
# PINS Update

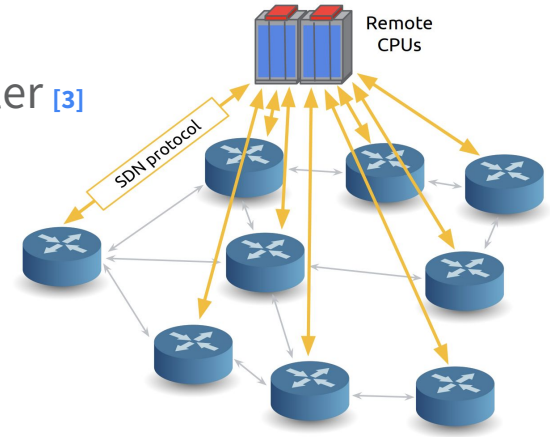
Bhagat Janarthanan, Google
Vamsi Punati, Google

Brian O'Connor, Intel
Reshma Sudarshan, Intel

# SDN & SONiC

# SDN

- Google - a major user of SDN in Data Center [1]
  - Simpler Traffic Engineering [2]
  - Easier Debugging - Network State visible to controller [3]
  - Control Plane runs on Dedicated Fast Servers
- SDN & Remote Controller Adoption growing
- Focus: Help SDN go mainstream




*[1]: Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network*
*[2]: Orion: Google's Software-Defined Networking Control Plane*

# SONiC

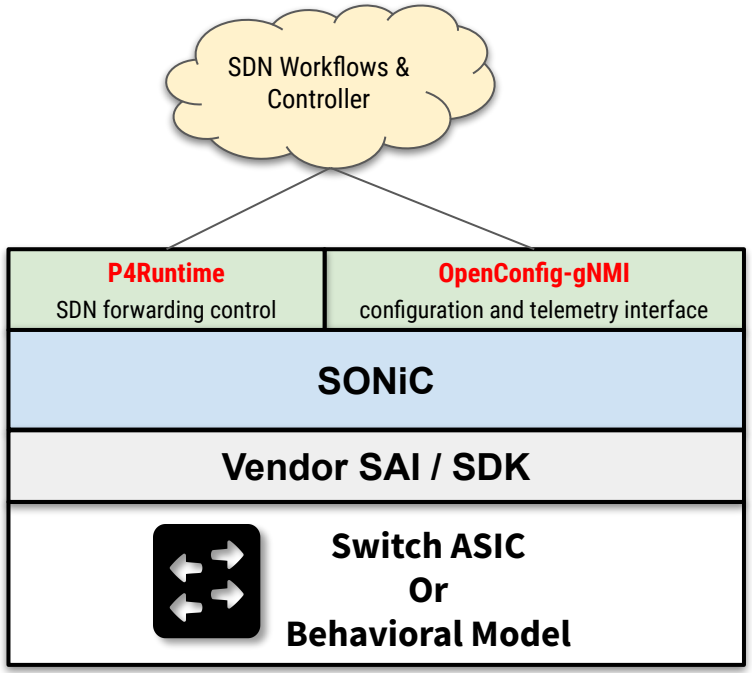
- Vendor Agnostic (thanks to SAI and common platform API)
- Decoupled Software from Hardware
- Enabling rapid innovation
- Vibrant Ecosystem
- Open Source

Can we extend SONiC to provide an incremental, opt-in path to SDN ?

- Focus on customer problems & business opportunities
- Build confidence in new infrastructure as we go

