ACL Enhancements on SONiC

Revision 2

Requirements 3

Scalability requirements 3

Assumptions 3

Limitations 3

References 3

SONiC system diagram for ACL configuration 4

Configuration flow for ACL 4

Current DB tables 4

Design changes 5

IPv6 ACL support 5

Egress ACL support 5

Binding ports to “any” 5

VLAN binding support 6

ACL show enhancement 6

The schema changes 6

ACL Tables Table 6

ACL Rules Table 6

Code changes 9

Class AclOrch 9

ACL Table Create or Delete 9

ACL Rule Create or Delete 9

Configuration file format 10

ACL status and stats 12

# Revision

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Revision | Author | Comment |
| 8/7/2017 | 0.1 | Zhenggen Xu |  |
|  |  |  |  |

# Requirements

1. Add IPv6 based ACL support with match conditions: Ether-Type, IP type, IP protocol, Src IPv6, Dst IPv6, L4 Src Port (with range support), L4 Dst Port (with range support), TCP flags and Interface port. Support priorities for the rules. Support Permit/Deny action. Support packet/byte counters. (phase 1)
2. Support binding ACL to “any” port (phase 2)
3. Support binding ACL to vlan (phase 2)
4. Support Egress ACL: both port binding and vlan binding (phase 2)
5. Show ACL status in DBs, HW etc (phase 2)

## Scalability requirements

Ingress ACL:

- 1K ACL rules for ipv4 only ACL

- If ipv6 ACL was enabled, system should support 512 rules for ipv4 and 512 rules for ipv6.

Egress ACL:

- 512 ACL rules for ipv4 only ACL

- If ipv6 ACL was enabled, system should support 256 rules for ipv4 and 256 rules for ipv6.

# Assumptions

1. IPv4 Ingress ACL is supported on SONiC.
2. SAI implementations for IPv6 ACL, VLAN based ACL and Egress ACL have been done by ASIC vendors.

