



OPEN SOURCE NETWORKING DAYS

How Does SONiC Operate On a Programmable Switch

Howard Hsu <howard_hsu@edge-core.com>

Edgecore Networks

@OSN_DAYS_TAIWAN

Beginning: Traditional networking switch



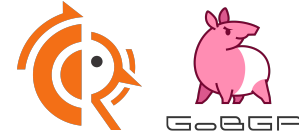
- Proprietary hardware components
- Proprietary NOS and functions
- Own ecosystem

Go from scratch : Bare-metal Switch



Applications

DHCP, DNS, HTTP, MySQL,



OS/Software



Hardware



Server



Switch

ONIE: The Open Network Install Environment

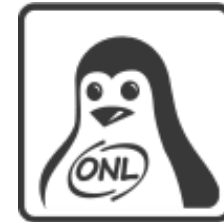


```
GNU GRUB  version 2.02
```

```
+-----+
|*ONIE: Install OS
| ONIE: Rescue
| ONIE: Uninstall OS
| ONIE: Update ONIE
| ONIE: Embed ONIE
+-----+
```

```
Use the ^ and v keys to select which entry is highlighted.
Press enter to boot the selected OS, `e' to edit the commands
before booting or `c' for a command-line.
```

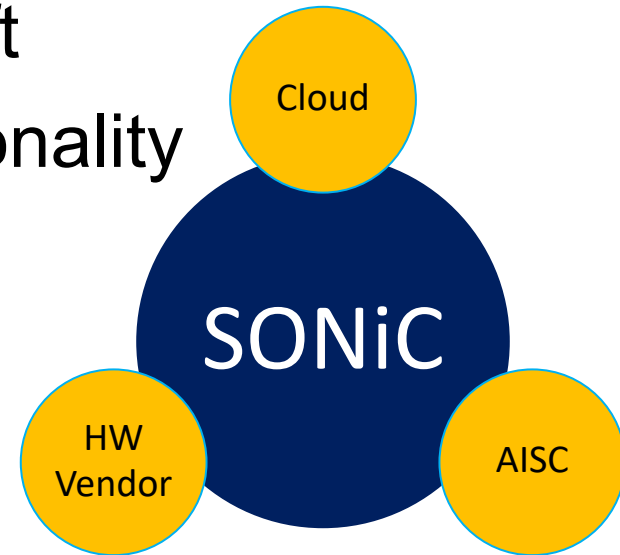
Build on top: Whitebox Network OS



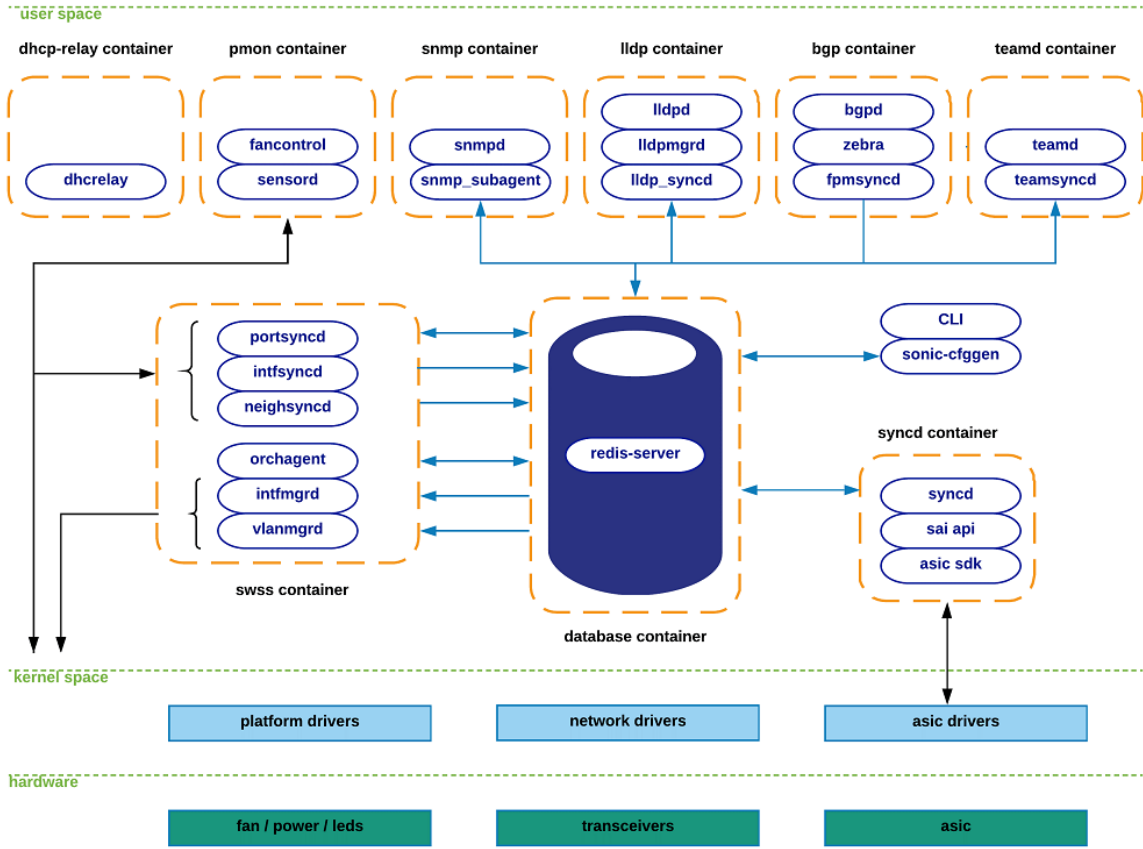
Open Network Linux

SONiC: Software for Open Networking in the Cloud

- Open source in 2016 by Microsoft
- Offers full-suite of network functionality
 - BGP
 - LLDP
 - SNMP
- Containerized network function
- Switch Abstraction Interface API (SAI)



SONiC Architecture



- DHCP relay
- PMON
- SNMP
- LLDP
- BGP
- TeamD
- Database
- SwSS
- SyncD

Platform Monitor container



- **Sensord**
 - Log sensor readings from hardware components
 - Alert when an alarm is signaled
- **Fancontrol**
 - Collect temperature stats from platform sensors
 - React on that by increasing/decreasing fan speed