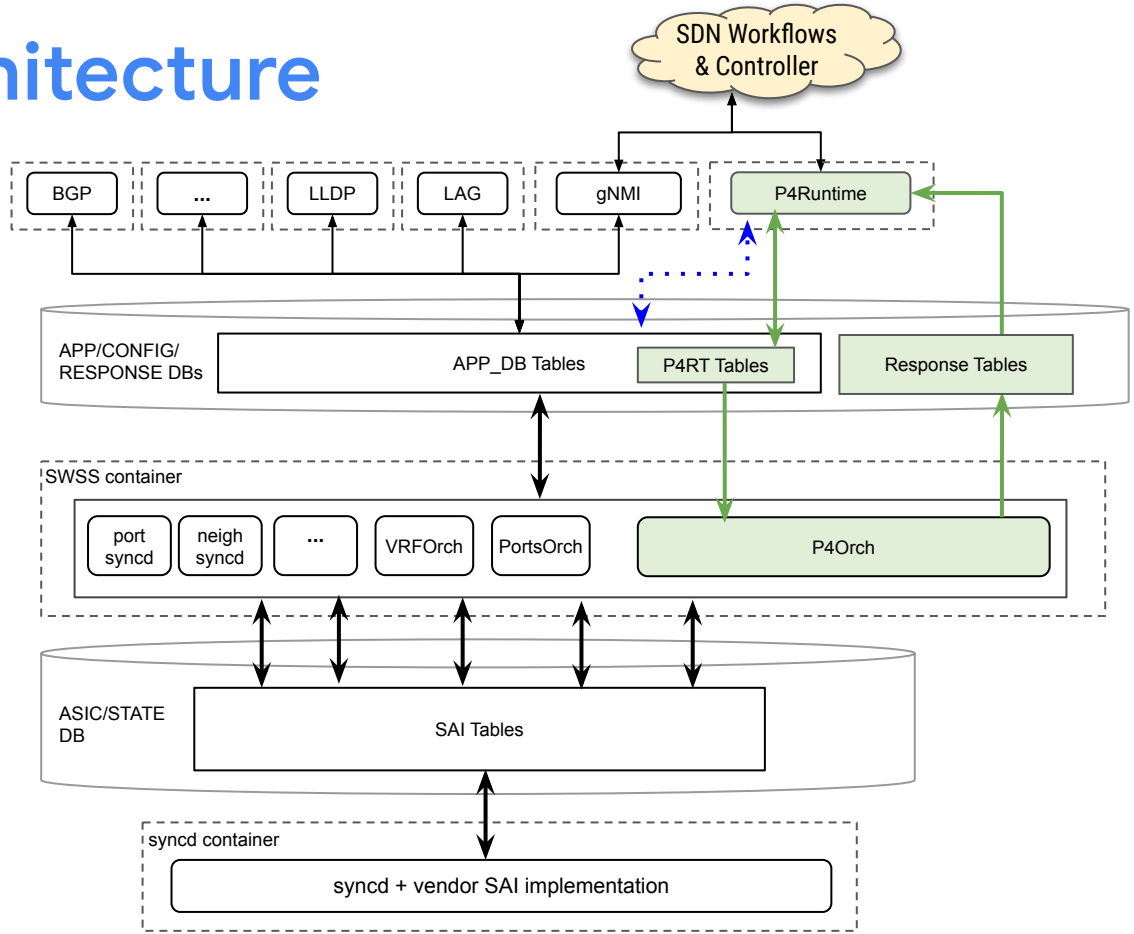
# Solution: PINS

- P4 used to model the SAI pipeline
- SDN protocol: P4Runtime
  - Standard, open, silicon-independent
  - Enables runtime-control of data plane objects
- Management protocols: OpenConfig
  - Standard, open, widely used
  - *Already used in SONiC*



