OPEN POSSIBILITIES.

SAI Challenger: The SONiC-Based Framework for SAI Testing and Integration

OCP GLOBAL SUMMIT

NOVEMBER 9-10, 2021

SAI Challenger: The SONiC-Based Framework for SAI Testing and Integration

Andriy Kokhan
Networking Solutions Architect, PLVision

OPEN POSSIBILITIES.

OPEN COMMUNITY®

OCP GLOBAL SUMMIT
NOVEMBER 9-10, 2021

About PLVision

NETWORKING

- Networking software development company based in Poland and Ukraine
- Focused on integration and development of Network Operating Systems
- Many years of collaboration with leading switch silicon vendors

Click to add text

- Vast experience with SONiC and SAI
- Contributor to ONF's Stratum
- New Community Member to OCP

sai

OPEN COMMUNITY®

SONiC

OPEN POSSIBILITIES.

OCP GLOBAL SUMMIT
NOVEMBER 9-10, 2021

Network Disaggregation

NETWORKING

New benefits:

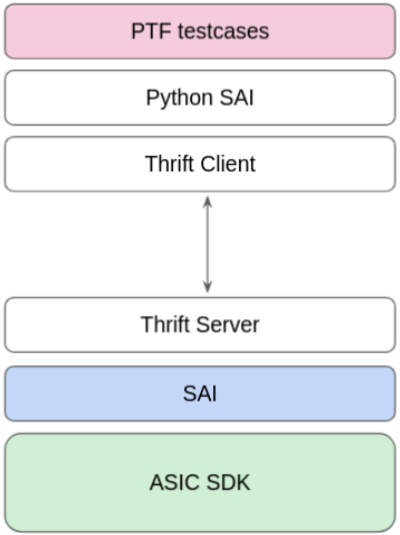
- openness – free access to sources
- standardization – availability of specifications
- diversification – freedom of choice

OR

New challenges (freedom is not free):

- integration – interoperability and new components
- validation – who? when? how?
- maintenance – components update, release cycle

SAI as a Crucial NOS Component



PTF testcases

Python SAI

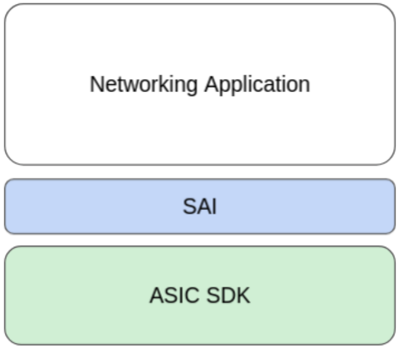
Thrift Client

Thrift Server

SAI

ASIC SDK

SAI Testing



Networking Application

SAI

ASIC SDK

NOS

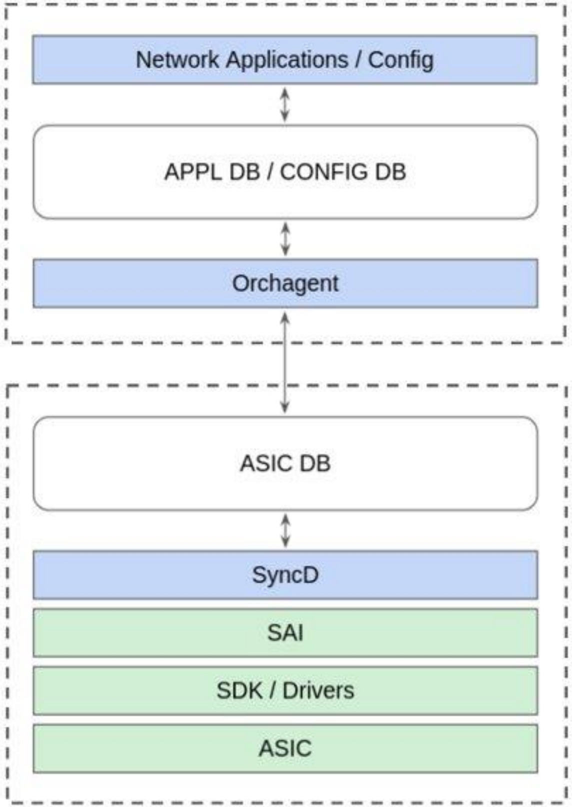
SAI defines a vendor-independent way of controlling forwarding elements in a uniform manner.

Usually, we test SAI as an independent component through the Thriftified Python interface.