Database container



- **APPL_DB**
 - Stores the state generated by all application containers
 - EX: routes, next-hops, neighbors, interfaces
- **CONFIG_DB**
 - Stores the configuration created by SONiC applications
 - EX: port configurations, interfaces, vlans
- **STATE_DB**
 - stores all the state resolve cross-modular dependencies between subsystems
- **ASIC_DB**
 - Stores the necessary state of ASIC configuration and operation

Switch State Services container



- Orchagent
 - The most critical component in the SwSS subsystem
 - As consumer for state coming from APPL_DB
 - As producer for state being pushed into ASIC_DB
- intfMgrd
 - Configure interfaces in the linux kernel
- vlanMgrd
 - Configure vlan-interfaces in the linux kernel

Syncd container



- Syncd
 - Links with the ASIC SDK library provided by the hardware-vendor
 - Injects state to the ASIC by invoking the interfaces provided for such effect
- SAI API
 - Defines the API to provide a vendor-independent forwarding elements control
- ASIC SDK
 - A SAI-friendly implementation
 - Typically provided in the form of a dynamic-linked-library which hooks up to a driving process

SAI: Switch Abstraction Interface



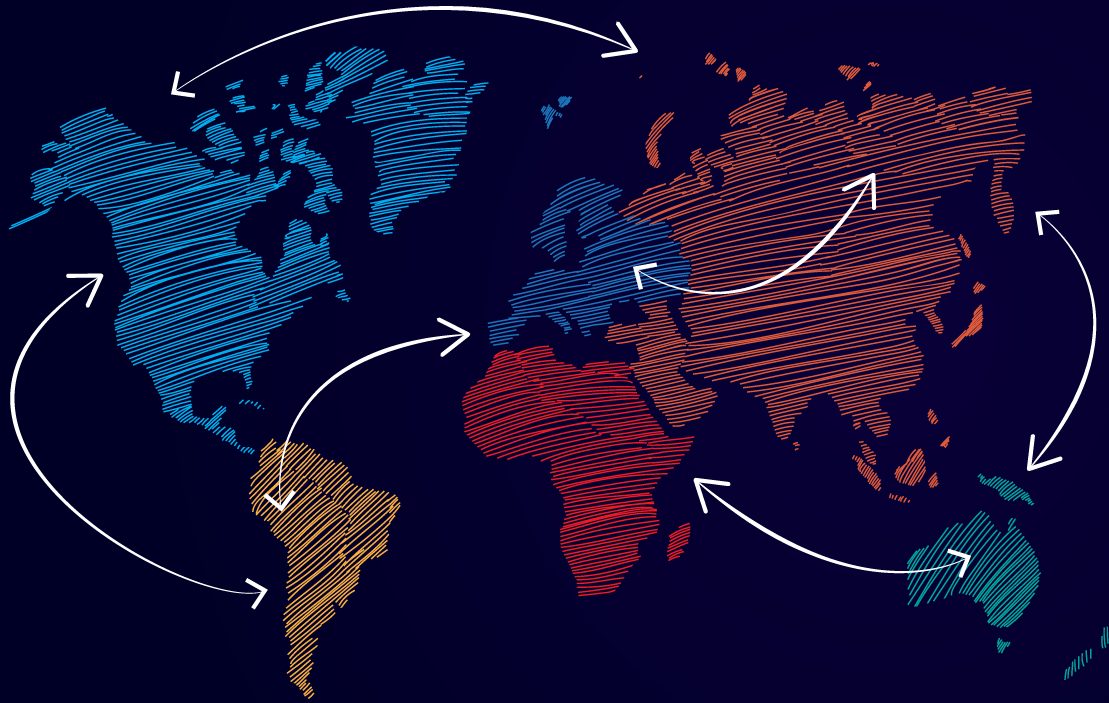
<https://github.com/opencomputeproject/SAI/inc>

- Open source and accepted by OCP
- Standardized C API to program ASICs

```
316  /**
317   * @brief Router entry methods table retrieved with sai_api_query()
318   */
319  typedef struct _sai_route_api_t
320  {
321      sai_create_route_entry_fn          create_route_entry;
322      sai_remove_route_entry_fn         remove_route_entry;
323      sai_set_route_entry_attribute_fn  set_route_entry_attribute;
324      sai_get_route_entry_attribute_fn  get_route_entry_attribute;
325
326      sai_bulk_create_route_entry_fn    create_route_entries;
327      sai_bulk_remove_route_entry_fn    remove_route_entries;
328      sai_bulk_set_route_entry_attribute_fn  set_route_entries_attribute;
329      sai_bulk_get_route_entry_attribute_fn  get_route_entries_attribute;
330
331  } sai_route_api_t;
```

