010 -> Vlan10
                                                    map_9900_Vlan900   9900 -> Vlan900
```

```
ip route show vrf Vrfa
10.0.0.0/24 dev Vlan10 proto kernel scope link src 10.0.0.250
10.0.0.1 proto bgp metric 20
             nexthop via 3.3.3.3 dev Vlan900 weight 1 onlink
             nexthop via 4.4.4.4 dev Vlan900 weight 1 onlink

/sbin/bridge fdb | grep VXLAN-900
78:89:49:59:10:00 dev VXLAN-900 vlan 900 extern_learn master Bridge
78:90:3a:af:d8:00 dev VXLAN-900 vlan 900 extern_learn master Bridge
78:90:3a:af:d8:00 dev VXLAN-900 dst 3.3.3.3 self extern_learn
78:89:49:59:10:00 dev VXLAN-900 dst 4.4.4.4 self extern_learn
```

L3VPN: IP VXLAN / EVPN Control Plane – Validation

Remote R5 SONiC CLI

show vxlan vlanvnimap

```
+---------+-------+
| VLAN    |   VNI |
+=========+=======+
| Vlan10  |  1010 |
+---------+-------+
| Vlan900 |  9900 |
+---------+-------+
```

show vxlan remotevtep

```
+---------+---------+-------------------+--------------+
| SIP     | DIP     | Creation Source   | OperStatus   |
+=========+=========+===================+==============+
| 5.5.5.5 | 3.3.3.3 | EVPN              | oper_up      |
+---------+---------+-------------------+--------------+
| 5.5.5.5 | 4.4.4.4 | EVPN              | oper_up      |
+---------+---------+-------------------+--------------+
```

show vxlan remotemac all

```
+---------+-------------------+----------------+-------+---------+
| VLAN    | MAC               | RemoteTunnel   |   VNI | Type    |
+=========+===================+================+=======+=========+
| Vlan10  | 78:b4:2c:55:d0:00 | 3.3.3.3        |  1010 | dynamic |
|         |                   | 4.4.4.4        |       |         |
+---------+-------------------+----------------+-------+---------+
| Vlan900 | 78:89:49:59:10:00 | 4.4.4.4        |  9900 | dynamic |
+---------+-------------------+----------------+-------+---------+
| Vlan900 | 78:90:3a:af:d8:00 | 3.3.3.3        |  9900 | dynamic |
+---------+-------------------+----------------+-------+---------+
```

L3VPN: IP VXLAN / EVPN Control Plane – Validation

Remote R5 FRR vtysh

```
show bgp l2vpn evpn
....
   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 3.3.3.3:2
 *> [1]:[0]:[00:34:34:34:34:34:34:34:34:01]:[32]:[0.0.0.0]:[0]
                    3.3.3.3                                0 3 i
                    RT:1010:1010 ET:8
 *> [2]:[0]:[48]:[78:b4:2c:55:d0:00]
                    3.3.3.3                                0 3 i
                    ESI:00:34:34:34:34:34:34:34:34:01
                    RT:1010:1010 ET:8
 *> [2]:[0]:[48]:[78:b4:2c:55:d0:00]:[32]:[10.0.0.1]
                    3.3.3.3                                0 3 i
                    ESI:00:34:34:34:34:34:34:34:34:01
                    RT:1010:1010 RT:9900:9900 ET:8 Rmac:78:90:3a:af:d8:00
 *> [3]:[0]:[32]:[3.3.3.3]
                    3.3.3.3                                0 3 i
                    RT:1010:1010 ET:8
Route Distinguisher: 3.3.3.3:3
 *> [1]:[4294967295]:[00:34:34:34:34:34:34:34:34:01]:[32]:[0.0.0.0]:[0]
                    3.3.3.3                                0 3 i
                    RT:1010:1010 ET:8 ESI-label-Rt:AA
 *> [4]:[00:34:34:34:34:34:34:34:34:01]:[32]:[3.3.3.3]
                    3.3.3.3                                0 3 i
                    ET:8 ES-Import-Rt:34:34:34:34:34:34 DF: (alg: 2, pref: 32767)
```

```
Route Distinguisher: 3.3.3.3:4
 *> [5]:[0]:[24]:[10.0.0.0]
                    3.3.3.3                  0             0 3 ?
                    RT:9900:9900 ET:8 Rmac:78:90:3a:af:d8:00
```

L3VPN: IP VXLAN / EVPN Control Plane – Validation

Remote R5 FRR vtysh

