PINS: P4 Integrated Network Stack

OCP Global Summit
November 9 - 10, 2021

PINS = SONiC + SDN

- **SONiC** is widely deployed, modular, open source, and vendor agnostic
  - Runs a traditional control plane (e.g. BGP)
  - Solid foundation for SDN-enabled switch OS

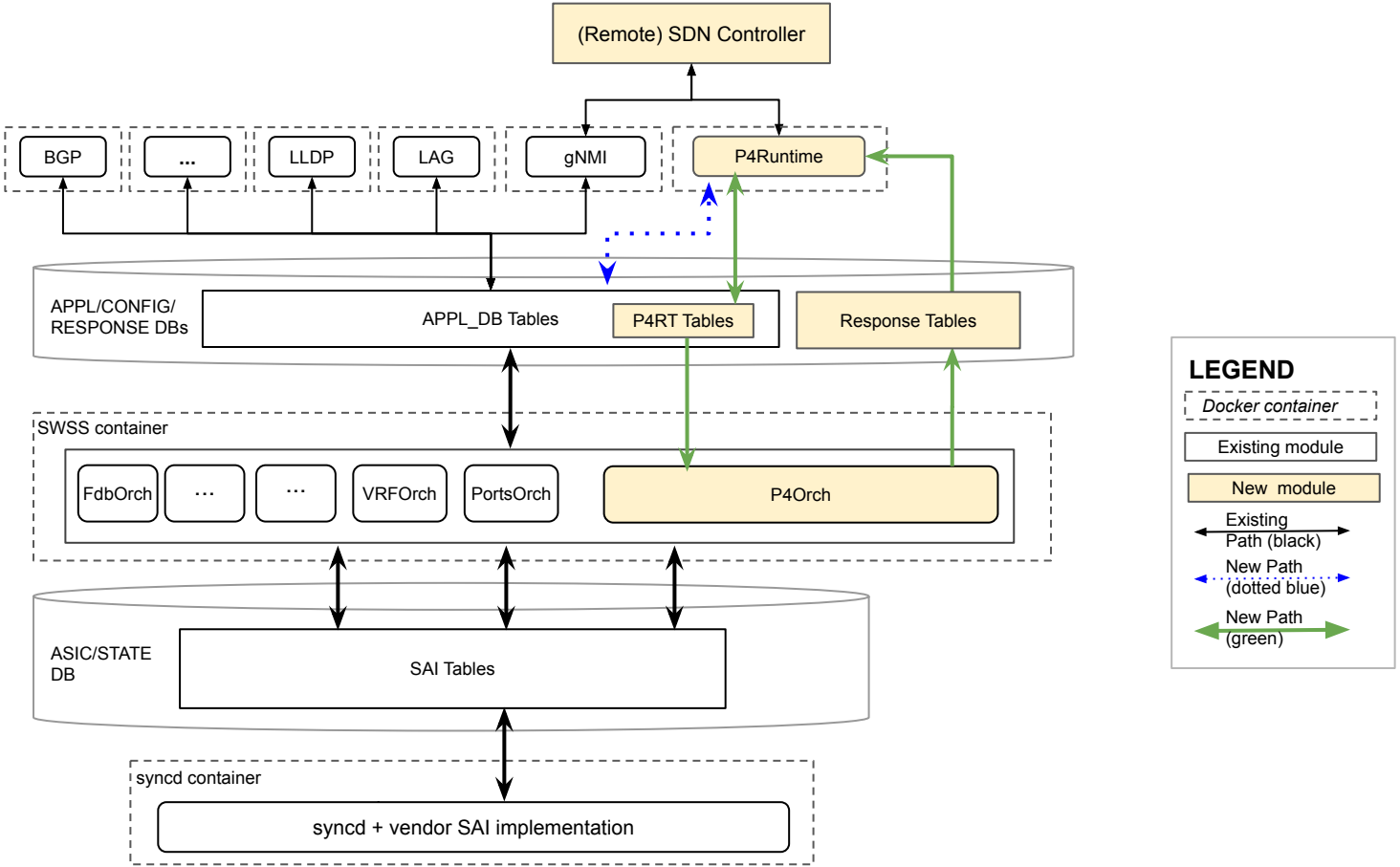
- Enabling SDN in SONiC requires:
  - *Formal Pipeline Specification*: **P4** used to model the SAI pipeline
    - Emerging as the industry standard
    - Works for fixed and programmable switching targets
    - Enables programmatic validation of the pipeline