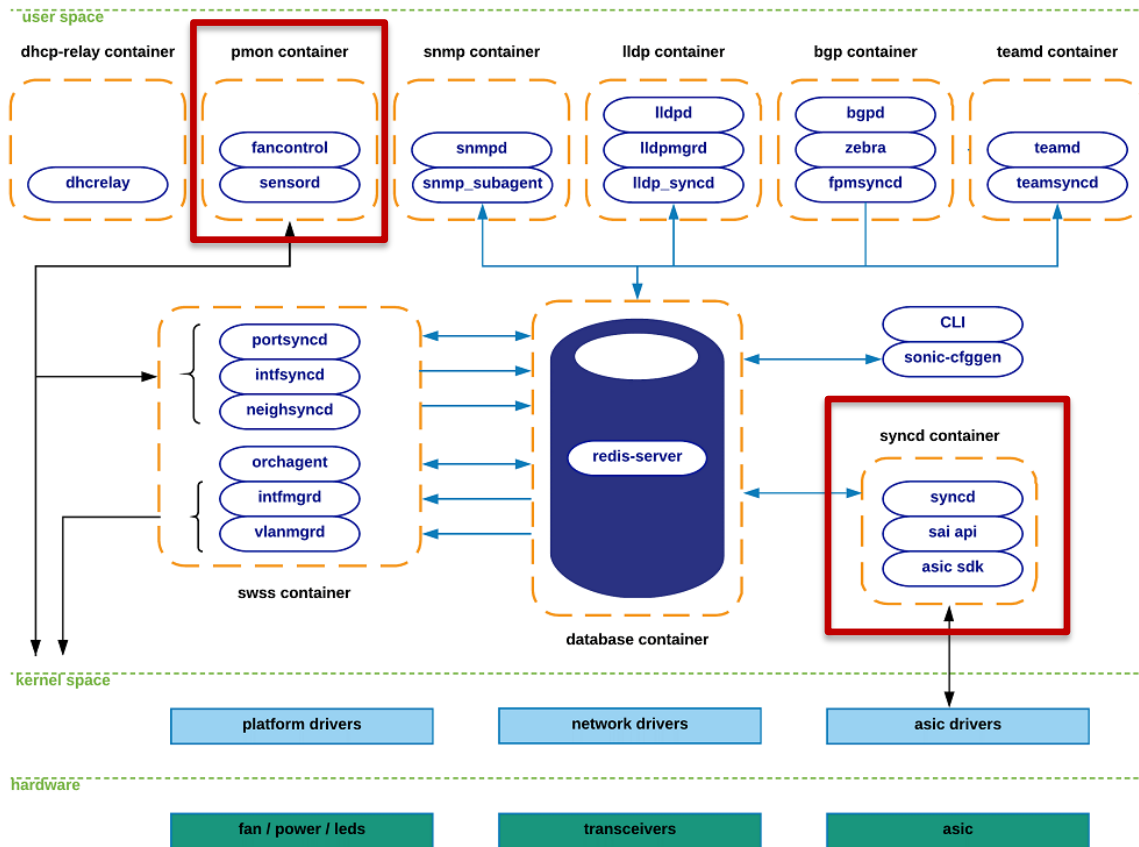
sai_route_api

| | |
|----------------------|------------------------|
| 📄 sai.h | 📄 sainexthopgroup.h |
| 📄 saiacl.h | 📄 saioobject.h |
| 📄 saibfd.h | 📄 saipolicer.h |
| 📄 saibrIDGE.h | 📄 saiport.h |
| 📄 saibuffer.h | 📄 saiqosmap.h |
| 📄 saicounter.h | 📄 saiqueue.h |
| 📄 saidebugcounter.h | 📄 sairoute.h |
| 📄 saidtel.h | 📄 sairouterinterface.h |
| 📄 saifdb.h | 📄 sairpfgroup.h |
| 📄 saihash.h | 📄 saisamplepacket.h |
| 📄 saihostif.h | 📄 saischeduler.h |
| 📄 saipmc.h | 📄 saischedulergroup.h |
| 📄 saipmcgroup.h | 📄 saisegmentroute.h |
| 📄 saisolationgroup.h | 📄 saistatus.h |
| 📄 sail2mc.h | 📄 saistp.h |
| 📄 sail2mcgroup.h | 📄 saiswitch.h |
| 📄 sailag.h | 📄 saisystemport.h |
| 📄 saimacsec.h | 📄 saitam.h |
| 📄 saimcastfdb.h | 📄 saitunnel.h |
| 📄 saimirror.h | 📄 saitypes.h |
| 📄 saimpls.h | 📄 saiudf.h |
| 📄 sainat.h | 📄 saivirtualrouter.h |
| 📄 saineighbor.h | 📄 saivlan.h |
| 📄 sainexthop.h | 📄 saiwred.h |



So, how does
it work on
programmable
ASIC?

Targets

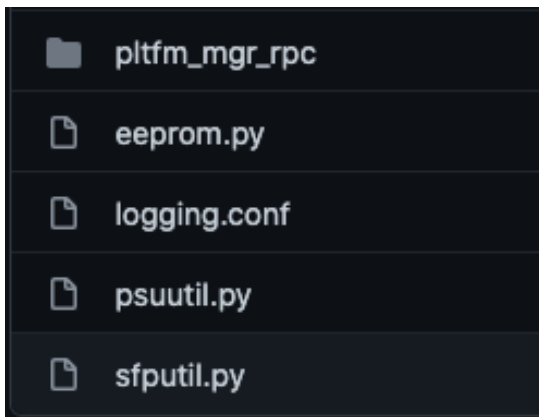


- DHCP relay
- PMON
- SNMP
- LLDP
- BGP
- TeamD
- Database
- SwSS
- SyncD

Hardware elements for PMON



- <https://github.com/Azure/sonic-buildimage>
- Locate at device/{vendor}/{model}/



plugins

```
1  {
2      "skip_pcied": false,
3      "skip_fancontrol": true,
4      "skip_thermalctld": true,
5      "skip_ledd": true,
6      "skip_xcvrd": false,
7      "skip_psud": false,
8      "skip_syseepromd": false
9  }
```

pmon_daemon_control.json

Sdk & Platform for Syncd



<https://github.com/Azure/sonic-buildimage/tree/master/platform/barefoot>

```
1 BFN_PLATFORM = bfnplatform_20201023_deb9.deb
2 $(BFN_PLATFORM)_URL = "https://github.com/barefootnetworks/sonic-release-pkgs/raw/dev/$(BFN_PLATFORM)"
3
4 SONIC_ONLINE_DEBS += $(BFN_PLATFORM)
5 $(BFN_SAI_DEV)_DEPENDS += $(BFN_PLATFORM)
```

bfm-platform.mk

```
1 BFN_SAI = bfnsdk_20201023_deb9.deb
2 $(BFN_SAI)_URL = "https://github.com/barefootnetworks/sonic-release-pkgs/raw/dev/$(BFN_SAI)"
3
4 $(BFN_SAI)_DEPENDS += $(LIBNL_GENL3_DEV)
5 $(eval $(call add_conflict_package,$(BFN_SAI),$(LIBSAIVS_DEV)))
6 $(BFN_SAI)_RDEPENDS += $(LIBNL_GENL3)
7
8 SONIC_ONLINE_DEBS += $(BFN_SAI)
9 $(BFN_SAI_DEV)_DEPENDS += $(BFN_SAI)
```