```
show bgp l2vpn evpn
....
Route Distinguisher: 5.5.5.5:4
 *> [2]:[0]:[48]:[78:2f:db:6c:90:00]
                    5.5.5.5                            32768 i
                    ET:8 RT:1010:1010
 *> [2]:[0]:[48]:[78:2f:db:6c:90:00]:[32]:[10.0.0.2]
                    5.5.5.5                            32768 i
                    ET:8 RT:1010:1010 RT:9900:9900 Rmac:78:f1:b6:4f:7c:00
 *> [3]:[0]:[32]:[5.5.5.5]
                    5.5.5.5                            32768 i
                    ET:8 RT:1010:1010
Route Distinguisher: 5.5.5.5:5
 *> [5]:[0]:[24]:[10.0.0.0]
                    5.5.5.5                  0         32768 ?
                    ET:8 RT:9900:9900 Rmac:78:f1:b6:4f:7c:00
```

```
show evpn es
Type: B bypass, L local, R remote, N non-DF
ESI                             Type ES-IF                VTEPs
00:34:34:34:34:34:34:34:34:01  R    -                    3.3.3.3,4.4.4.4


show evpn rmac vni all

VNI 9900 #RMACs 2

RMAC              Remote VTEP
78:90:3a:af:d8:00 3.3.3.3
78:89:49:59:10:00 4.4.4.4
```

```
cat /proc/net/arp
IP address       HW type     Flags       HW address            Mask     Device
10.0.0.2         0x1         0x2         78:2f:db:6c:90:00     *        Vlan10
192.168.122.1    0x1         0x2         7e:df:61:31:ca:51     *        eth0
192.168.123.1    0x1         0x2         ae:08:c4:72:f4:8c     *        eth4
5.7.1.7          0x1         0x2         78:66:a8:cf:80:00     *        Ethernet80
```

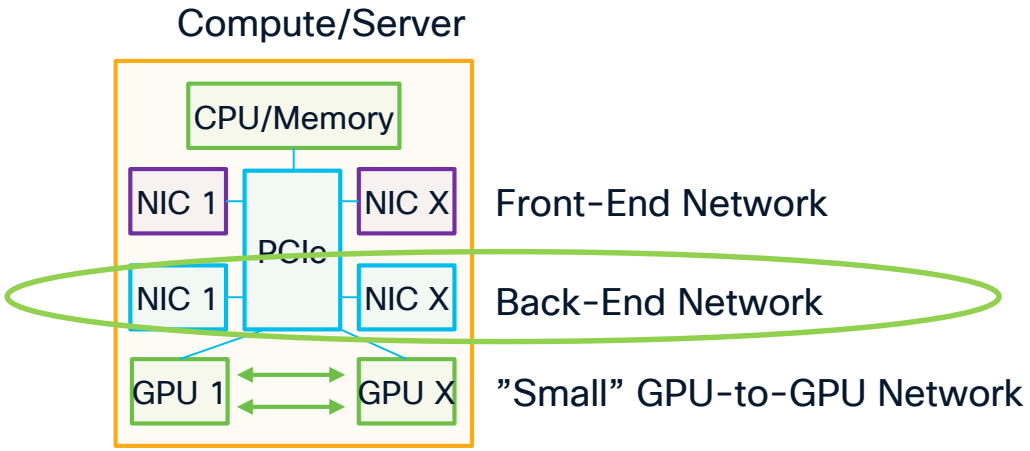
AI Usecases

Back-End Network

- AI Training or Inference

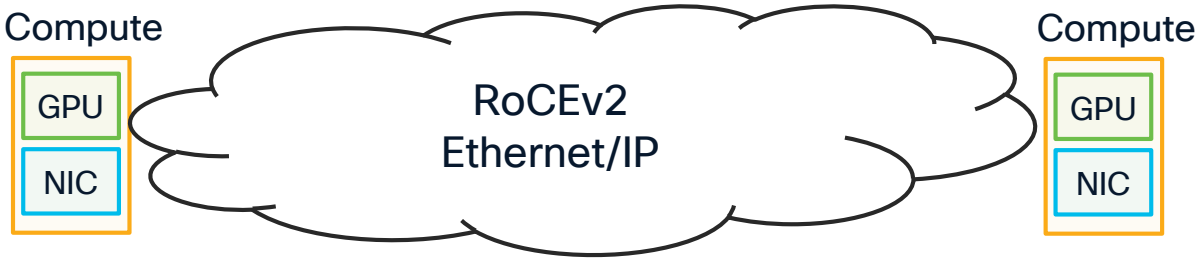
- GPU to GPU communication / data exchange over Remote Direct Memory Access (RDMA)
  - RDMA provides direct memory access without involving Operating system / CPU
  - NIC RDMA support required

- Requirements:
  - High Bandwidth
    - Effective load-balancing
  - Low Latency
    - Inference for real-time experience
    - Training is also latency sensitive
  - Loss Less
    - Restart / Re-transmission is extremely expensive
  - Multi-Tenancy



Ethernet / IP Based – Back-End Network

- RoCEv2 – Ethernet + IP/UDP header to carry RDMA data
- Network transports IP/UDP packets

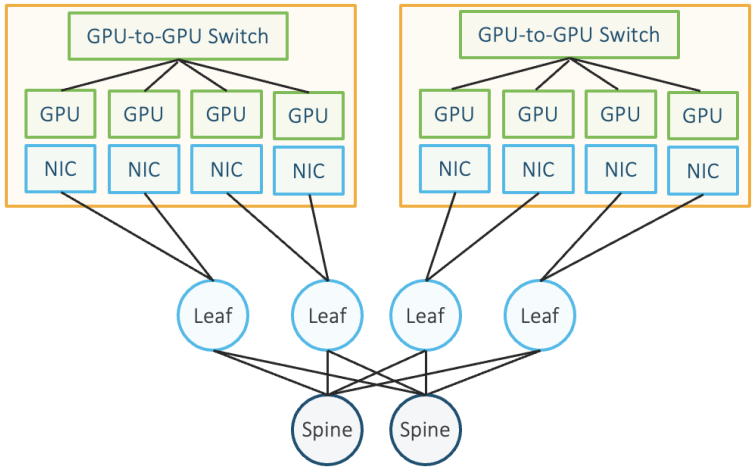