  - *Remote interface for controlling forwarding entries*: **P4Runtime**
    - Standard, open, silicon-independent
    - Enables runtime-control of data plane objects
  - Remote interface for management / operations: **OpenConfig, gNMI, gNOI**
    - Standard, open, widely used
    - *Already used in SONiC today*

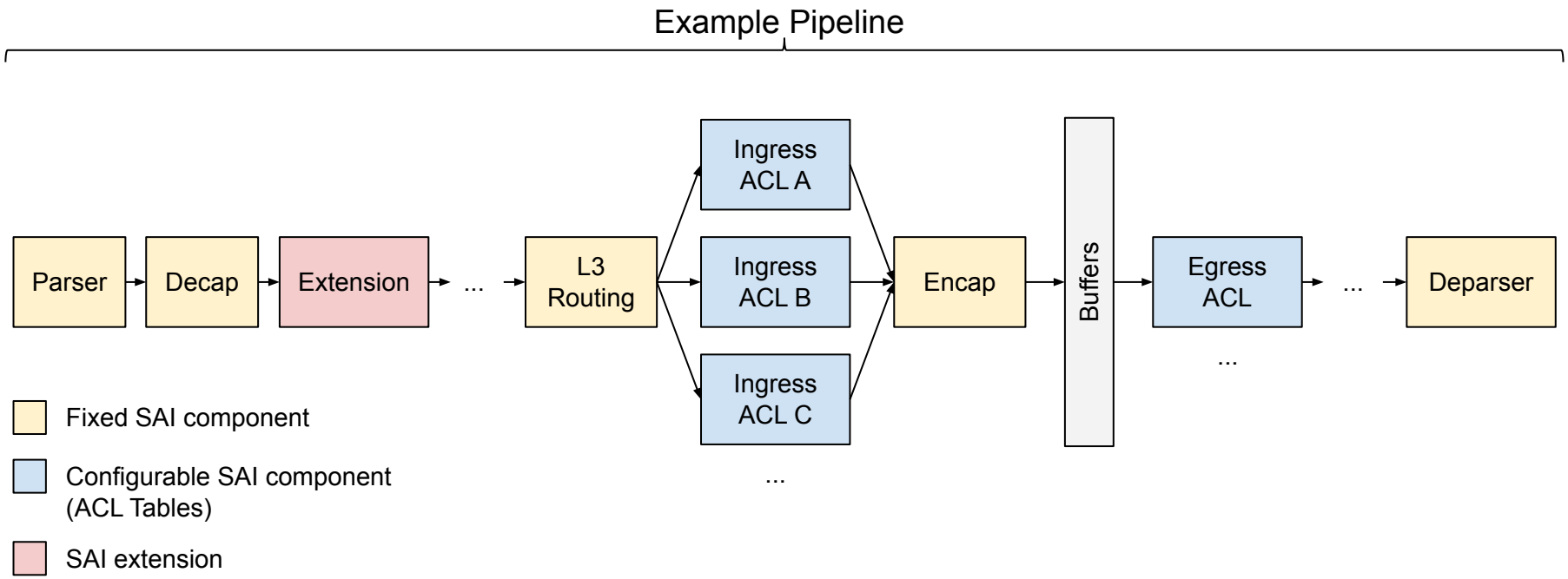
PINS Highlights

- **Hybrid Control Plane Support:** Gives network operators a choice on network control plane and which parts run where (locally or remotely).
- **Opt-In Path Towards SDN**: The P4Runtime server is added to SONiC as an optional interface enabling users to implement new functionality using SDN, and to incrementally migrate towards an SDN solution.
- **Familiar Interface**: P4 is used to model the SAI pipeline, and enables users to control all essential networking features, including L2 bridging, L3 routing, ACLs, tunnels, and more.
- **Rapid Innovation**: New features can quickly be modeled in P4 and exposed to control plane applications using P4Runtime.
- **Automated Validation**: P4 and P4Runtime enables tools to be used to test and validate every packet path automatically in the forwarding pipeline.

PINS Architecture



Reference SAI Pipeline



SAI P4 Routing

Legend:
P4 Table *(maps to SAI header)*

sai/routing.p4

```
...

@p4runtime_role(P4RUNTIME_ROLE_ROUTING)
 @id(ROUTING_IPV4_TABLE_ID)
 table ipv4_table {
   key = {
     // Sets vrf_id in sai_route_entry_t.
     local_metadata.vrf_id : exact @id(1) @name("vrf_id")
         @refers_to(vrf_table, vrf_id);
     // Sets destination in sai_route_entry_t to an IPv4 prefix.
     headers.ipv4.dst_addr : lpm @format(IPV4_ADDRESS) @id(2)
                                 @name("ipv4_dst");
   }
   actions = {
     @proto_id(1) drop;
     @proto_id(2) set_nexthop_id;
     @proto_id(3) set_wcmp_group_id;
   }
   const default_action = drop;
   size = ROUTING_IPV4_TABLE_MINIMUM_GUARANTEED_SIZE;
 }

...
```

IP Routing *(sairoute.h)*

Match Keys:
- VRF ID
- IP Dest

Action (one of):
- Set Next Hop Group ID
- Set Next Hop ID

or

Next Hop Group
(sainexthopgroup.h)

Match Keys:
- Next Hop Group ID

Action:
- Set Next Hop ID **via WCMP**

Next Hop *(sainexthop.h)*

Match Keys:
- Next Hop ID

Action:
- Set RIF ID
- Set Neighbor IP

Router Interface
(sairouterinterface.h)

Match Keys:
- RIF ID

Action:
- Set Dest Port
- Set Src MAC

Neighbor
(saineighbor.h)

Match Keys:
- RIF ID
- Neighbor IP

Action:
- Set Dest MAC