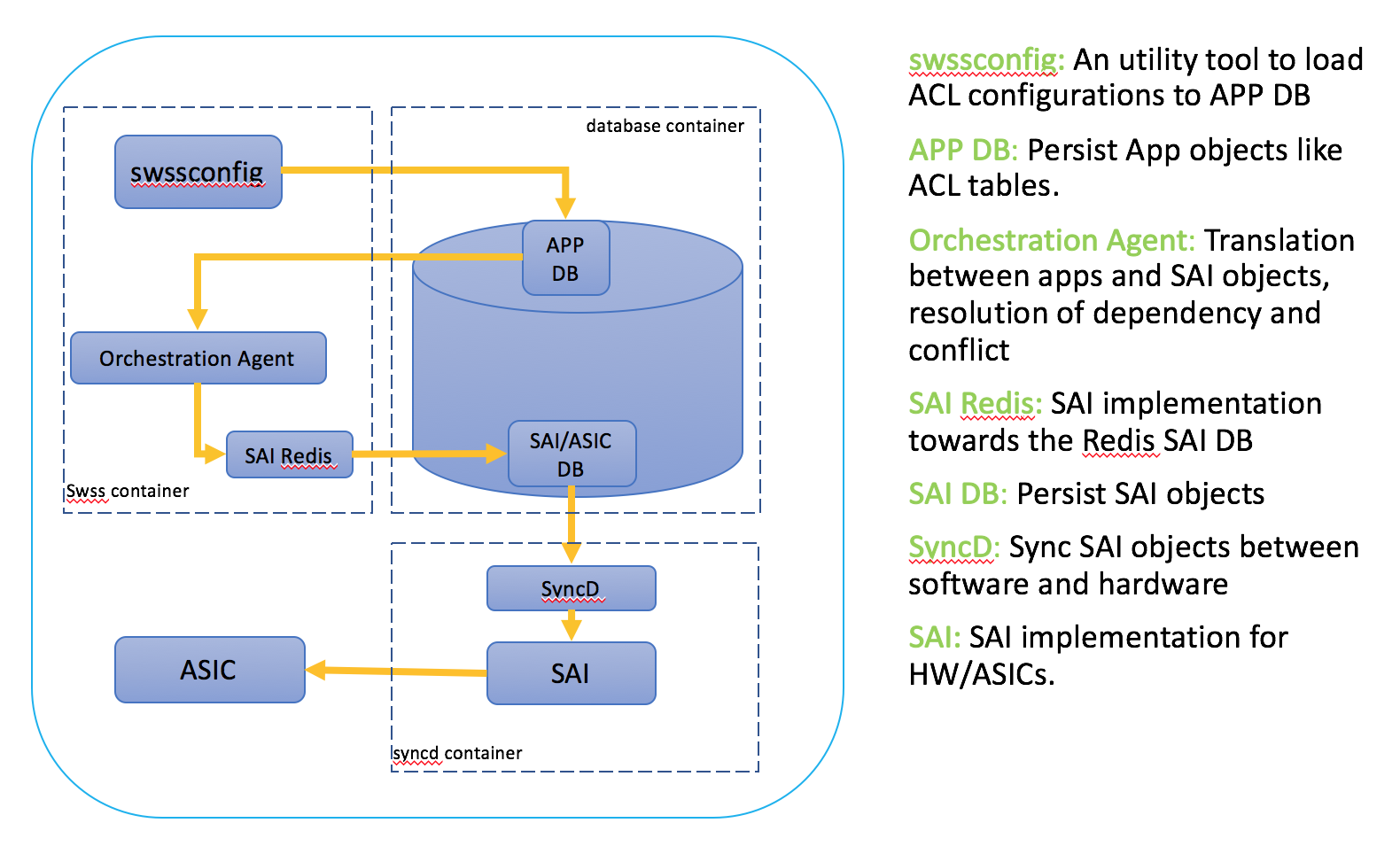
# Limitations

* For Egress ACL port binding, only one port can be specified.
* For VLAN binding, one vlan can be specified.
* This design document does not include configure DB design for ACL.

# References

https://github.com/Azure/SONiC/wiki/ACL-High-Level-Design

# SONiC system diagram for ACL configuration



### Configuration flow for ACL

We are using json files to prepare the ACL configurations. swssconfig is used to read the json-files and inject them into App DB.

There are multiple instances in the Redis DB, and APP DB is #0. The DB is schema less and holds key-value data. Orchestration agent is running in the docker “swss”. and listening on the APP DB changes, in ACL case, it is listening on the ACL tables(to be discussed later) in the APP DB for configuration changes, after some validation, it uses SAI Redis APIs to write the changes to SAI/ASIC DB (instance #1 of Redis). Syncd (running in “syncd” docker container) is subscribed to the ASIC DB table changes, and will get the updates and push the changes to libSAI towards ASIC.

### Current DB tables

ACL Table

ACL Tables is in App DB under the key ACL\_TABLE:table\_id.

table\_id is some string which will be specified by the user and should be unique across the App DB. table\_id is used to refer the table when adding rules and updating or deleting the table.

The table has the following properties:

policy\_desc - name of the ACL policy table description

type - table type which defines a list of supported matches that could be used in rules belonging to this table

port – binding port (list)

ACL Rule

ACLRules is in the App DB under the key ACL\_RULE\_TABLE:table\_id:rule\_id. table\_id is the table ID the rule belongs to and the rule\_id is some string which should be unique across the Table. rule\_id will be used to refer the Rule when it is needed to update or delete the Rule.

Rules have the following properties:

priority - rule priority in the table

match:value - packet properties this rule will match

action:value - action to be applied to the rule if match was successful

# Design changes

swssconfig is generic enough and no need to make changes to support new features in ACL.

## IPv6 ACL support

In the APP DB, to support Ipv6 ACL there are two ways:

1, Add a new ACL table to handle IPv6.

2, Expand the fields in the existing L3 table to support IPv6.

Solution 2 probably will reduce the existing Ipv4 ACL rule numbers even if we only enable ipv4 rules on the box. Solution 1 only reduces the ACL rules numbers if customers enable both Ipv4 and ipv6 ACLs. This design will use the solution 1.

This ACL table for ipv6 will use “L3V6” type. Details follow.

In the ACL rule table, we will introduce the fields to present IPv6 SRC and DST addresses.

## Egress ACL support

A stage field to support Ingress and Egress ACL is added to ACL table. And when we create the ACL objects in SAI DB/HW, stage will be used for the objects.

Checks will be done for egress ACL to allow only maximum one port for port binding.

## Binding ports to “any”

A value of “any” is added to the port field for applying the ACLs on all ports (wildcard the ports).