bfm-sai.mk















syncd container



Sonic-release-pkgs



<https://github.com/barefootnetworks/sonic-release-pkgs/tree/dev>

| | | | |
|--|---|---|---|
|  bfnplatform_20200701_deb9.deb | Added 20200701 packages with SAI v1.6.3 (#3) |  bfnsdk_20200701_deb9.deb | Added 20200701 packages with SAI v1.6.3 (#3) |
|  bfnplatform_20200710_deb9.deb | Added 20200710 packages with fixes (#4) |  bfnsdk_20200710_deb9.deb | Added 20200710 packages with fixes (#4) |
|  bfnplatform_20200721_deb9.deb | Added BFN SDK packages 20200721 with SAI v1.6.4 |  bfnsdk_20200721_deb9.deb | Added BFN SDK packages 20200721 with SAI v1.6.4 |
|  bfnplatform_20200731_sai_1.5.2_de... | Added BFN SDK packages 20200731 with SAI v1.5.2 |  bfnsdk_20200731_sai_1.5.2_deb9.deb | Added BFN SDK packages 20200731 with SAI v1.5.2 |
|  bfnplatform_20201009_deb9.deb | Added BFN SDK packages 20201009 with SAI v1.6.4 |  bfnsdk_20201009_deb9.deb | Added BFN SDK packages 20201009 with SAI v1.6.4 |
|  bfnplatform_20201023_deb9.deb | Added BFN SDK packages 20201023 with SAI v1.6.4 |  bfnsdk_20201023_deb9.deb | Added BFN SDK packages 20201023 with SAI v1.6.4 |
|  bfnplatform_20201023_sai_1.5.2_de... | Added SDK packages 20201023 with SAI v1.5.2 |  bfnsdk_20201023_sai_1.5.2_deb9.deb | Added SDK packages 20201023 with SAI v1.5.2 |

1. Modify bfn-platform.mk & bfn-sai.mk
2. \$ make target/docker-syncd-bfn.gz
3. Upload new container to SONiC

Upgrade SONiC syncd



1. `$ docker load < docker-syncd-bfn.gz`
2. Tag the image as latest
3. `$ sudo systemctl restart swss`

| IMAGE | NAMES |
|---|----------------|
| <code>docker-syncd-bfn:latest</code> | syncd |
| <code>docker-snmp:latest</code> | snmp |
| <code>docker-sonic-mgmt-framework:latest</code> | mgmt-framework |
| <code>docker-sonic-telemetry:latest</code> | telemetry |
| <code>docker-router-advertiser:latest</code> | radv |
| <code>docker-dhcp-relay:latest</code> | dhcp_relay |
| <code>docker-lldp:latest</code> | lldp |
| <code>docker-teamd:latest</code> | teamd |
| <code>docker-orchagent:latest</code> | swss |
| <code>docker-fpm-frr:latest</code> | bgp |
| <code>docker-platform-monitor:latest</code> | pmon |
| <code>docker-database:latest</code> | database |

`docker ps`

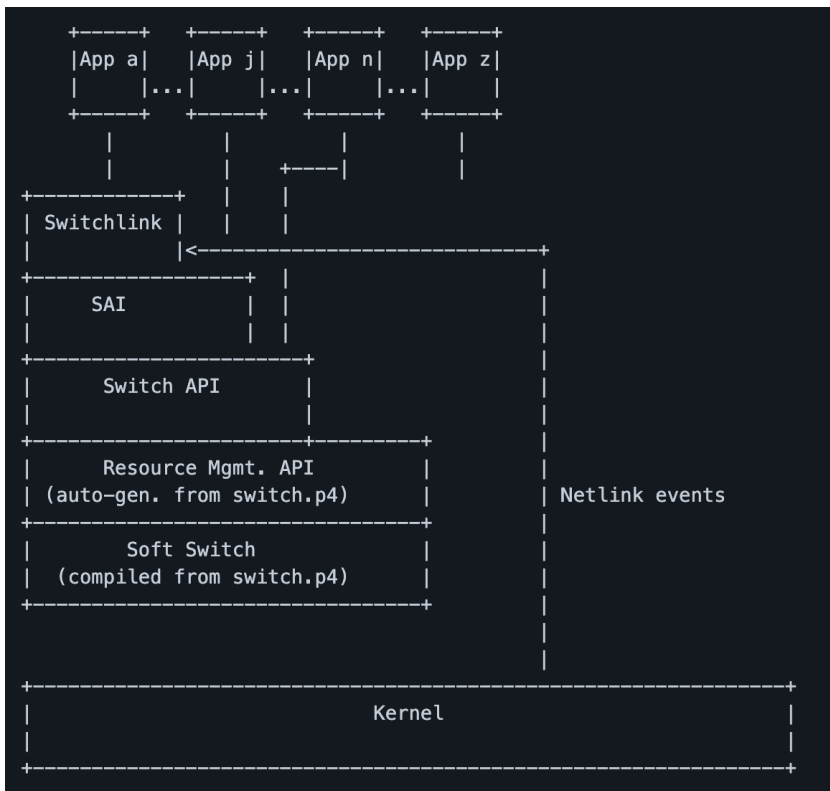
`docker images`

| REPOSITORY | TAG |
|--|--------------------------------|
| <code>docker-snmp</code> | latest |
| <code>docker-snmp</code> | master.0-dirty-20201112.110734 |
| <code>docker-teamd</code> | latest |
| <code>docker-teamd</code> | master.0-dirty-20201112.110734 |
| <code>docker-nat</code> | latest |
| <code>docker-nat</code> | master.0-dirty-20201112.110734 |
| <code>docker-router-advertiser</code> | latest |
| <code>docker-router-advertiser</code> | master.0-dirty-20201112.110734 |
| <code>docker-platform-monitor</code> | latest |
| <code>docker-platform-monitor</code> | master.0-dirty-20201112.110734 |
| <code>docker-database</code> | latest |
| <code>docker-database</code> | master.0-dirty-20201112.110734 |
| <code>docker-orchagent</code> | latest |
| <code>docker-orchagent</code> | master.0-dirty-20201112.110734 |
| <code>docker-lldp</code> | latest |
| <code>docker-lldp</code> | master.0-dirty-20201112.110734 |
| <code>docker-sonic-telemetry</code> | latest |
| <code>docker-sonic-telemetry</code> | master.0-dirty-20201112.110734 |
| <code>docker-dhcp-relay</code> | latest |
| <code>docker-dhcp-relay</code> | master.0-dirty-20201112.110734 |
| <code>docker-sonic-mgmt-framework</code> | latest |
| <code>docker-sonic-mgmt-framework</code> | master.0-dirty-20201112.110734 |
| <code>docker-fpm-frr</code> | latest |
| <code>docker-fpm-frr</code> | master.0-dirty-20201112.110734 |
| <code>docker-sflow</code> | latest |
| <code>docker-sflow</code> | master.0-dirty-20201112.110734 |
| <code>docker-syncd-bfn</code> | latest |
| <code>docker-syncd-bfn</code> | master.0-dirty-20201112.110734 |