Can we make SAI testing even more efficient?

SONiC Architecture

- **Orchagent** – converts configuration from SONiC applications representation into SAI representation and writes it into ASIC DB.
- **ASIC DB** – the only communication channel between SyncD and SONiC applications.
- **SyncD** – SONiC daemon that uses vendor SAI to configure a switching silicon.



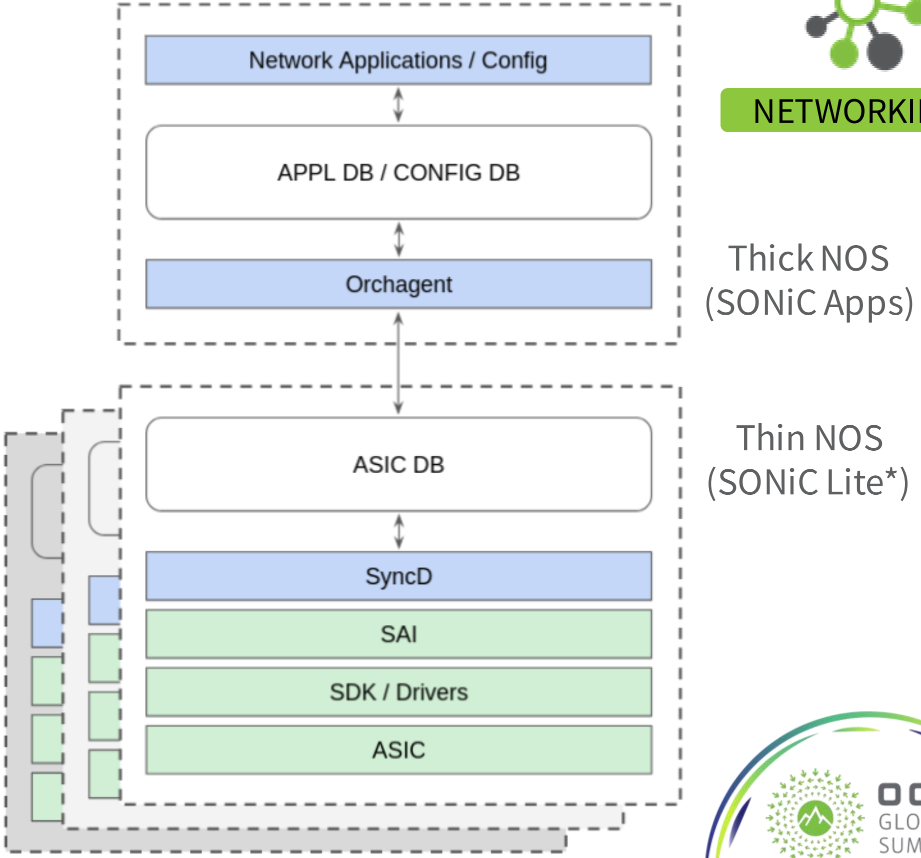
Thick NOS
(SONiC Apps)

Thin NOS
(SONiC Lite*)

NETWORKING

SONiC Platforms

- Normally, the same SONiC applications are used by all SONiC platforms.
- Each SONiC platform uses its own docker-syncd-<platform> image.
- Still the same SyncD sources are used by all SONiC platforms.



NETWORKING

OPEN POSSIBILITIES.