# PINS Architecture
# &
# The Green Path

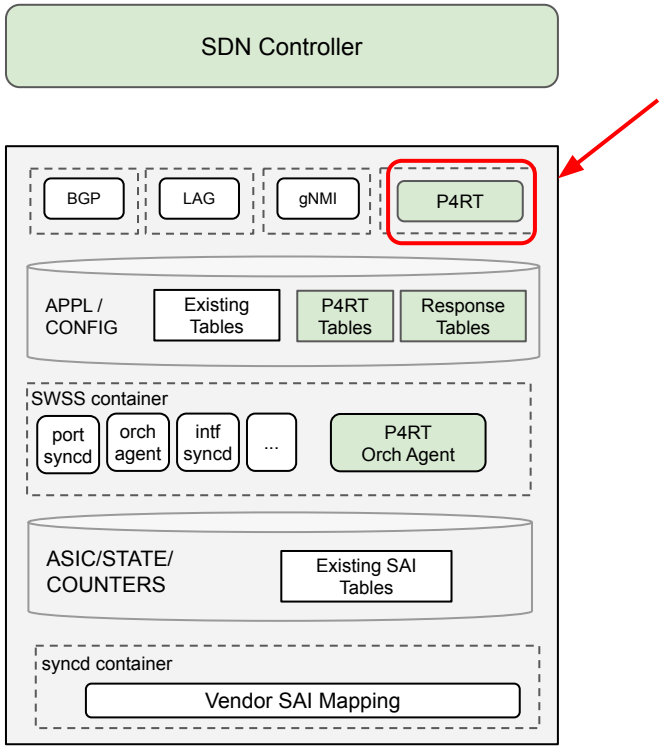
# PINS Architecture



# P4Runtime App

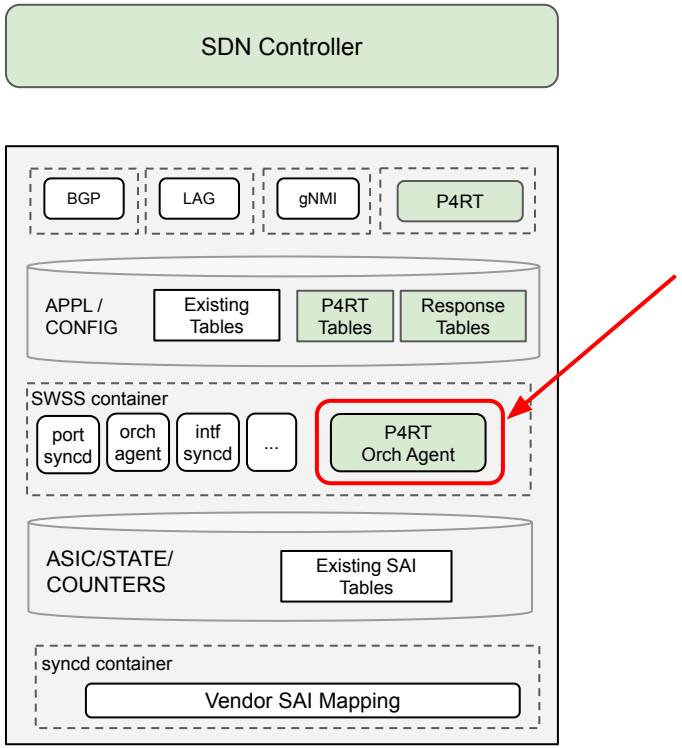
- P4Runtime Server in a new Docker container

- Features
  - Arbitration & Roles
  - Table Format (Set Pipeline)
  - Write & Readback
  - Packet I/O

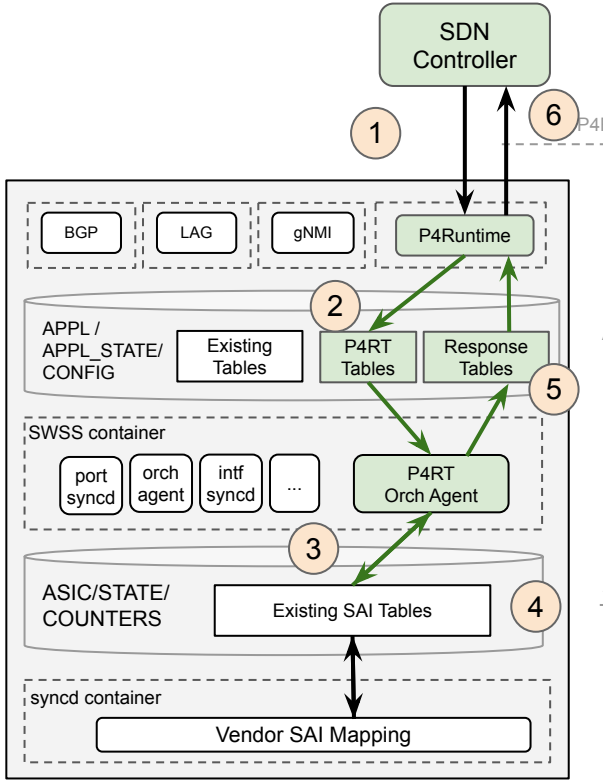


# P4 Orchagent

- New orchagent in SWSS
  - Parses APPL_DB entries
  - Maintains objects, refcounts
  - Translates intent to ASIC_DB
- Handles ordering dependencies
- Supports response path



# Example: Installing a route




```
RPC: Write
type: INSERT
ipv4_table_entry
  match
    vrf_id: "vrf-8"
    ipv4_dst: "192.168.0.0/24"
  action: set_nexthop_id
    nexthop_id: "s1"
```

APPL DB

```
P4RT:FIXED_IPV4_TABLE:{"match/vrf_id":"vrf-8",
                       "match/ipv4_dst":"192.168.0.0/24"}
  "action" = "set_nexthop_id"
  "param/nexthop_id" = "s1"
```

ASIC DB

```
"ASIC_STATE:SAI_OBJECT_TYPE_ROUTE_ENTRY:{"dest":"192.168.0.0/24",
      "switch_id":"oid:0x21000000000000","vr":"oid:0x3000000000008"}
  "SAI_ROUTE_ENTRY_ATTR_NEXT_HOP_ID" = "oid:0x500000000097b"
```