SAI P4 ACLs

sai/custom/acl_ingress.p4

```
...
 table acl_ingress_table {
   key = {
     headers.ipv4.isValid() || headers.ipv6.isValid() : optional  @sai_field(SAI_ACL_TABLE_ATTR_FIELD_ACL_IP_TYPE/IP);
     headers.ipv4.isValid() : optional                            @sai_field(SAI_ACL_TABLE_ATTR_FIELD_ACL_IP_TYPE/IPV4ANY);
     headers.ipv6.isValid() : optional                            @sai_field(SAI_ACL_TABLE_ATTR_FIELD_ACL_IP_TYPE/IPV6ANY);
     headers.ethernet.ether_type : ternary                        @sai_field(SAI_ACL_TABLE_ATTR_FIELD_ETHER_TYPE);
     headers.ethernet.dst_addr : ternary                          @sai_field(SAI_ACL_TABLE_ATTR_FIELD_SRC_IP);
     headers.ipv4.dst_addr : ternary                              @sai_field(SAI_ACL_TABLE_ATTR_FIELD_DST_IP);
     headers.ipv6.src_addr : ternary                              @sai_field(SAI_ACL_TABLE_ATTR_FIELD_SRC_IPV6);
     headers.ipv6.dst_addr : ternary                              @sai_field(SAI_ACL_TABLE_ATTR_FIELD_DST_IPV6);
     ttl : ternary                                                @sai_field(SAI_ACL_TABLE_ATTR_FIELD_TTL);
     dscp : ternary                                               @sai_field(SAI_ACL_TABLE_ATTR_FIELD_DSCP);
     ecn : ternary                                                @sai_field(SAI_ACL_TABLE_ATTR_FIELD_ECN);
     ip_protocol : ternary                                        @sai_field(SAI_ACL_TABLE_ATTR_FIELD_IP_PROTOCOL);
     headers.icmp.type : ternary                                  @sai_field(SAI_ACL_TABLE_ATTR_FIELD_ICMPV6_TYPE);
     local_metadata.l4_dst_port : ternary                         @sai_field(SAI_ACL_TABLE_ATTR_FIELD_L4_DST_PORT);
     local_metadata.ingress_port : optional                       @sai_field(SAI_ACL_TABLE_ATTR_FIELD_IN_PORT);
     headers.arp.target_proto_addr : ternary                      @composite_field(
                                                                    @sai_udf(base=SAI_UDF_BASE_L3, offset=24, length=2),
                                                                    @sai_udf(base=SAI_UDF_BASE_L3, offset=26, length=2)
                                                                  );
   }
   actions = {
     copy();
     trap();
     forward();
     acl_drop(standard_metadata);
     @defaultonly NoAction;
   }
   const default_action = NoAction;
   meters = acl_ingress_meter;
   counters = acl_ingress_counter;
   size = 128;
 }
...
```