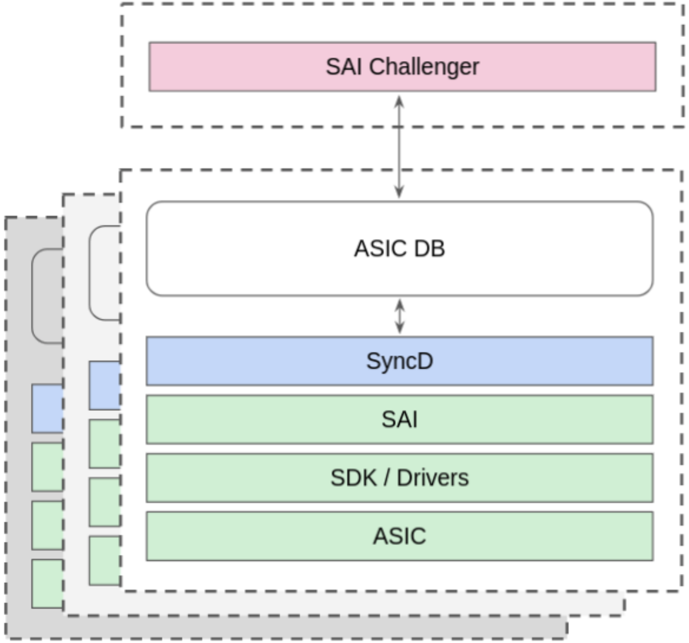
OCP GLOBAL SUMMIT
NOVEMBER 9-10, 2021

SAI Challenger

- **SyncD** – an application that uses vendor SAI to configure a switching silicon.
- **Redis DB** – as a northbound API.
- Simple **CRUD API** to operate on SAI data in ASIC DB.
- **pyTest** – as a framework for TCs development and execution.
- **PTF** to send/receive packages over the dataplane.

SAI Challenger

Docker-based environment with pre-installed Redis DB and SyncD SONiC application + CRUD API.



Thin NOS (SONiC Lite*)

Use Cases

NETWORKING

Development

Simplified networking applications prototyping due to the native integration with SONiC – the SDN-like approach with syncd as a Thin NOS.

Testing

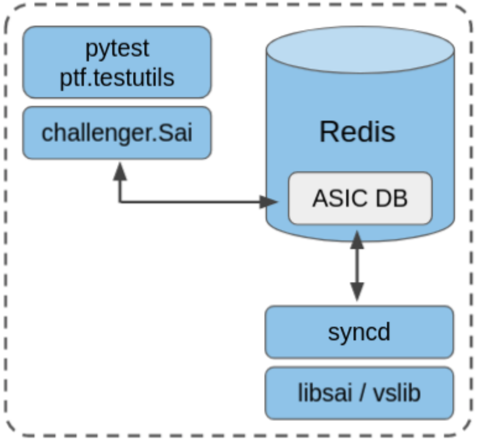
SAI integration and testing in one shot.
"Pure SAI" test cases code with no extra wrappers.
TCs development on top of SONiC libsaivs (no HW).

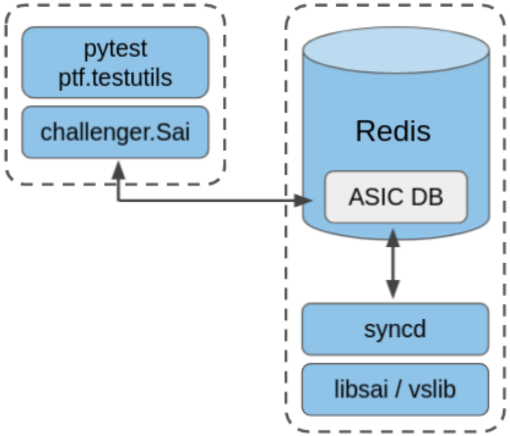
Debugging

Ease of reproducing the user scenarios based on SONiC sairedis.rec files. Simple CLI for SAI debugging.

Operation Modes

NETWORKING





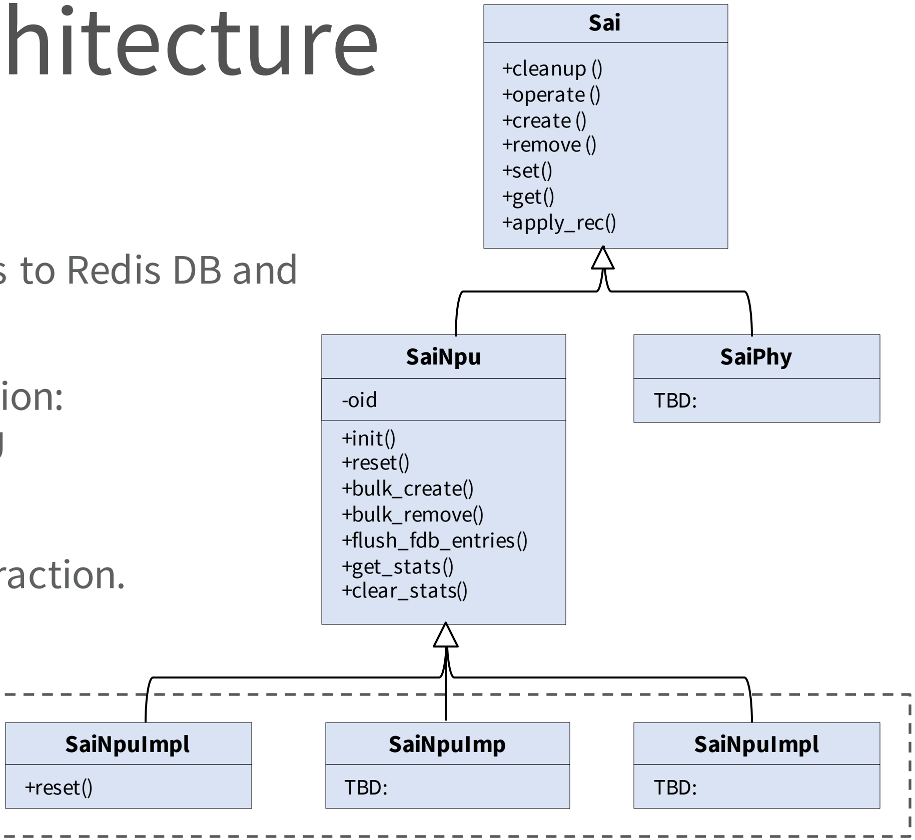
The standalone mode - both SyncD and pyTest are running in the same Docker container.