```
root@r5:/etc/sonic# config platform cisco udf-hash set -h
Usage: config platform cisco udf-hash set [OPTIONS]

  Enable UDF hash

Options:
  -n []           Namespace name or all, valid for multi-
asic only
  -o INTEGER      Offset in UDP
  -b INTEGER      Number of bits for hash mask
  -qpid           Use QPID
  -s              Save the configuration
  -?, -h, --help  Show this message and exit.
```

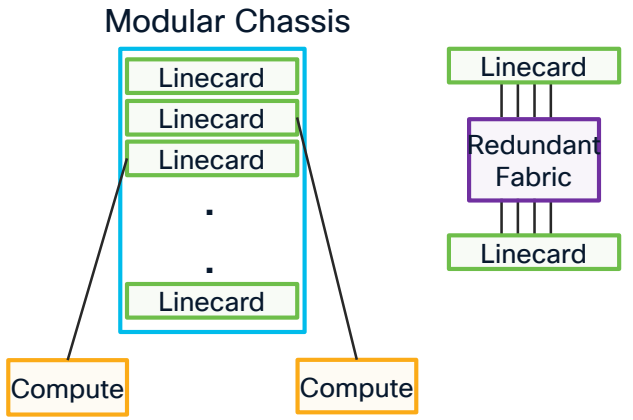
Adding destination queue pair field (part of InfiniBand header) into hashing algorithm

Back-End Network Common Design Options

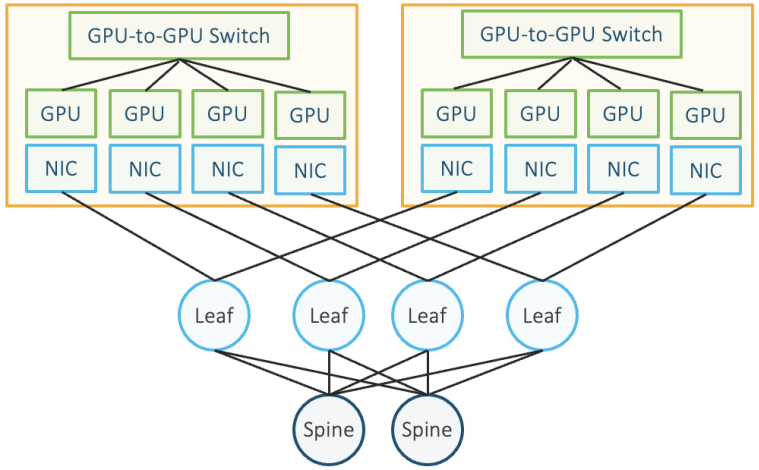
CLOS Design



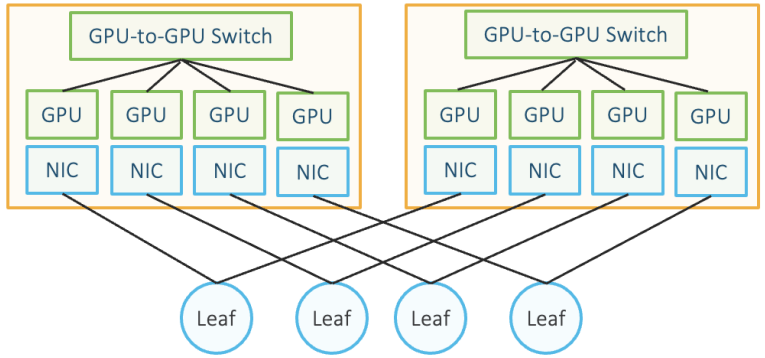
Scale UP Design with Modular Chassis



Rail-Optimized Design



Rail-Only Design



Rail-Optimized / Rail-Only Whitepaper: https://arxiv.org/pdf/2307.12169

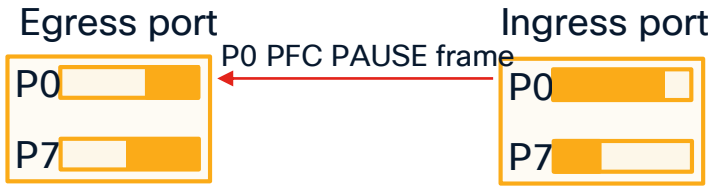
Back-End Network – Forwarding Optimization

- PFC/ECN
- FlowLet Load Balancing
- Packet Spraying
- Static Balancing / Pinning

PFC / ECN

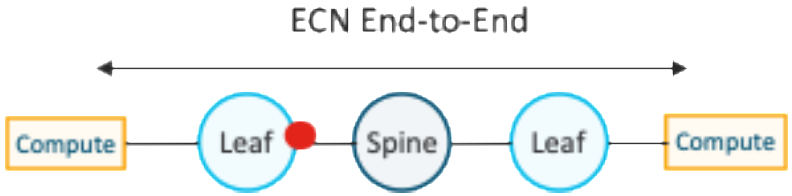
- RoCEv2 leverages existing Ethernet/IP enhancements
  - Ethernet – Priority Flow Control (PFC)
    - IEEE – 802.1Qbb (2008)
  - IP – Early Congestion Notification (ECN)
    - RFC3168 (2001)

PFC



- Per priority ethernet pause
- Lossless ethernet
- Requires buffers
- Cascading PAUSE

ECN



- ECN uses last 2bits in IP header Traffic Class field
- ECN capable devices modifies ECN bits to signal congestion

PFC/ECN – Best together
- ECN prevents compute to send more data into overload network
- PFC prevents packet drops inside overload network
- PFC/ECN Tunning is required

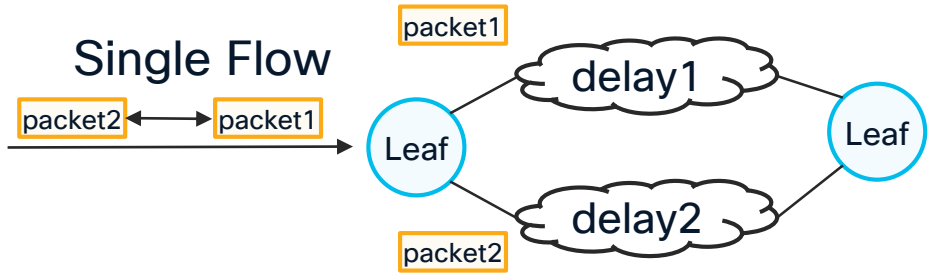
SONiC PFC / ECN .JSON example