Users can define custom ACLs in P4

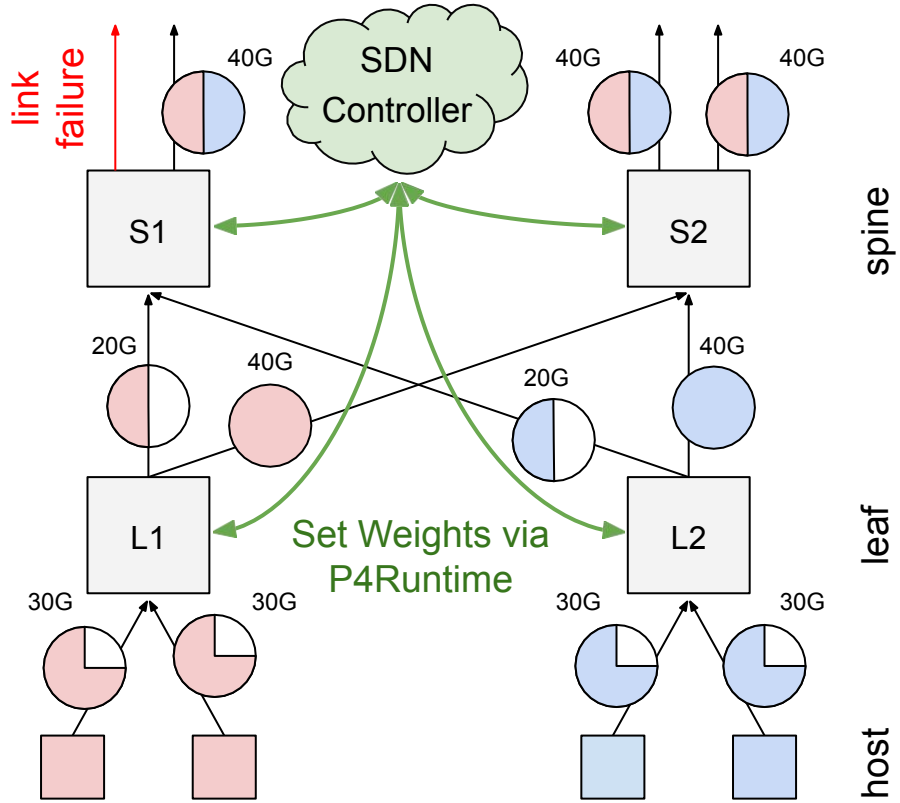
- Match fields
- Actions
- Counter
- Meters
- Table Size