Switch.p4



<https://github.com/p4lang/switch>



p4src

- acl.p4
- egress_filter.p4
- fabric.p4
- hashes.p4
- int_transit.p4
- ipv4.p4
- ipv6.p4
- l2.p4
- l3.p4
- meter.p4
- mirror.p4
- multicast.p4

- nat.p4
- nexthop.p4
- openflow.p4
- p4_files.am
- port.p4
- qos.p4
- rewrite.p4
- security.p4
- sflow.p4
- switch.p4
- switch_config.p4
- tunnel.p4

Pipeline selection from configuration



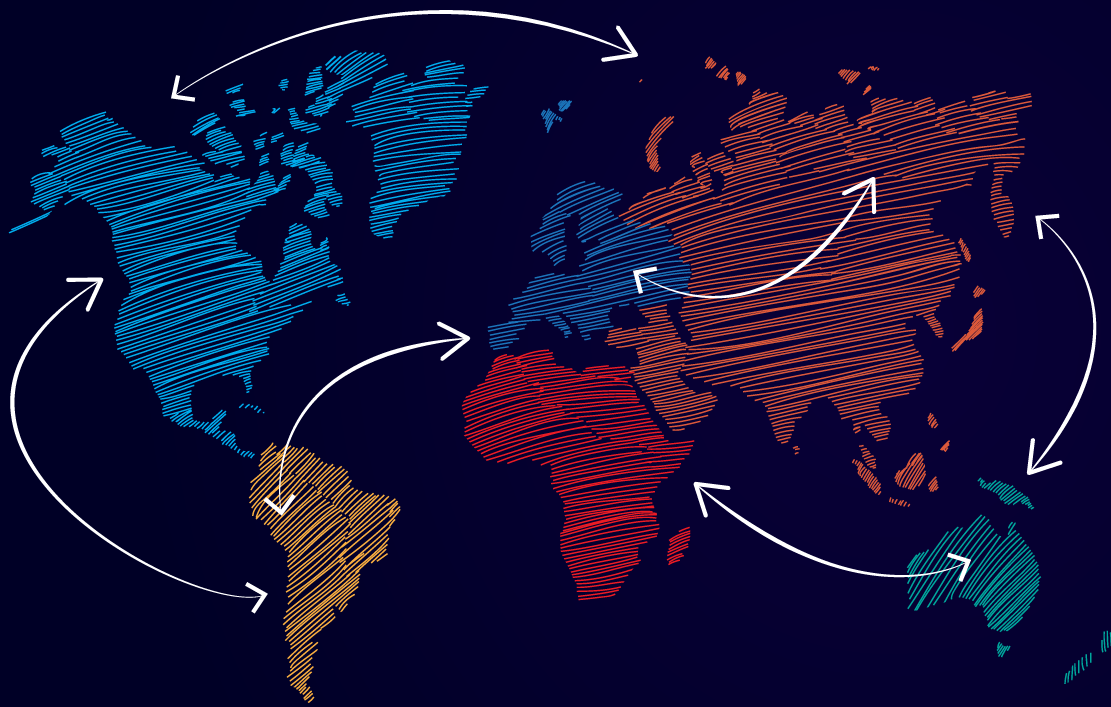
```
$ vim /etc/sonic/config_db.json
```

```
"DEVICE_METADATA": {  
  "localhost": {  
    "hostname": "sonic",  
    "p4_profile": "xxx_profile",  
    .....
```

```
$ sudo config reload -y
```

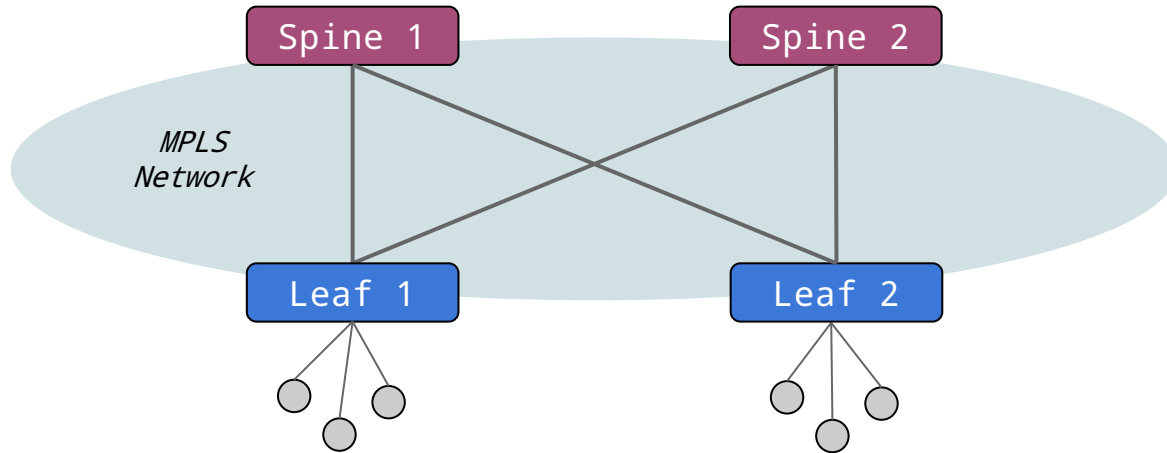
In syncd docker image
/opt/bfn

```
install -> install_x2_profile  
install_x1_profile  
install_x2_profile  
install_y1_profile
```