## VLAN binding support

To support VLAN binding, we add the binding type in the ACL table, and a vlan field to specify the VLAN.

## ACL show enhancement

ACL show CLI will be enhanced so that we know whether the tables/rules are in APP DB, ASIC DB or HW, also include the total size and free entries.

# The schema changes

for ACL tables and ACL rules table in APP DB as below (in-red):

## ACL Tables Table

ACL Table

key = ACL\_TABLE:name ; acl\_table\_name must be unique

;field = value

policy\_desc = 1\*255VCHAR ; name of the ACL policy table description

type = "mirror"/"l3/l3v6" ; type of acl table, every type of

; table defines the match/action a

; specific set of match and actions.

stage = "ingress/egress" ; ingress or egress stage

binding\_type = "port/vlan"

ports = [0-max\_ports]\*port\_name ; the ports to which this ACL

; table is applied, can be emtry

; value annotations

port\_name = 1\*64VCHAR ; name of the port, must be unique, could be “any”

max\_ports = 1\*5DIGIT ; number of ports supported on the chip

vlan = [1-4095] ; vlan value

## ACL Rules Table

ACL Rule

Defines rules associated with a specific ACL Policy

key: ACL\_RULE\_TABLE:table\_name:seq ; key of the rule entry in the table,

; seq is the order of the rules

; when the packet is filtered by the

; ACL "policy\_name".

; A rule is always assocaited with a

; policy.

;field = value

action = "permit"/"deny"/"mirror" ; action when the fields are

; matched (mirror action only

; available to mirror acl table

; type)

mirror\_session\_name = mirror\_session\_name ; refer to the mirror session

; (only available to mirror acl

; table type)

l2\_prot\_type = "ipv4/ipv6" ; options of the l2\_protocol\_type

; field.

l3\_prot\_type = "icmp"/"tcp"/"udp"/"any" ; options of the l3\_protocol\_type

; field

ipv4\_src = ipv4\_prefix/"any" ; options of the source ipv4

; address (and mask) field

ipv4\_dst = ipv4\_prefix/"any" ; options of the destination ipv4

; address (and mask) field

; l2\_prot\_type detemines which

; set of the addresses taking

; effect, v4.

ipv6\_src = ipv6\_prefix/"any" ; options of the source ipv6

; address (and mask) field

ipv6\_dst = ipv6\_prefix/"any" ; options of the destination ipv6

; address (and mask) field

; l2\_prot\_type detemines which

; set of the addresses taking

; effect, v6.

l4\_src\_port = port\_num/[port\_num\_L-port\_num\_H] ; source L4 port or the

; range of L4 ports field

l4\_dst\_port = port\_num/[port\_num\_L-port\_num\_H] ; destination L4 port or

; the range of L4 ports

; field

;value annotations

seq = DIGITS ; unique sequence number of the rules assocaited

; within this ACL policy. When applying this ACL

; policy, the seq determines the order of the

; rules applied.

port\_num = 1\*5DIGIT ; a number between 0 and 65535

port\_num\_L = 1\*5DIGIT ; a number between 0 and 65535,

; port\_num\_L < port\_num\_H

port\_num\_H = 1\*5DIGIT ; a number between 0 and 65535,

; port\_num\_L < port\_num\_H

ipv6\_prefix = 6( h16 ":" ) ls32

/ "::" 5( h16 ":" ) ls32

/ [ h16 ] "::" 4( h16 ":" ) ls32

/ [ \*1( h16 ":" ) h16 ] "::" 3( h16 ":" ) ls32

/ [ \*2( h16 ":" ) h16 ] "::" 2( h16 ":" ) ls32

/ [ \*3( h16 ":" ) h16 ] "::" h16 ":" ls32

/ [ \*4( h16 ":" ) h16 ] "::" ls32

/ [ \*5( h16 ":" ) h16 ] "::" h16

/ [ \*6( h16 ":" ) h16 ] "::"

h16 = 1\*4HEXDIG

ls32 = ( h16 ":" h16 ) / IPv4address

ipv4\_prefix = dec-octet "." dec-octet "." dec-octet "." dec-octet “/” %d1-32

dec-octet = DIGIT ; 0-9

/ %x31-39 DIGIT ; 10-99

/ "1" 2DIGIT ; 100-199

/ "2" %x30-34 DIGIT ; 200-249

The list of allowed matches and actions depends on the table the rule is associated.