The client-server mode - SyncD and pyTest are running in the separate Docker containers.

High-Level Architecture

Three layers of abstraction:

1. The basic class Sai interfaces to Redis DB and defines generic CRUD API.
2. The SAI switch type abstraction:
   - SAI_SWITCH_TYPE_NPU
   - SAI_SWITCH_TYPE_PHY
3. The concrete NPU/PHY abstraction.



Sai

+cleanup ()
+operate ()
+create ()
+remove ()
+set()
+get()
+apply_rec()

SaiNpu

-oid

+init()
+reset()
+bulk_create()
+bulk_remove()
+flush_fdb_entries()
+get_stats()
+clear_stats()

SaiPhy

TBD:

SAI Platform Plugins

SaiNpuImpl

+reset()

SaiNpuImp

TBD:

SaiNpuImpl

TBD:

Types of Testing

NETWORKING

Unit testing – SAI testcases with no traffic running. E.g.:

- per SAI object's attribute get/set/get operations;
- SAI objects scaling testing;
- SAI object's attributes negative testing;

Functional testing – SAI testcases that implement simple networking scenarios for the specific SAI objects (FDB, VLAN, RIF, etc.). These TCs test the dataplane by running traffic with scapy utility.

Integration testing – SAI testcases that are based on **sairedis.rec** files either generated by SONiC Orchagent or manually written.

Unit Testing

NETWORKING

```
@pytest.mark.parametrize(
    "attr,attr_type,attr_val",
    [
        ("SAI_VLAN_ATTR_VLAN_ID",               "sai_uint16_t",       "100"),
        ("SAI_VLAN_ATTR_MEMBER_LIST",           "sai_object_list_t",  "0:null"),
        ("SAI_VLAN_ATTR_MAX_LEARNED_ADDRESSES", "sai_uint32_t",       "0"),
        ("SAI_VLAN_ATTR_STP_INSTANCE",          "sai_object_id_t",    None),
        ("SAI_VLAN_ATTR_LEARN_DISABLE",         "bool",               "false"),
    ]
)
def test_get_before_set_attr(npu, dataplane, sai_vlan_obj, attr, attr_type, attr_val):
    status, data = npu.get_by_type(sai_vlan_obj, attr, attr_type, do_assert=False)
    npu.assert_status_success(status)

    assert data.value() == attr_val
```

Functional Testing

```
vlan_mbr_oid = self.create(SaiObjType.VLAN_MEMBER,
        [
            "SAI_VLAN_MEMBER_ATTR_VLAN_ID",           vlan_oid,
            "SAI_VLAN_MEMBER_ATTR_BRIDGE_PORT_ID",    bp_oid,
            "SAI_VLAN_MEMBER_ATTR_VLAN_TAGGING_MODE", "SAI_VLAN_TAGGING_MODE_TAGGED"
        ])


if npu.run_traffic:
    pkt = simple_tcp_packet(eth_dst=macs[1], eth_src=macs[0],
                            ip_dst='10.0.0.1', ip_id=101, ip_ttl=64)
    send_packet(self, 0, pkt)
    verify_packets(self, pkt, [1])


self.remove(vlan_mbr_oid)
```