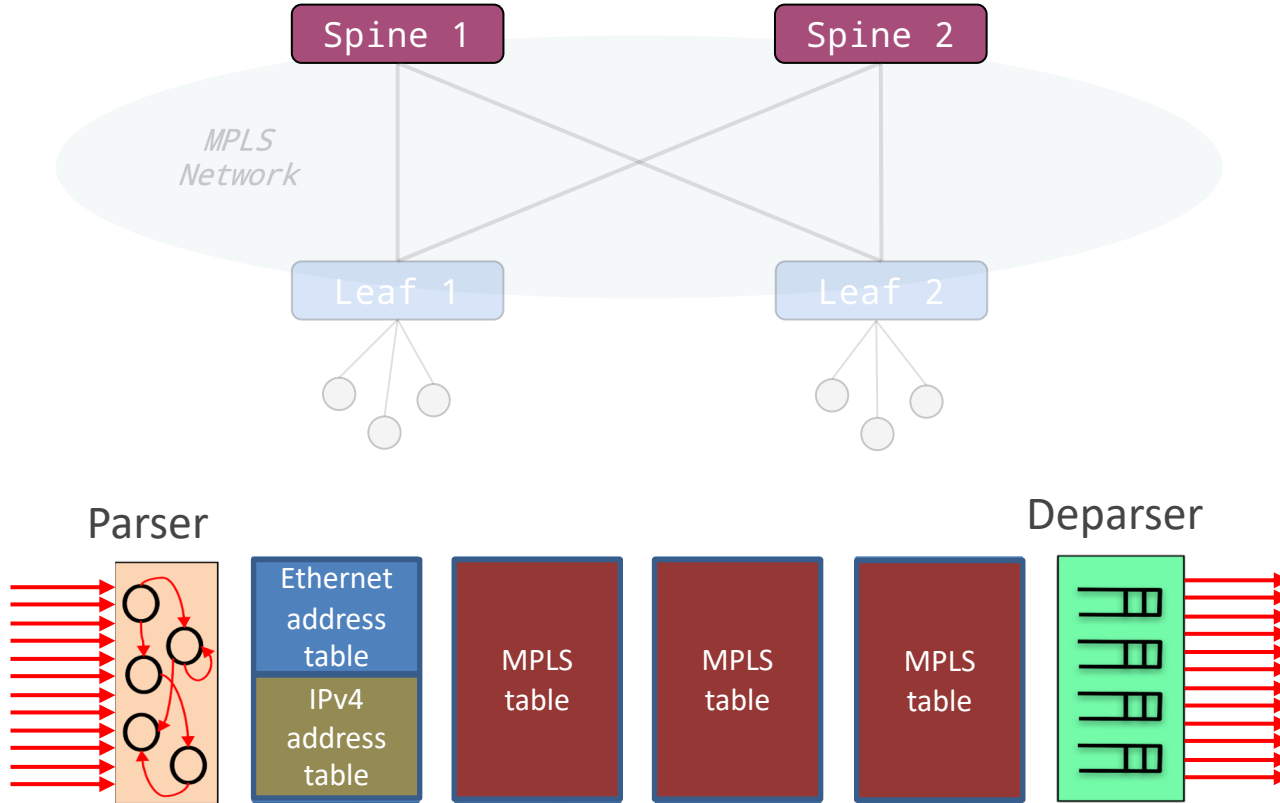
Next, how do we leverage with pipeline selection in SONiC?