ACL table with type “L3V6” has follow properties:

Table 1: Matches allowed in the table of the type "L3V6"

|  |  |  |
| --- | --- | --- |
| Keyword for the match criteria | Type | Description |
| ETHER\_TYPE | uint16\_t | Hexadecimal integer [0..FFFF]  (with or w/o leading "0x") |
| IP\_TYPE | string | One of: "IPv4"/"NON\_IPv4"/"ARP" |
| IP\_PROTOCOL | uint8\_t | Hexadecimal unsigned integer [0..FF] |
| SRC\_IPV6 | ip\_address | A valid IPv6 subnet in format IP/Mask |
| DST\_IPV6 | ip\_address | A valid IPv6 subnet in format IP/Mask |
| L4\_SRC\_PORT | uint16\_t | Decimal unsigned integer [0..65535] |
| L4\_DST\_PORT | uint16\_t | Decimal unsigned integer [0..65535] |
| TCP\_FLAGS | uint8\_t | Hexadecimal unsigned integer [0..FF] |
| L4\_SRC\_PORT\_RANGE | uint16\_t, uint16\_t | Two dash separated decimal unsigned integers [0..65535] |
| L4\_DST\_PORT\_RANGE | uint16\_t, uint16\_t | Two dash separated decimal unsigned integers [0..65535] |

Table 2: Actions allowed in the table of the type "L3V6"

|  |  |  |
| --- | --- | --- |
| Keyword for the action | Type | Description |
| PACKET\_ACTION | string | Packet action value: "FORWARD" or "DROP" |

Orchestration agent needs to be updated in order to support ACL new features in the AppDB and the SAI API. The class AclOrch will be enhanced to handle this.

Orchestration Agent will process basing on App DB changes, and also apply the validation logic

# Code changes

## Class AclOrch

We will use the same Class AclOrch to process the updates from the ACL tables, and validate the data (App DB) and transfer the ACL data from App DB to ASIC DB via SAI Redis API. Also caching the ACL objects for deletion/update and dump.

## ACL Table Create or Delete

ACL table with SRC\_IPV6 and DST\_IPV6 fields instead of “SRC\_IP” and “DST\_IP” will be created. This is done in AclOrch::createBindAclTable() which was called by AclOrch::doAclTableTask().

The Stage field will be taken into account when creating objects in ASIC DB and HW. Also additional validations are done for Egress ACL, e,g, port can only have one for egress ACL.

Code needs to be added to handle binding type, in case of vlan binding, the relevant code will be added to bind the ACL to vlan.

For port binding to “any”, we will basically skip the port binding so port will be wild-carded.

## ACL Rule Create or Delete

AclOrch:: doAclRuleTask () is called to process the changes from ACL rule table. To support Ipv6 ACL, for this new table type “L3V6”, we will validate the table type along with the user input of ip address fields.

On table create, AclOrch will verify if the rule already exists (using rule\_id), if so, it processed as update. Regular create/delete will update the internal class structures and create/delete the appropriate SAI objects.

# Configuration file format

- Valid json file. The file should be in the format swssconfig can process. This assumes lists surrounded by square brackets, dictionaries with curly brackets (braces), tuples inside dictionary separated with semicolon and enumerated elements separated with the comma.

- Logical consistency. The configuration provided should be complete. Rules should not refer non-existing tables, etc.

- Order: Tables should appear before Rules.

- Rules should have at least one match criteria and one action

- List of ports to bind to the table should contain physical port names or “any”.

- Egress ACL with port binding only allow one physical port.

- ACL with vlan binding allows one valid vlan value (1-4095)

- Maximum number of rules allowed: 1K rules total in the all "L3" + "L3V6" ingress tables, 512 rules total in the all "L3" + "L3V6" Egress tables,

#### Local configuration file Format:

Local json file example for IPv6 ACL:

Acl-table.json

[

{

"ACL\_TABLE:data-aclv6":{

"policy\_desc":"data-aclv6",

"ports":"Ethernet16,Ethernet17,Ethernet18",

“binding\_type”: “port”

"stage":"Ingress",

"type":"L3V6"

},

"OP":"SET"