Integration Testing

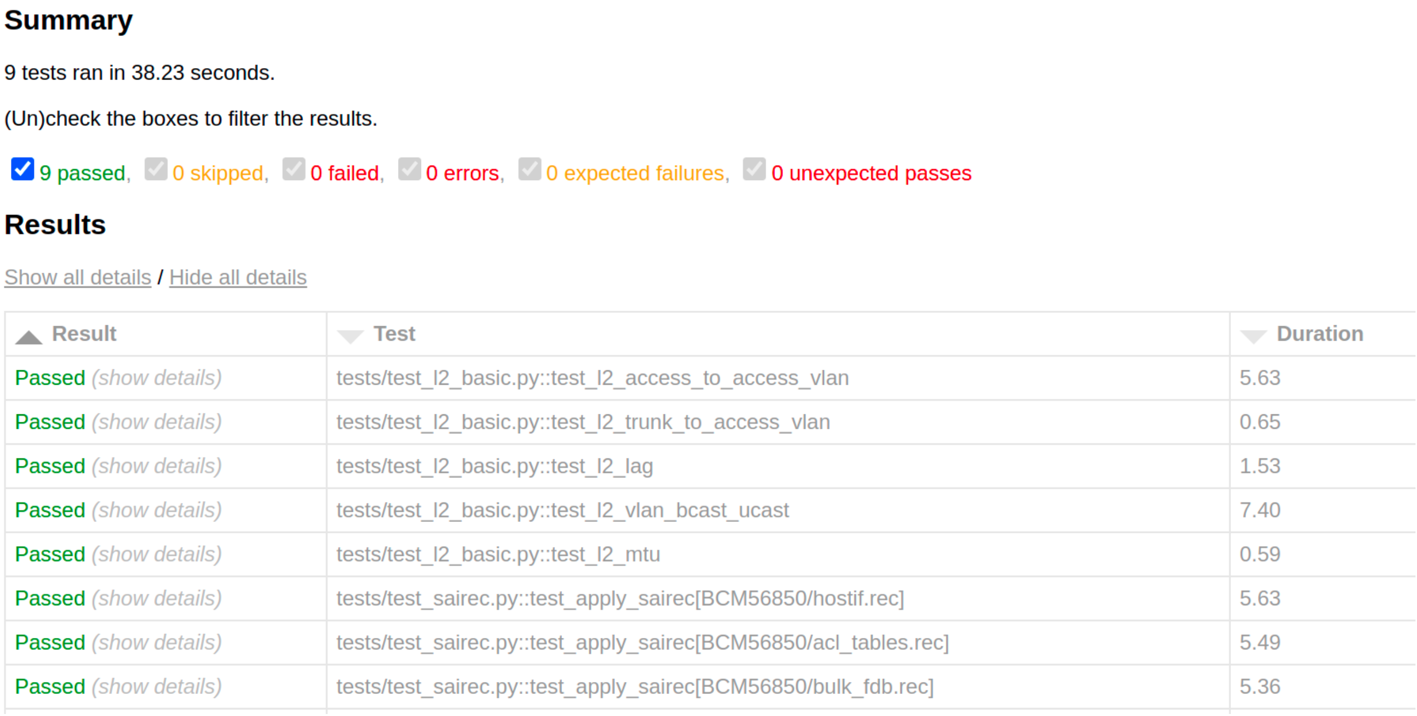
```
@pytest.mark.parametrize(
    "fname",
    [
        "BCM56850/full.rec",
        "BCM56850/hostif.rec",
        "BCM56850/acl_tables.rec",
        "BCM56850/bulk_fdb.rec",
    ],
)
def test_apply_sairec(npu, exec_params, dataplane, fname):
    npu.apply_rec("/sai/sonic-sairedis/tests/" + fname)
    npu.reset()
```

… or just do it manually

NETWORKING

```
sai --help

sai create switch \
    SAI_SWITCH_ATTR_INIT_SWITCH true \
    SAI_SWITCH_ATTR_TYPE SAI_SWITCH_TYPE_NPU

sai get oid:0x21000000000000 SAI_SWITCH_ATTR_PORT_LIST sai_object_list_t

sai get oid:0x1000000000002 SAI_PORT_ATTR_OPER_STATUS

sai stats get oid:0x1000000000002 \
    SAI_PORT_STAT_IF_IN_UCAST_PKTS  SAI_PORT_STAT_IF_OUT_UCAST_PKTS

sai stats clear oid:0x1000000000002 \
    SAI_PORT_STAT_IF_IN_UCAST_PKTS SAI_PORT_STAT_IF_OUT_UCAST_PKTS
```

Test Results



Summary

9 tests ran in 38.23 seconds.