Adjust pipeline table size for different purpose

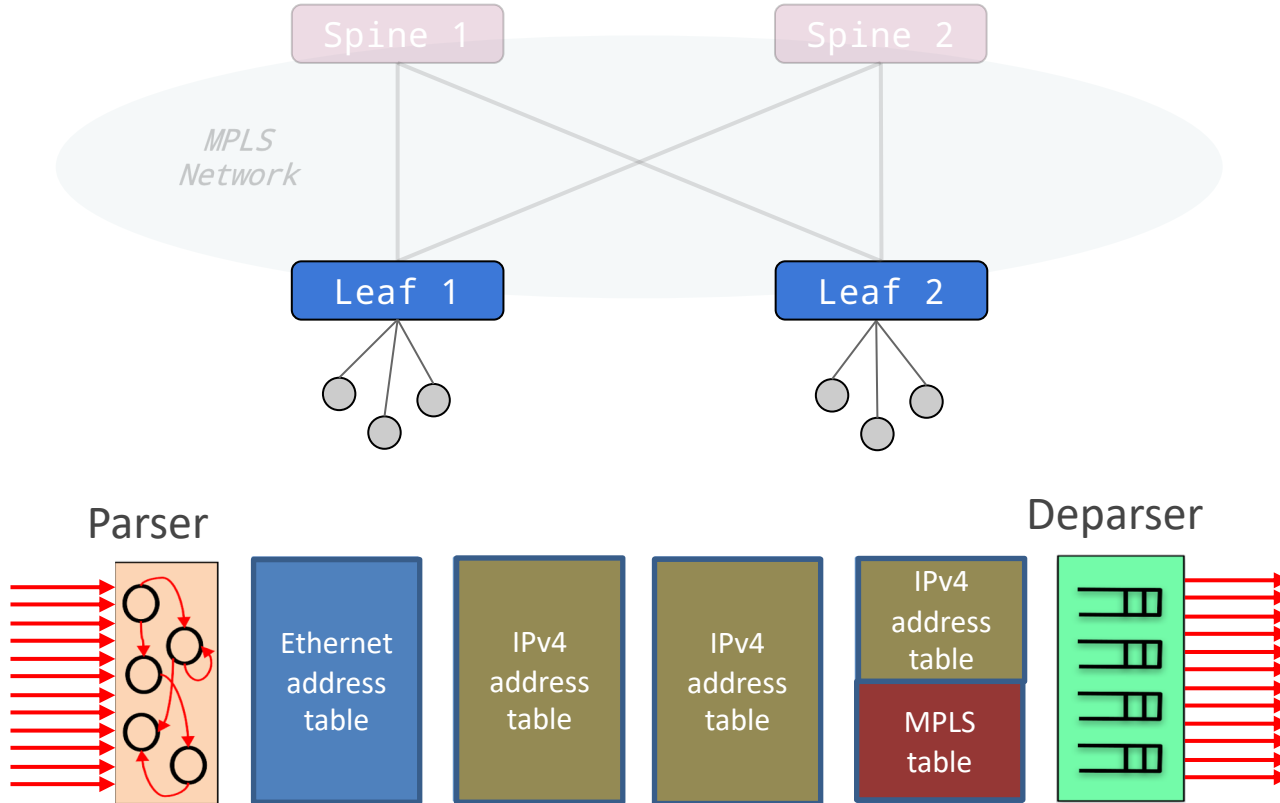


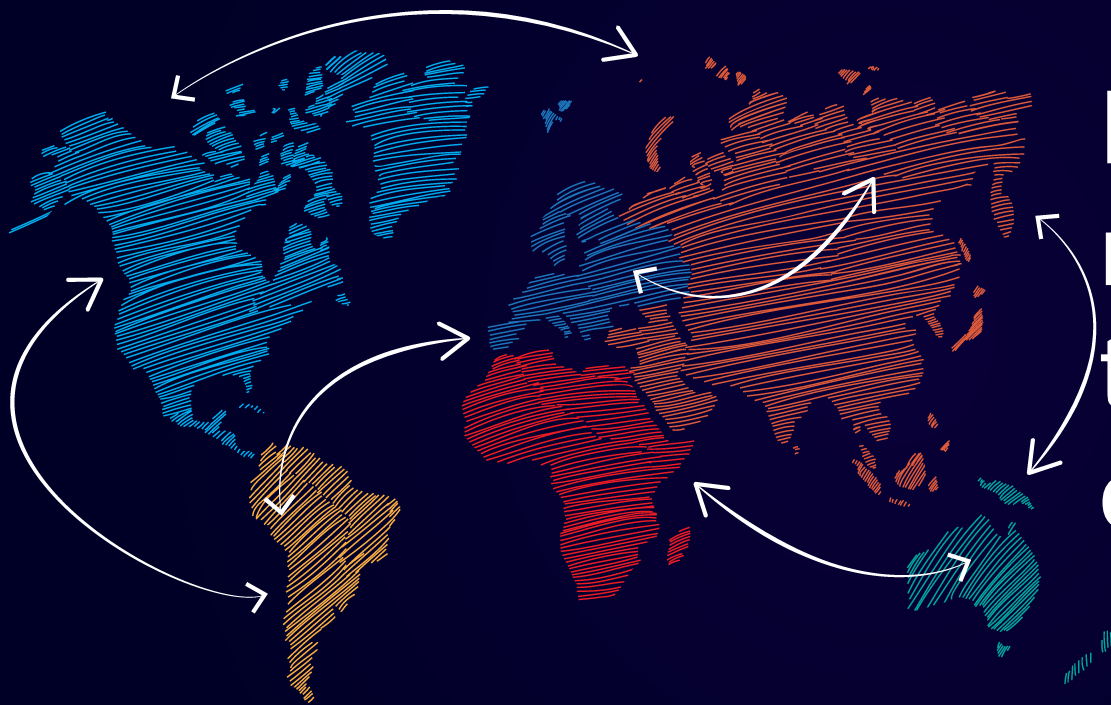
- Host in same leaf require VLAN/L2/L3 forwarding
- Host at different leaf require MPLS through spine

Adjust pipeline for a larger MPLS table



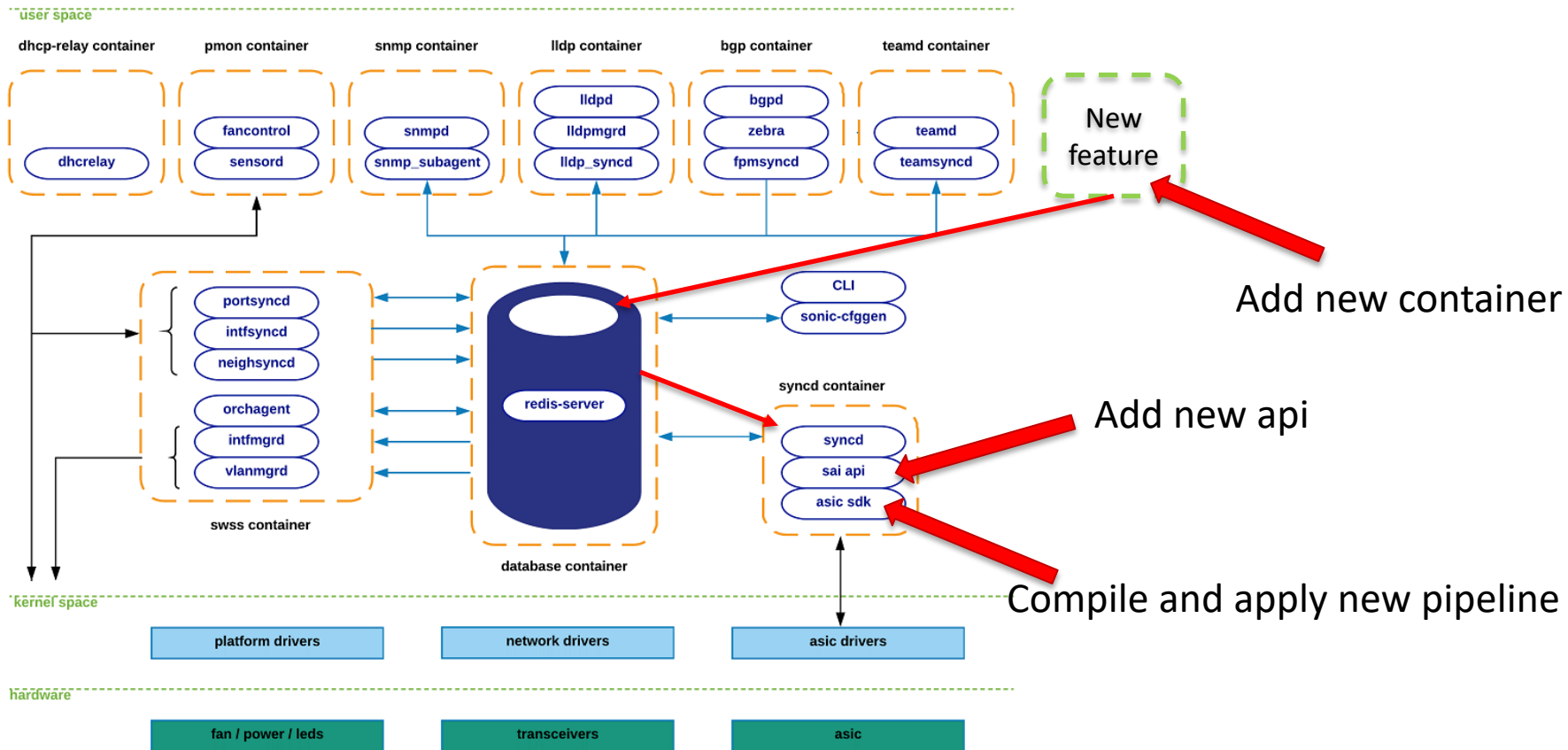
Adjust pipeline for a larger routing table