P4 fields mapped to SAI using annotations

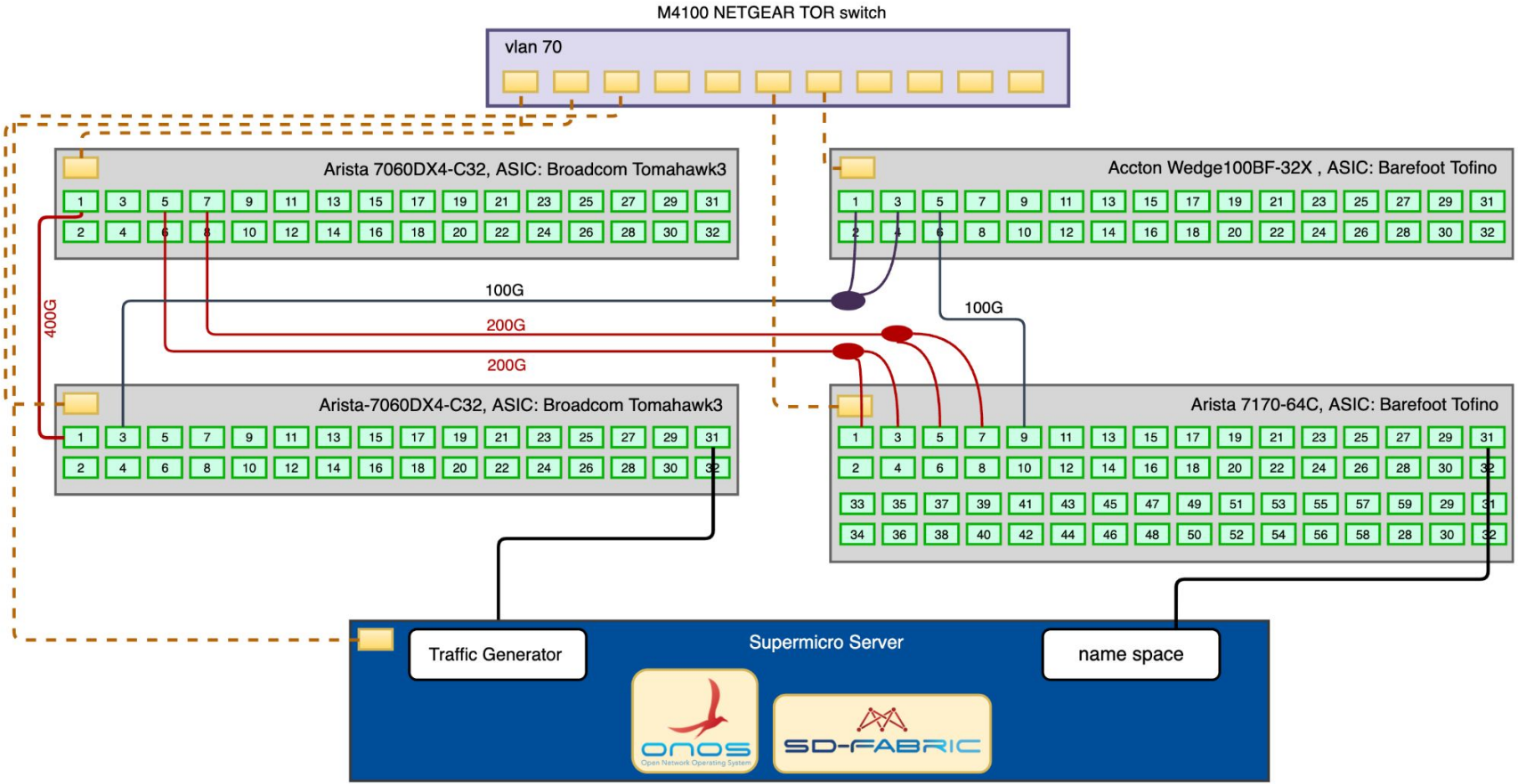
ACL tables are configured on the switch when the P4 pipeline is pushed via P4Runtime

