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: Ubuntu, Debian, Red Hat

Hardware: Server, Switch

Software: ONIE, Open Network Linux (ONL), Go-BGP, Edge-Core Networks
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
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

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

```c
typedef struct _sai_route_api_t {
    sai_create_route_entry_fn create_route_entry;
    sai_remove_route_entry_fn remove_route_entry;
    sai_set_route_entry_attribute_fn set_route_entry_attribute;
    sai_get_route_entry_attribute_fn get_route_entry_attribute;
    sai_bulk_create_route_entry_fn create_route_entries;
    sai_bulk_remove_route_entry_fn remove_route_entries;
    sai_bulk_set_route_entry_attribute_fn set_route_entries_attribute;
    sai_bulk_get_route_entry_attribute_fn get_route_entries_attribute;
} sai_route_api_t;
```
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}/

```json
1 {
  "skip_pcied": false,
  "skip_fancontrol": true,
  "skip_thermalctld": true,
  "skip_ledd": true,
  "skip_xcvrd": false,
  "skip_psud": false,
  "skip_syseepromd": false
}
```
Sdk & Platform for Syncd

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

```
BFN_PLATFORM = bfnplatform_20201023_deb9.deb
$(BFN_PLATFORM)_URL = "https://github.com/barefootnetworks/sonic-release-pkgs/raw/dev/$(BFN_PLATFORM)"
SONIC_ONLINE_DEBS += $(BFN_PLATFORM)
$(BFN_SAI_DEVPackages) += $(BFN_PLATFORM)
```

```
BFN_SAI = bfn sdk 20201023_deb9.deb
$(BFN_SAI)_URL = "https://github.com/barefootnetworks/sonic-release-pkgs/raw/dev/$(BFN_SAI)"
$(BFN_SAI)_DEPENDE += $(LIBPATH_GNRL3)$(LIBPATH)
$(eval $(call add_conflict_package,$(BFN_SAI),$LIBSAI_DEV))
$(BFN_SAI)_RDEPENDS += $(LIBPATH_GNRL3)
SONIC_ONLINE_DEBS += $(BFN_SAI)
$(BFN_SAI_DEVDEVPackages) += $(BFN_SAI)
```
1. Modify bfn-platform.mk & bfn-sai.mk
2. $ make target/docker-syncd-bfn.gz
3. Upload new container to SONiC

https://github.com/barefootnetworks/sonic-release-pkgs/tree/dev
Upgrade SONiC syncd

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

```
REPOSITORY                  TAG               IMAGE
docker-snmp                latest            docker-snmp/latest
docker-snmp                latest            docker-snmp/latest
docker-sonic-mgmt-framework/latest
docker-sonic-telemetry/latest
docker-router-advertiser/latest
docker-dhcp-relay/latest
docker-1ldp/latest
docker-teamd/latest
docker-orchagent/latest
docker-fpm-frr/latest
docker-platform-monitor/latest
docker-platform-monitor/latest
docker-platform-monitor/latest
docker-platform-monitor/latest
docker-platform-monitor/latest
docker-platform-monitor/latest
snmp
mgmt-framework
telemetry
radv
dhcp_relay
lldp
teamd
swss
bgp
pmon
database

docker images
```
Pipeline selection from configuration

$ vim /etc/sonic/config_db.json

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

$ sudo config reload -y
Next, how do we leverage with pipeline selection in SONiC?
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

- Parser
  - Ethernet address table
  - IPv4 address table
  - IPv4 address table

- Spine 1
- Spine 2

- Leaf 1
- Leaf 2

- MPLS Network

- Deparser
  - IPv4 address table
  - MPLS table
How to apply new pipeline table features on SONiC?
Possible way to add new feature in SONiC (1/2)

Add new container
Add new api
Compile and apply new pipeline
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)
Any Questions?

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