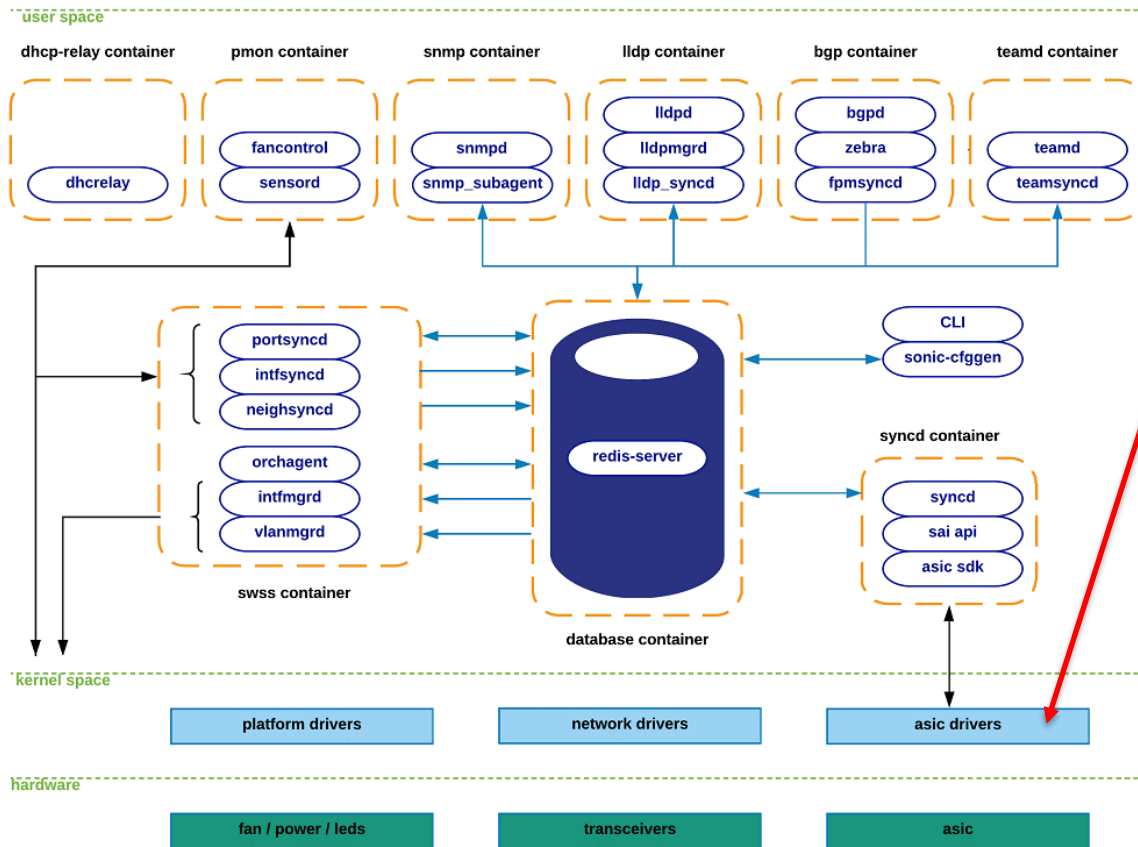


How to apply
new pipeline
table features
on SONiC?

Possible way to add new feature in SONiC (1/2)



Possible way to add new feature in SONiC (2/2)

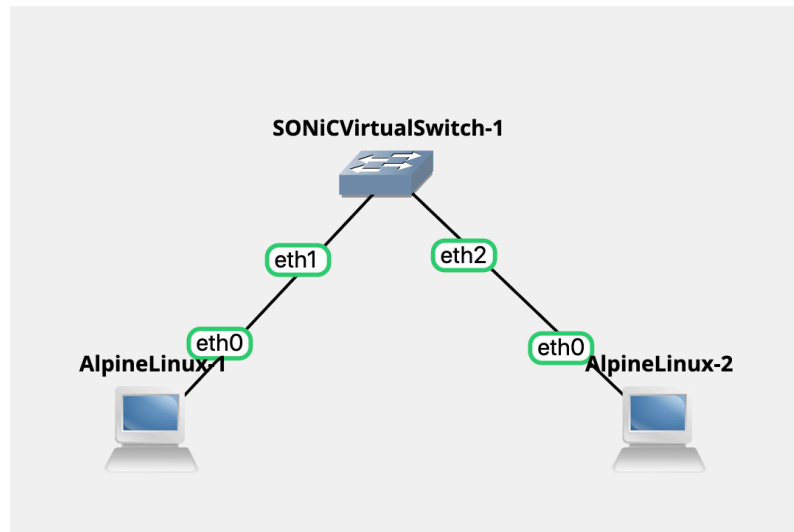


All-in-One:

1. Compile new pipeline
2. Add new feature
3. Control the table it owns

docker-sonic-vs.gz and docker-sonic-p4.gz

- Docker image for all-in-one software virtual switch and p4 software switch (gzip tar archive)
- Run gns3 for emulation



How to configure SONiC?



- Configuration Database
 - <https://github.com/Azure/sonic-swss/blob/master/doc/Configuration.md>
- Command Line Interface (CLI)
 - <https://github.com/Azure/sonic-utilities/blob/master/doc/Command-Reference.md>

Any Questions?



If you want to know more about
SONiC on Edgecore switch,
please visit our booth

A world map is centered on a dark blue background. The continents are filled with a dense, hand-drawn style of lines in various colors: North America is light blue, South America is yellow, Europe and Africa are red, and Asia and Australia are teal. Six white, curved arrows form a circular path around the map, pointing clockwise from the top to the bottom, symbolizing a global network or cycle.

OPEN SOURCE NETWORKING DAYS