```
"MAP_PFC_PRIORITY_TO_QUEUE": {
        "AZURE": {
            "0": "0",
            "1": "1",
            "2": "2",
            "3": "3",
            "4": "4",
            "5": "5",
            "6": "6",
            "7": "7"
        }
    },
    "TC_TO_QUEUE_MAP": {
        "AZURE": {
            "0": "0",
            "1": "1",
            "2": "2",
            "3": "3",
            "4": "4",
            "5": "5",
            "6": "6",
            "7": "7"
        }
    },
    "DSCP_TO_TC_MAP": {
        "AZURE": {
            "0" : "1",
            "1" : "1",
            "2" : "1",
            "3" : "3",
snip
            "63": "1"
        }
    },
```

```
},
   "WRED_PROFILE": {
       "AZURE_LOSSLESS" : {
           "ecn"                    : "ecn_green_yellow",
           "green_drop_probability" : "5",
           "green_min_threshold"    : "2097152",
           "green_max_threshold"    : "10485760",
           "yellow_min_threshold"   : "0",
           "yellow_drop_probability": "0",
           "yellow_max_threshold"   : "6144000"
       }
   },
},
   "SCHEDULER": {
       "scheduler.0": {
           "type"  : "DWRR",
           "weight": "14"
       },
       "scheduler.1": {
           "type"  : "DWRR",
           "weight": "15"
       }
   },
   "PORT_QOS_MAP": {

       "Ethernet0": {
           "dscp_to_tc_map"  : "AZURE",
           "tc_to_queue_map" : "AZURE",
           "pfc_enable"      : "3,4",
           "pfcwd_sw_enable" : "3,4",
           "tc_to_pg_map"    : "AZURE",
           "pfc_to_queue_map": "AZURE"
       },
```

FlowLet Load Balancing

- FlowLet load-balancing
  - Flowlet idea came earlier to improve load-balancing of bursty TCP
  - https://groups.csail.mit.edu/netmit/wordpress/wp-content/themes/netmit/papers/texcp-hotnets04.pdf




```
cisco@sonic:~$ sudo config ars add -h
Usage: config ars add [OPTIONS] NAME

  Add object in ARS.