# Open Source Status

| SAI.p4 | P4Runtime Service |
| --- | --- |
| Parsed Headers<br>• Ethernet<br>• IPv4, IPv6<br>• TCP, UDP<br>• ICMP, ARP, GRE<br>Fixed SAI Tables<br>• IPv4 and IPv6 Routing with WCMP<br>• Next hops with RIF and Neighbor MACs<br>• Mirroring<br>• *L3 Admit (Dest MAC and ingress port) (planned)*<br>• *GRE Encap (planned)*<br>Configurable SAI Tables (User-defined)<br>• Pre-routing ACL<br>• Post-routing ACL<br>• Egress ACL<br><br>*Several reference SAI.p4 profiles available!* | • Table Entry Write and Read<br>• Packet In with Metadata (ingress / target egress ports)<br>• Packet Out (Direct TX and Submit to Ingress)<br>• P4Runtime Roles and Arbitration<br>• Port ID translation (optional)<br>• P4 Program (p4info) validation |

# Community Roadmap

In progress

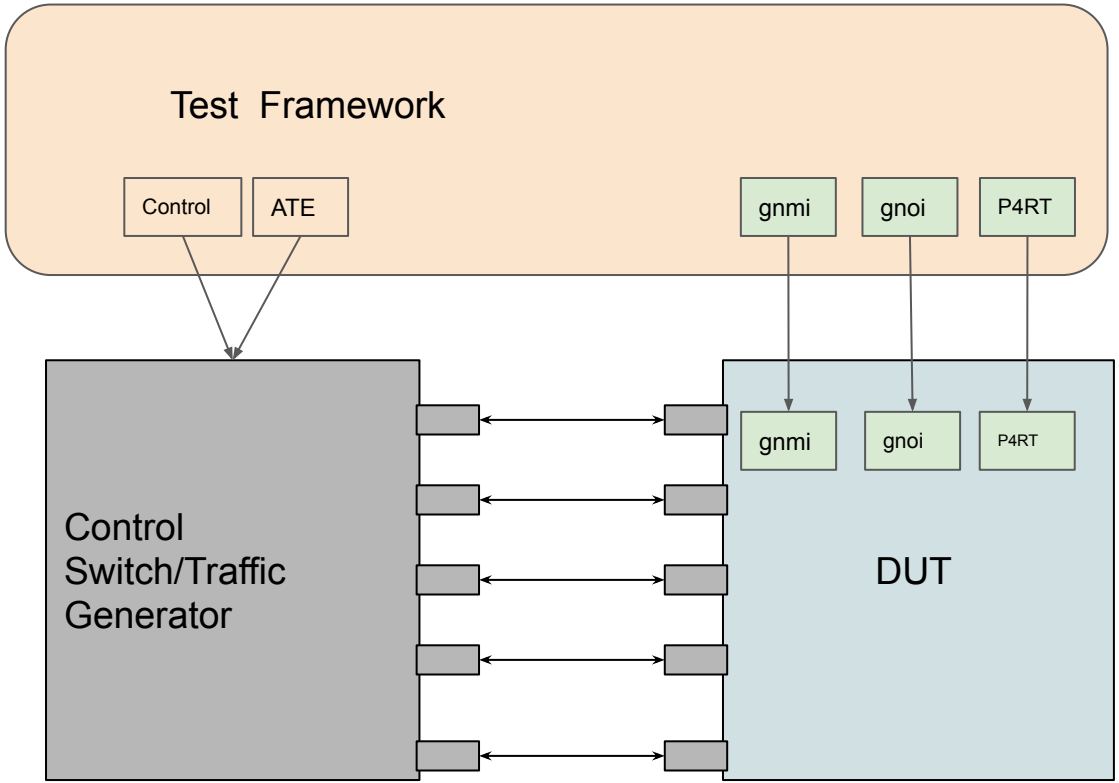
- **System level testing** *(Vamsi)*
- **SAI Generic Extension Path** *(Reshma)*
- P4Runtime Application Configuration

Looking ahead / Opportunities for Collaboration

- **Chassis Switches** *(Bhagat)*
- **Virtual Switches** *(Bhagat)*
- P4 Pipeline updates and warm boot