}

]

acl-rules.json

[

{

"ACL\_RULE\_TABLE:data-aclv6:Rule\_4":{

"PACKET\_ACTION":"FORWARD",

"SRC\_IPV6":"2001:db8:bad:a55::0/64",

"DST\_IPV6":"2001:db8:bad:a66::1/128",

"priority":4000

},

"OP":"SET"

}

]

[

{

"ACL\_RULE\_TABLE:data-aclv6:Rule\_3":{

"PACKET\_ACTION":"DROP",

"SRC\_IPV6":"2001:db8:bad:a55::0/64",

"priority":3000

},

"OP":"SET"

}

]

[

{

"ACL\_RULE\_TABLE:data-aclv6:Rule\_2":{

"IP\_TYPE" : "IPv6ANY",

"PACKET\_ACTION":"FORWARD",

"priority":2000

},

"OP":"SET"

}

]

[

{

"ACL\_RULE\_TABLE:data-aclv6:Rule\_1":{

"IP\_TYPE" : "IPv6ANY",

"PACKET\_ACTION":"DROP",

"priority":1000

},

"OP":"SET"

}

]

#### openconfig file

SONiC also supports openconfig module for ACL, the example can be found here:

<https://github.com/Azure/sonic-buildimage/blob/023a5b9714c8d2c41ab4469beae3569c0e43933a/src/sonic-config-engine/tests/t0-sample-acl.json>

This openconfig format can be used by user, and then on SONiC a translation script: translate\_acl will be used to translate it to local json file to be loaded by swssconfig.

IPv6, VLAN and Egress ACL support need to be considered in the openconfig and parser.

# ACL status and stats

ACL counters are saved to Redis DB instance #2 with a predefined interval, the default interval is 10 seconds, and it is hard coded for now.

An ACL show script was provided to show ACL properties and counters, it will be integrated to the “show” command structure:

Example output: “show acl”

RULE NAME TABLE NAME TYPE STAGE BINDING BINDINGVALUE PRIO ACTION PACKETS COUNT BYTES COUNT

----------- ------------ ------ ------ -------- --------------- ------------- ------ ---- ----

Rule\_4 data-aclv6 L3V6 Ingress PORT Ethernet0, Ethernet4 4000 FORWARD 41 5166

Rule\_3 data-aclv6 L3V6 Ingress PORT Ethernet0, Ethernet4 3000 DROP 10 1188

Rule\_2 data-aclv6 L3V6 Ingress PORT Ethernet0, Ethernet4 2000 FORWARD 53 6278

Rule\_4 dataacl L3 Ingress PORT Ethernet0, Ethernet4 4000 FORWARD 23 2438

Rule\_3 dataacl L3 Ingress PORT Ethernet0, Ethernet4 3000 DROP 14 938

Rule\_2 dataacl L3 Ingress PORT Ethernet0, Ethernet4 2000 FORWARD 54 10408

“show acl rule details” will include the match conditions as well:

RULE NAME TABLE NAME TYPE STAGE BINDING BINDINGVALUE MATCH-CONDITIONS

PRIO ACTION PACKETS COUNT BYTES COUNT

----------- ------------ ------ ------ -------- --------------- ------------- ------ ---- ---- ----- -------

Rule\_4 data-aclv6 L3V6 Ingress PORT Ethernet0, Ethernet4 SRC\_IPV6:"2001:db8:bad:a55::0/64 DST\_IPV6:"2001:db8:bad:a66::1/128" 4000 FORWARD 41 5166

“show acl rule status” will show the ACL status whether the ACL is in APP DB, ASIC DB or HW.

RULE NAME TABLE NAME TYPE APPDB ASICDB HW

----------- ------------ ------ ------ -------- --------------- ------------- ------ ---- ----

Rule\_4 data-aclv6 L3V6 Yes Yes Yes

Rule\_3 data-aclv6 L3V6 Yes Yes Yes

Rule\_2 data-aclv6 L3V6 Yes Yes Yes

Rule\_4 dataacl L3 Yes Yes Yes

Rule\_3 dataacl L3 Yes Yes Yes

Rule\_2 dataacl L3 Yes Yes Yes

“show acl table hw” is proposed to show the ACL table in HW for debug purpose with table size and free entry numbers.

TABLE NAME TYPE Total Free

----------- ------------ ------ ------

data-aclv6 L3 512 509

data-acl L3V6 512 509