Options:
  --mode [flowlet-quality|per-packet-quality]
                                  This enumeration type defines the method
                                  used to determine how path reassignment is
                                  done.
  --idle-time TEXT                minimum idle-time in microseconds between
                                  two packets of the same flow for flowlet
                                  identification
```

Packet1 and Packet2 are from same flow
Leaf can send packet2 via different link/path than packet1 if:

- Time between packet1 and packet2 is higher than |delay1 – delay2|

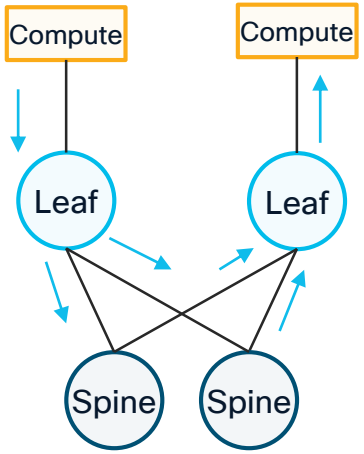
- **Tunning to prevent packet re-ordering is required**

Packet Spraying

- Optimal network link utilization
- Packet Re-ordering on egress Compute NIC

```
cisco@sonic:~$ sudo config ars add -h
Usage: config ars add [OPTIONS] NAME

  Add object in ARS.

Options:
  --mode [flowlet-quality|per-packet-quality]
                                  This enumeration type defines the method
                                  used to determine how path reassignment is
                                  done.
  --idle-time TEXT                minimum idle-time in microseconds between
                                  two packets of the same flow for flowlet
                                  identification
```

Packet Spray

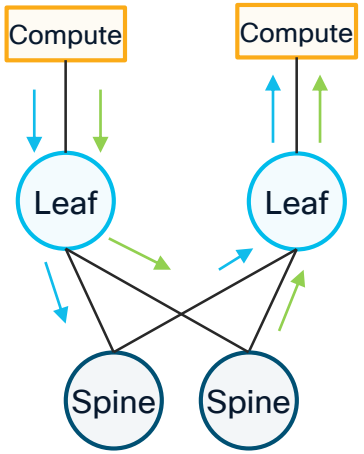


Packet Re-Ordering

- **NIC has to support packet re-ordering**
- **Difficult to trace re-ordering in network**

Static Pinning

- Traffic Engineering to select outgoing interface / Spine and end-to-end path
- SRv6 Traffic Engineering provides flexible solution

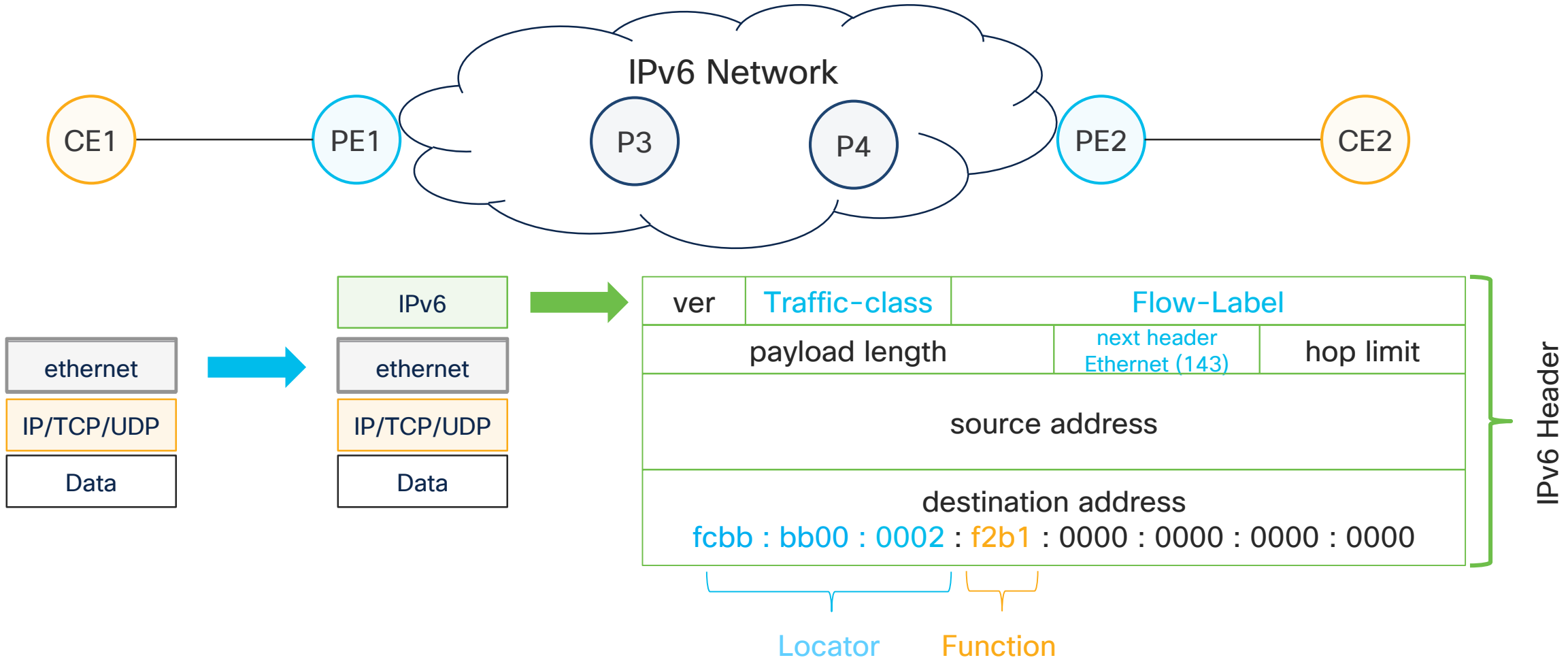


FRR vtysh – SRv6 example