# PINS
# &
# System Level Testing

# Test Topology



# Testing Frameworks

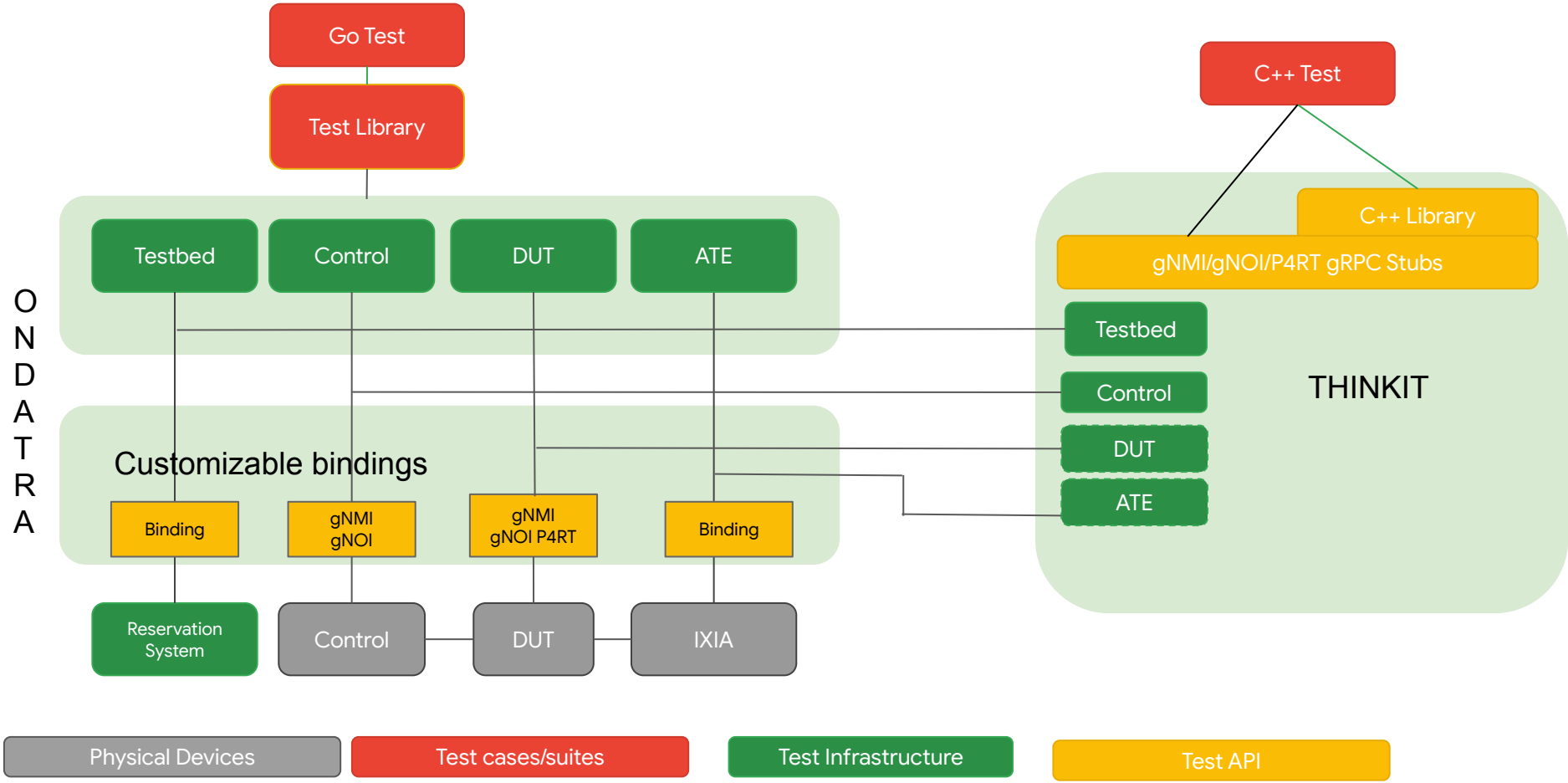
**Thinkit**

- Suited for dataplane & control plane
- Inbuilt Traffic library
- Connectivity to Ixia/Traffic Generator
- gNMI and P4 configurations for DUT/Control Switch
- Port connectivity b/w DUT and Control switch
- C++
- ~ 100 tests and growing

**Ondatra**

- Suited for management plane & operational
- Rich feature set to support openconfig models
- Interfaces to reserve test bed and pass topology information to tests
- Connectivity to DUT/Control switch on gNMI/gNOI/Console
- gNMI configurations for DUT/Control Switch
- Go
- ~ 250 tests and growing

# Thinkit on Ondatra



# Upstreaming Status

- Thinkit
  - Library and sample tests are available in PINS WG
  - https://github.com/pins/pins-infra
- Ondatra
  - Framework upstreamed
  - https://github.com/openconfig/ondatra

- Thinkit on Ondatra (In progress)
  - One framework to run all thinkit and ondatra tests

# PINS
# SAI Generic Extensions

# PINS SAI Generic Extensions - What is it ?

- New programmable capabilities implies
  - Realization of new scenarios
  - Allow users to extend the data plane pipeline using P4
  - Use the SAI pipeline as baseline
  - Enable extensions to share existing SAI objects
- Use cases
  - Congestion control and avoidance
  - L4 Load Balancing
  - Network function embedding (BNG, UPF)
  - PTP / time synchronization
- Pipeline changes are easy in P4