(Un)check the boxes to filter the results.

☑ 9 passed, ☑ 0 skipped, ☑ 0 failed, ☑ 0 errors, ☑ 0 expected failures, ☑ 0 unexpected passes

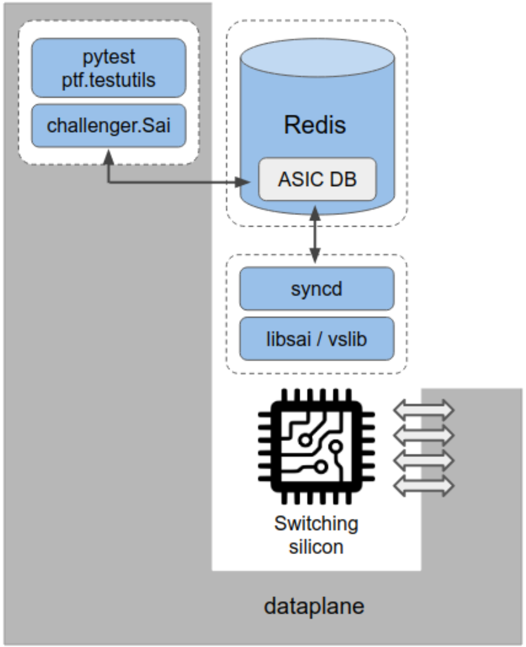
Results

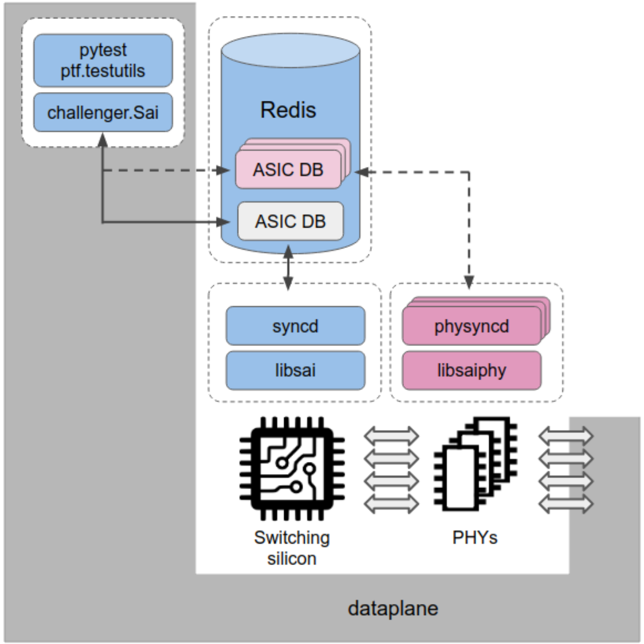
Show all details / Hide all details

Result	Test	Duration
Passed *(show details)*	tests/test_l2_basic.py::test_l2_access_to_access_vlan	5.63
Passed *(show details)*	tests/test_l2_basic.py::test_l2_trunk_to_access_vlan	0.65
Passed *(show details)*	tests/test_l2_basic.py::test_l2_lag	1.53
Passed *(show details)*	tests/test_l2_basic.py::test_l2_vlan_bcast_ucast	7.40
Passed *(show details)*	tests/test_l2_basic.py::test_l2_mtu	0.59
Passed *(show details)*	tests/test_sairec.py::test_apply_sairec[BCM56850/hostif.rec]	5.63
Passed *(show details)*	tests/test_sairec.py::test_apply_sairec[BCM56850/acl_tables.rec]	5.49
Passed *(show details)*	tests/test_sairec.py::test_apply_sairec[BCM56850/bulk_fdb.rec]	5.36

Single Approach for ASIC & PHY

NETWORKING





Call to Action

- Current open-source path: https://github.com/PLVision/sai-challenger (NOTE: pending review from the community to be moved to OCP Github)
- How to participate:
  - Add *SaiNpuImpl* plugin for your platform under *sai-challenger/npu/*
  - Propose new use cases with SONiC Lite in mind
  - Implement new basic topologies under *sai-challenger/topologies/*
  - Extend test scenarios, CLI, core functionality
- Blog post: blog/opensource/sai-challenger-sonic-based-framework

Thank you!

OCP GLOBAL SUMMIT

NOVEMBER 9-10, 2021