```
segment-routing
 srv6
  locators
   locator MAIN
    prefix fcbb:bbbb:1::/48
    format usid-f3216
  !
!

router# show segment-routing srv6 locator
Locator:
Name                 ID      Prefix                   Status
-------------------- ------- ------------------------ -------
MAIN                      1 fcbb:bbbb:1::/48         Up
```

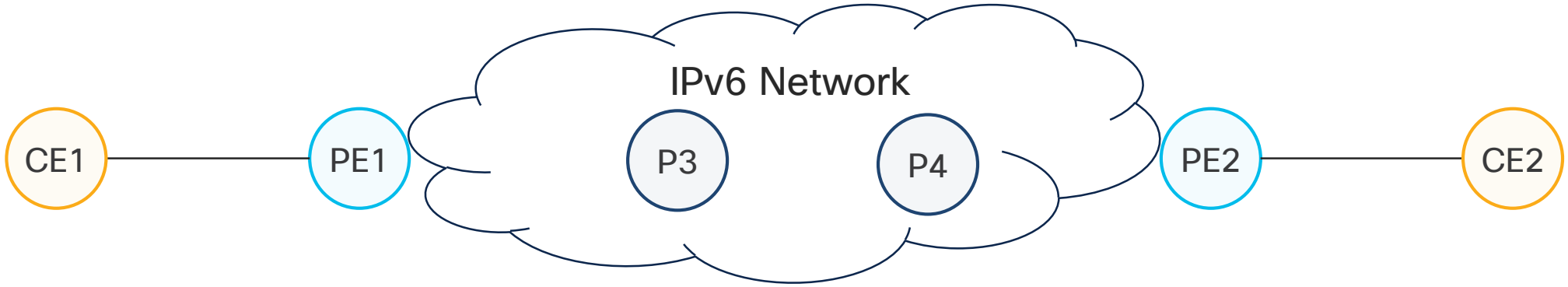
- **External Controller may be required for more complex traffic engineering**

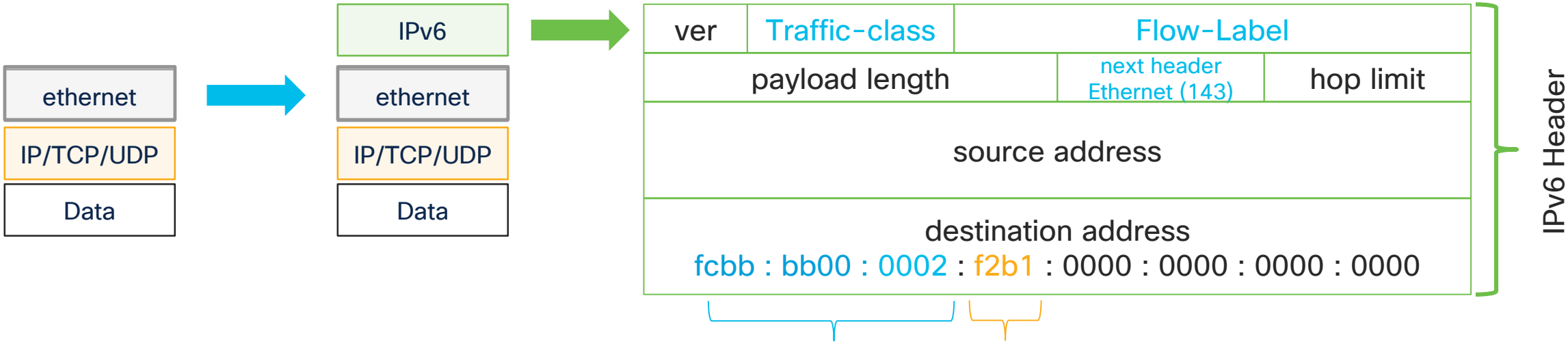
SRv6 Illustration



SRv6 SID => LOC:FUNCT:ARG

SRv6 Illustration (2)





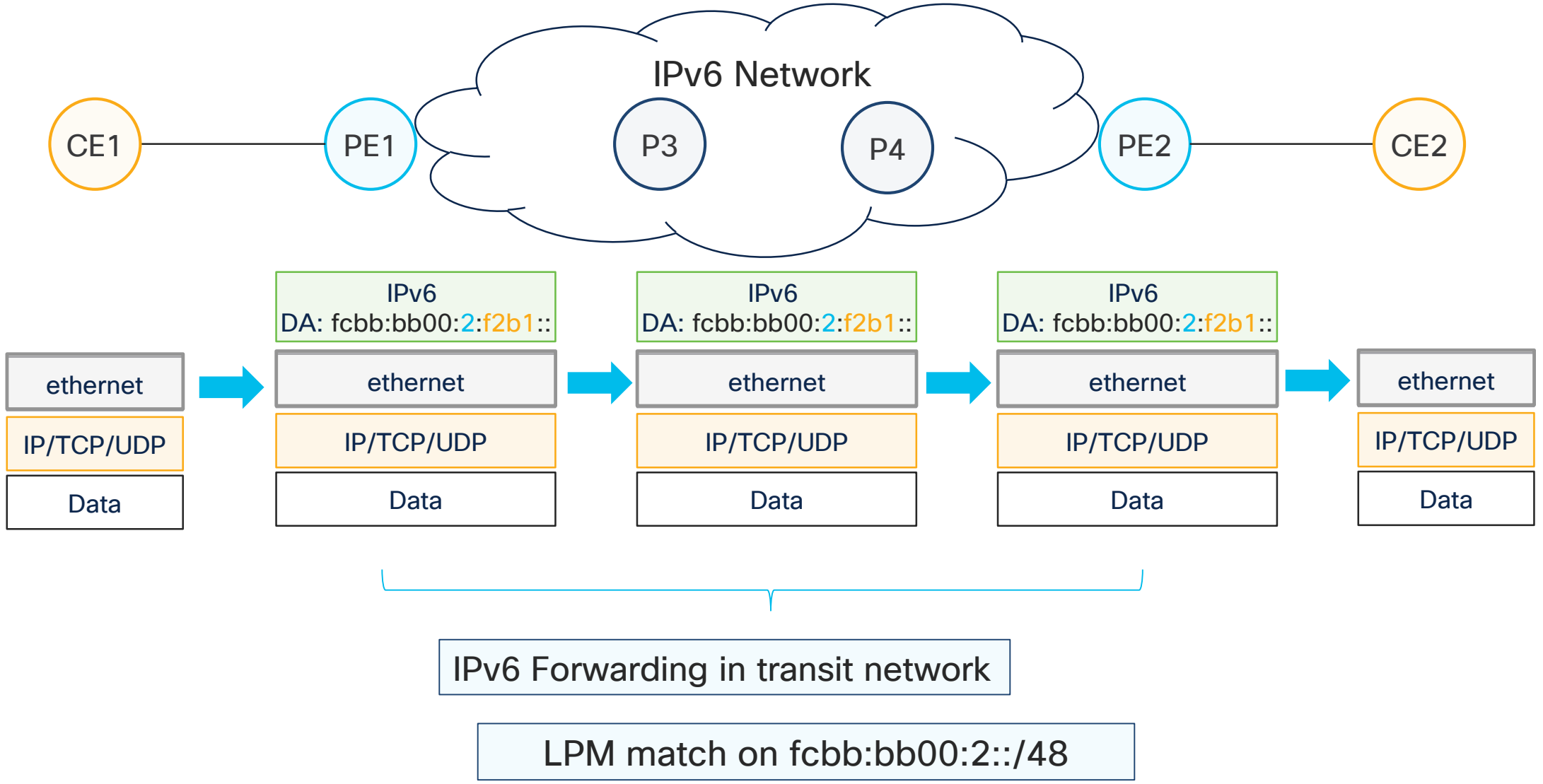
Instructions:

Goto PE2

PE2: (e.g.)

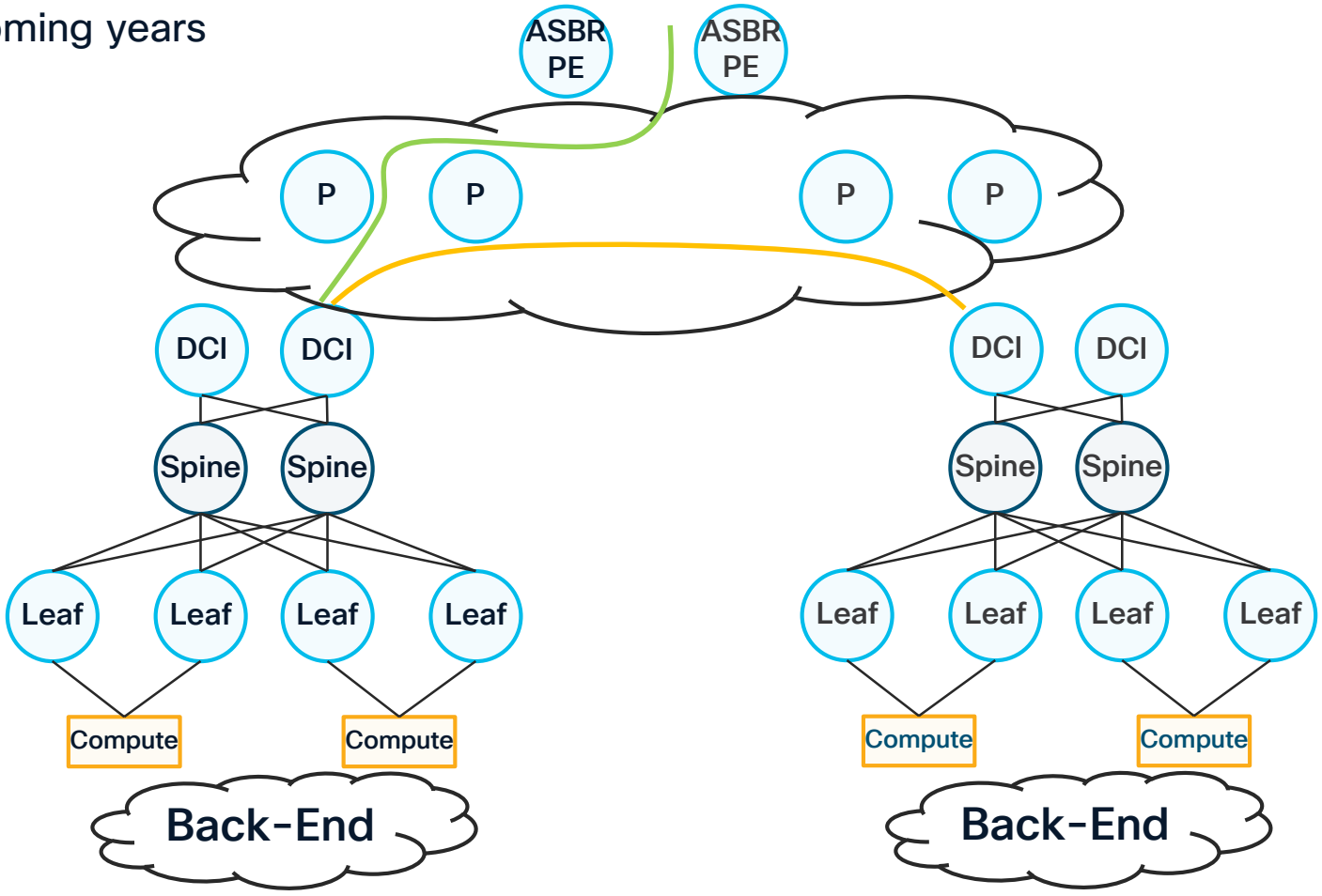
1. Decapsulate IPv6 header
2. Perform MAC lookup in Orange Bridge Domain