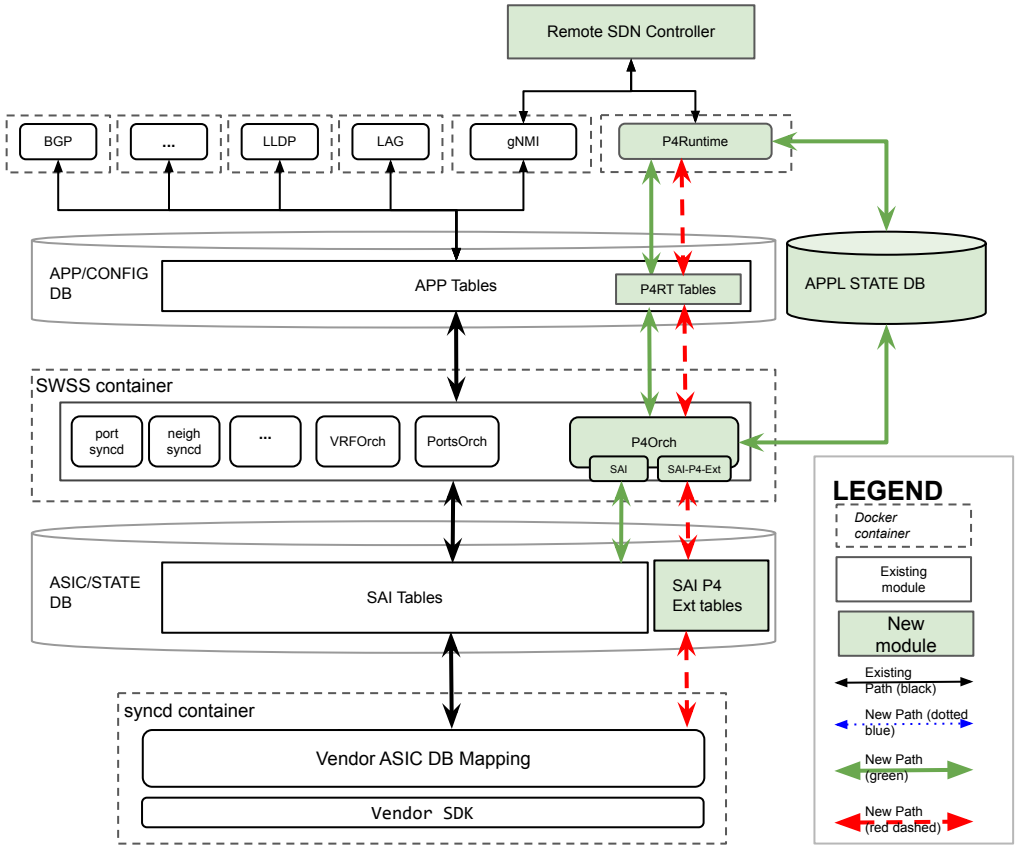
# PINS SAI Generic Extensions Architecture

Generic SONiC control-plane to support new Extensions don't require changes to SONiC or SAI

- Generic SwSS Orch to handle application requests
- Generic SAI interface that is not regenerated for every new extensions

Continue to support PINS constructs

- P4 Orch to manage generic objects
- Response Path
- Application State DB



# New Generic Programmable Object Type in SAI

SAI Generic programmable Object

Carries essential metadata inline required to program the HW block

- Name of the new non-SAI match action table
- Set of SAI attributes types
- Attribute values

```
Programmable_object_attr.id =
SAI_GENERIC_PROGRAMMABLE_ATTR_OBJECT_NAME
Programmable_object_attr.value = “Table Name”
Programmable_object_attr.id =
SAI_GENERIC_PROGRAMMABLE_ATTR_ENTRY
Programmable_object_attr.value = {
    "attributes": [
        {
            <attribute_name>: {
                "sai_metadata": {
                    "sai_attr_value_type":
"<SAI_ATTR_VALUE_TYPE_T>",
                    "brief": "Brief Attribute Description",
                    "sai_attr_flags": "<SAI_ATTR_FLAGS_T>",
                    "allowed_object_types": [ "<LIST OF
ALLOWED OBJECT TYPES>" ],
                    "default_value": "<DEFAULT ATTR VALUE>"
                },
                "value": <VALUE of the attribute>
```