Use Cases for SDN

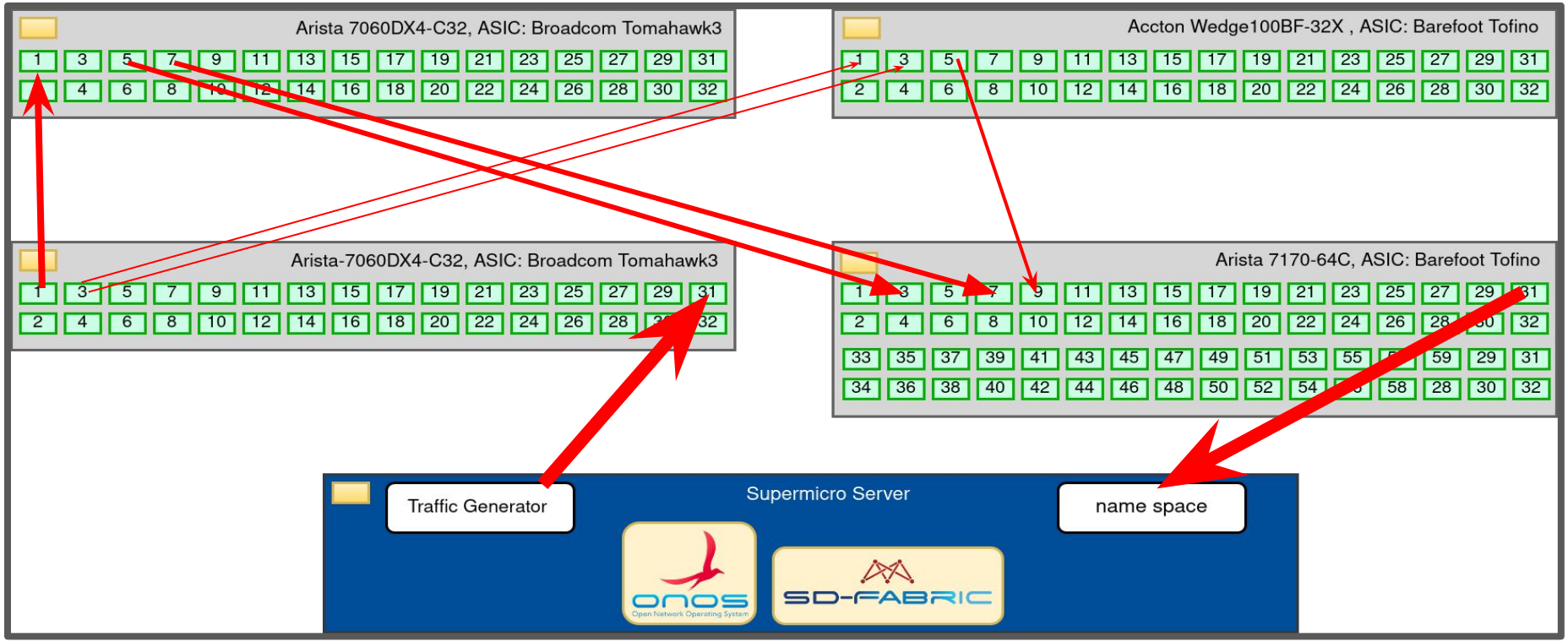
- Software Defined WAN
- Hitless Route Sequencing
- Inline Network Functions
  - Load balancers
  - Firewalls
  - Telemetry
- Inband Network Telemetry (INT)
- **U**nequal **C**ost **M**ulti**P**ath (UCMP, aka WCMP)
  - Focus of this demo



PINS Demo Topology



Weighted Cost Multipath



In Summary

- **PINS** brings SDN capabilities to **SONiC 202111** release
- **SD-Fabric** and **ONOS** are used to program the routing tables with WCMP
- Common, well-defined **P4** program enables interoperability across disparate hardware

PINS Working Group