SRv6 Illustration (3)



Backbone / Core

- Traffic Engineering to provide optimal path for data collection and distribution
  - SRv6 provides optimal and flexible solution
- Expected backbone traffic grow in the coming years
- SONiC is not ready yet



Summary

Summary: SONiC or Vendor OS?

- SONiC offers flexibility / customization but is not for everyone
  - Be ready for extra effort to deploy and operate (no free lunch)
  - Be sure you have right expertise
- Cisco validated SONiC release should be preferred option
- Stick to solution blueprint / validated design
- **Vendor OS such as IOS XR provides rich feature set with high scale**
  - Different type of Data Plane: MPLS, IP (VXLAN, SRv6)
  - Complex Feature set: Traffic Engineering, Fast Re-Reroute, ...
  - Deployed in largest and most complex networks such as SP Access/Core, SP DC, Peering, DCI, ...

Complete your session evaluations

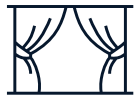
Complete a minimum of 4 session surveys and the Overall Event Survey to be entered in a drawing to win 1 of 5 full conference passes to Cisco Live 2026.

Earn 100 points per survey completed and compete on the Cisco Live Challenge leaderboard.

Level up and earn exclusive prizes!

Complete your surveys in the Cisco Live mobile app.

Continue your education



Visit the Cisco Showcase for related demos

Book your one-on-one Meet the Engineer meeting

Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs

Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand

Thank you

cisco Live !


CISCO