# SONiC PINS SoftSwitch with P4DPDK

Same software stack to manage

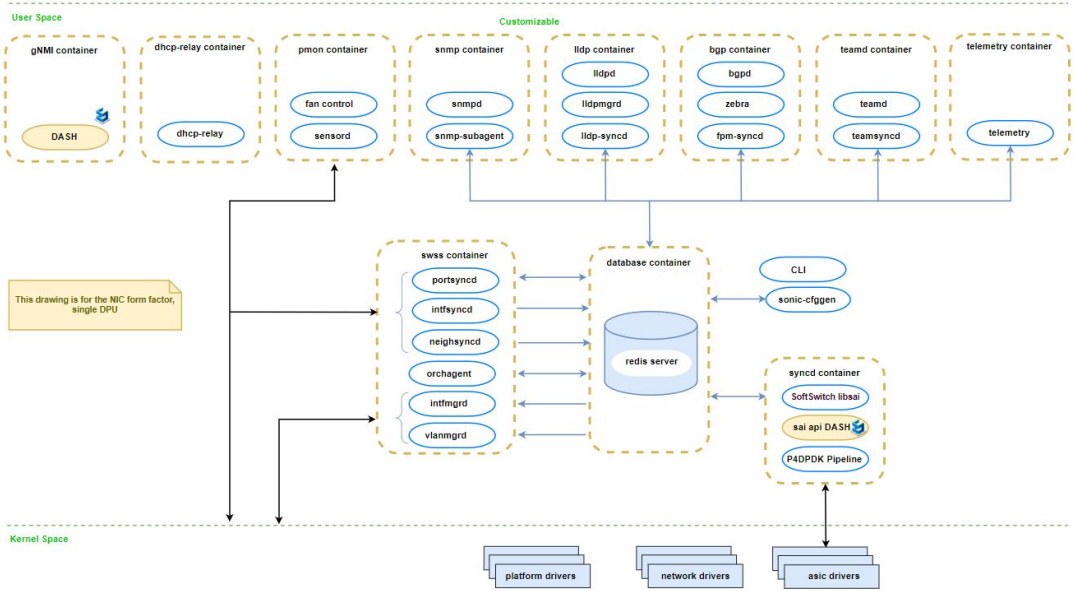
SONiC HW and SW

SONiC runs in Host / VM

- OS de-coupled from customer's environment
- Separate software lifecycle

P4-DPDK Dataplane in Host

P4 Compiled Dataplane



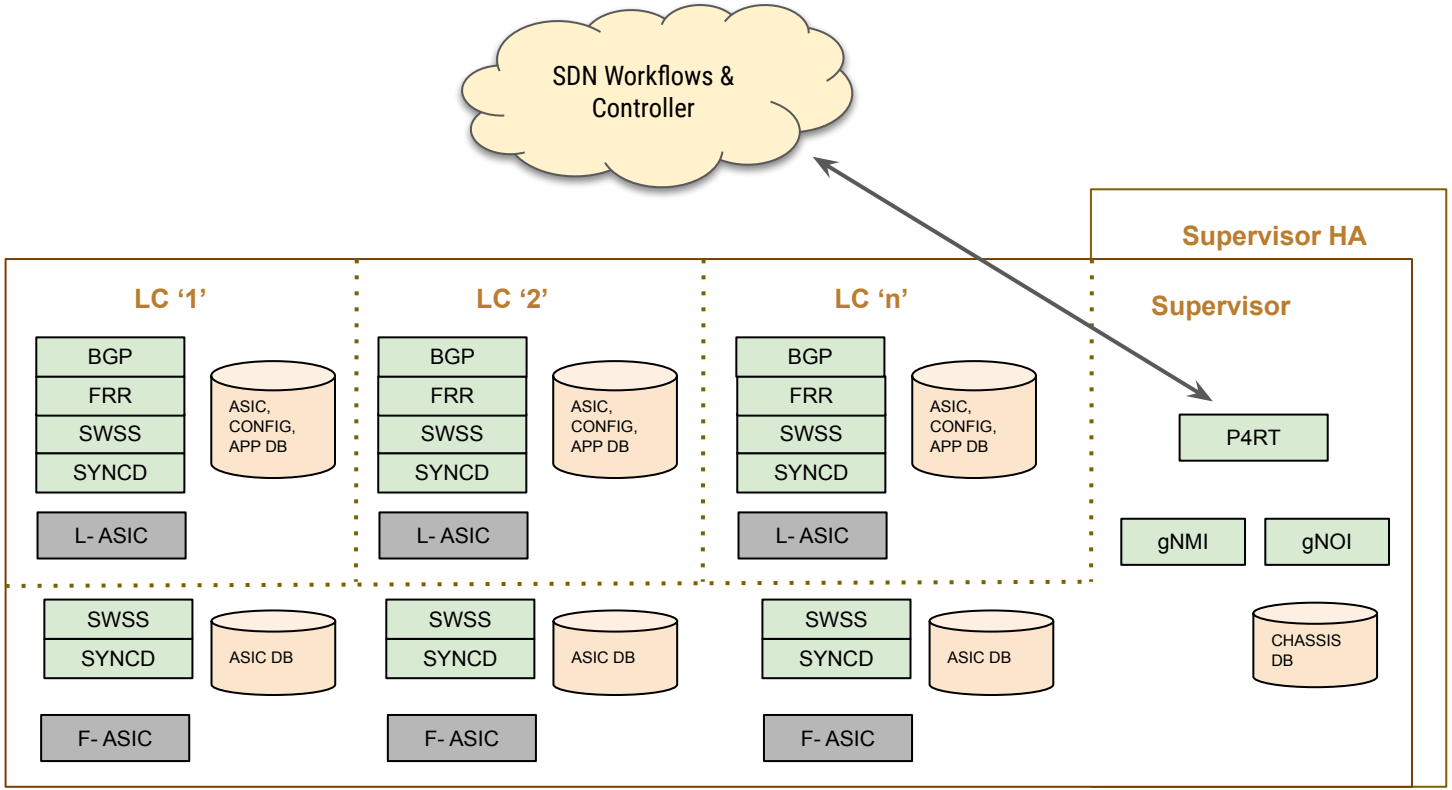

SONiC Softswitch Architecture

```
action set_nexthop_id(
nexthop_id_t nexthop_id)
    nexthop_id_valid = true;
    nexthop_id_value =
nexthop_id;
```

```
table ipv4_table {
  key = {
     hdr.ipv4.dst_addr : lpm;}
  actions = {
     set_nexthop_id;
     @defaultonly NoAction;}
  const default_action = NoAction;
```

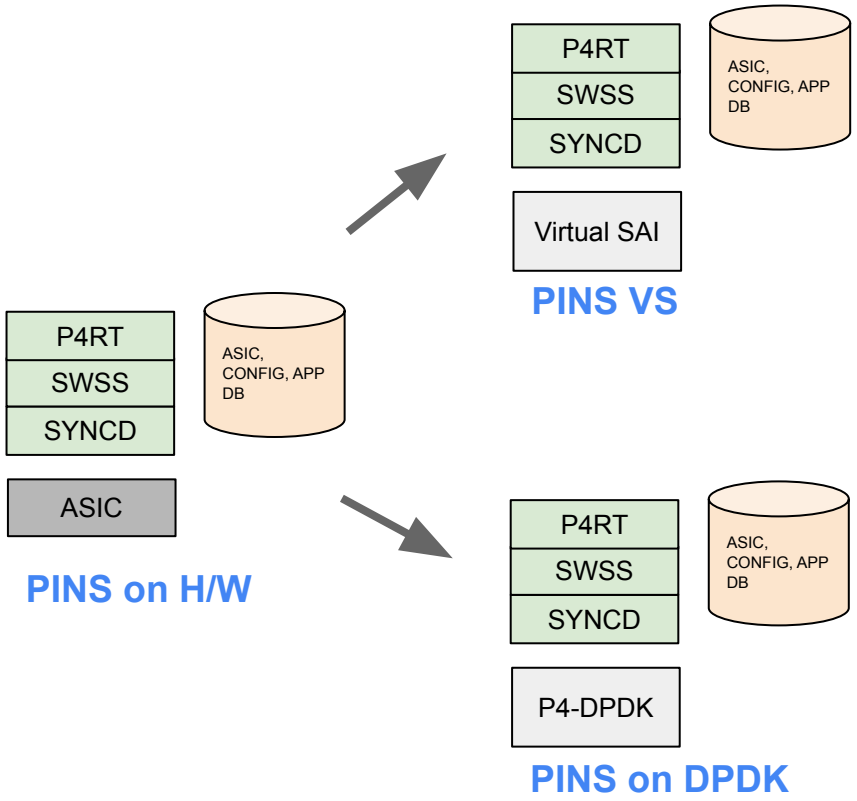
# PINS Future Challenges & Opportunities

# PINS on Chassis Architecture



# PINS on a Virtual Switch

- Control Plane Validation
  - SDN Controller & Top Half of the stack Validation
- Future
  - Dataplane & Controlplane
  - P4-DPDK



# Call for Action

- Open for collaboration and ideas
- Join for discussions and contributions
  - Intersection of various communities - P4, SONiC, SAI

- PINS subgroup in SONiC
  - Email: sonic-pins-subgroup@googlegroups.com

Q & A