ABF (ACL Based Forwarding) on ASR9K

1 Introduction

In today's high performance internetworks, organizations need the freedom to implement packet forwarding and routing according to their own defined polices in a way that goes beyond traditional routing protocol concerns. In XR platform this is achieved by using ACL Based Forwarding (ABF). ABF is a subset of PBR (Policy Based Routing) infrastructure in the XR platform. In summary this features allows traffic matching specific ACL rule to be forwarded to user specified nexthop instead of route selected by a routing protocol. ASR9K supports ABF v4 and ABF v6 for forwarding IPv4 and IPv6 traffic respectively using ABF rules. Note that ABF is an ingress only feature, i.e., traffic matching ingress ACL can be forwarded using ABF.

1.1 The Benefits of ACL Based Forwarding

- Source-Based Transit Provider Selection – Internet service providers and other organizations can use ABF to route traffic originating from different sets of users through different internet connections across the policy routers.
- Cost Savings – Organizations can achieve cost savings by distributing interactive and batch traffic among low-bandwidth, low-cost permanent paths and high-bandwidth, high-cost, switched paths.
- Load Sharing – In addition to the dynamic load-sharing capabilities offered by destination-based routing that the Cisco IOS-XR software provides network manager can implement policies to distribute traffic among multiple paths based on the traffic characteristics.

1.2 ABF Configuration

In XR platforms ABF is supported using ACL infrastructure. Enhancements are done to allow specification of ABF nexthop in the ACE rule. There are different variants of specifying ABF nexthop in a ACE. Following illustrates different options for specifying ABF nexthops.

1.2.1 **ABF Nexthop:**

We can specify list of specified IP addresses (up to 3 IP addresses will be supported for ABF along with VRF) in the ACE rule. In this case each nexthop IP address can specify the next hop router in the path towards the destination to which the packets must be forwarded. The first IP address associated with a currently "up" connected interface will be used to route the packets. Following is the example

ABFv4 Example :

ipv4 access-list abf-1

10 permit ipv4 any 100.100.100.0/24 nexthop1 VRF RED ipv4 1.1.1.1 nexthop2 VRF BLUE ipv4 2.2.2.2 nexthop3 ipv4 3.3.3.3

ABFv6 Example :

Ipv6 access-list abf-2

10 permit ipv4 any 2010::1/64 nexthop1 VRF A ipv6 2020::1 nexthop2 VRF B ipv6 2030::1 nexthop3 VRF C ipv6 2040::1

In the above example, there are 3 nexthops specified in the ACE rule. nexthop1 has the highest priority and nexthop3 has the lowest priority. if nexthop1 is reachable, i.e, there exist a route for the nexthop1 then it will be selected for forwarding the traffic matching the ACE. Nexthop2 is only selected if nexthop 1 is not reachable, i.e., there is no route present for the nexthop1 and nexthop2 is reachable, i.e., nexthop2 has a route and so on.

1.2.2 **ABF Default Route**

List of default IP addresses (up to 3 IP addresses will be supported for ABF) – IP address can specify the next hop router in the path towards the destination to which the packets must be forwarded, if there is no explicit route for the destination address of the packet in the routing table. The packet DA lookup should result in default-route. Under this condition ABF will be used to select the nexthop. The first IP address associated with a currently "up" connected interface will be used to route the packets. Following is the example for the same. Please note "default" keyword in the ACE rule.

ABFv4 Example:

ipv4 access-list abf-1

10 permit ipv4 any 100.100.100.0/24 **default** nexthop1 VRF RED ipv4 1.1.1.1 nexthop2 VRF BLUE 2.2.2.2 nexthop3 VRF GREEN 3.3.3.3

ABFv6 Example:

ipv6 access-list abf-2

10 permit ipv4 any 2010::1/64 **default** nexthop1 VRF A ipv6 2020::1

In the above example if matching traffic does not have any route present for the packet DA and it has only default route, in that case ABF nexthop1 is selected for forwarding. If None of the ABF nexthops are UP then packet is routed using default route as expected. Note that if Packet DA has route learnt by routing protocol in that case ABF default Nexthop are

not selected, instead packet is forwarded using route for packet DA and incoming interface VRF. Lets illustrate this with an example:

In the above example for ABFv4, lets says incoming packet DA is 100.100.100.100

- If 100.100.100.100 lookup results in default path, then ABF nexthop is selected in the order of priority when UP, where nexthop1 has highest priority and nexthop3 has lowest priority.

- If 100.100.100.100 has a non-default route, then packet will be forwarded using packet DA lookup, ABF path is not taken in this case.

- If 100.100.100.100 has default route and none of the ABF nexthops are UP then packet will be forwarded to the default route.

1.2.3 **ABF VRF select**

This allows to specify only next hop VRF instead of specific IP address for the nexthop. With this packet DA lookup will be performed in the ABF Nexthop VRF. If route is present then packet is forwarded otherwise dropped.

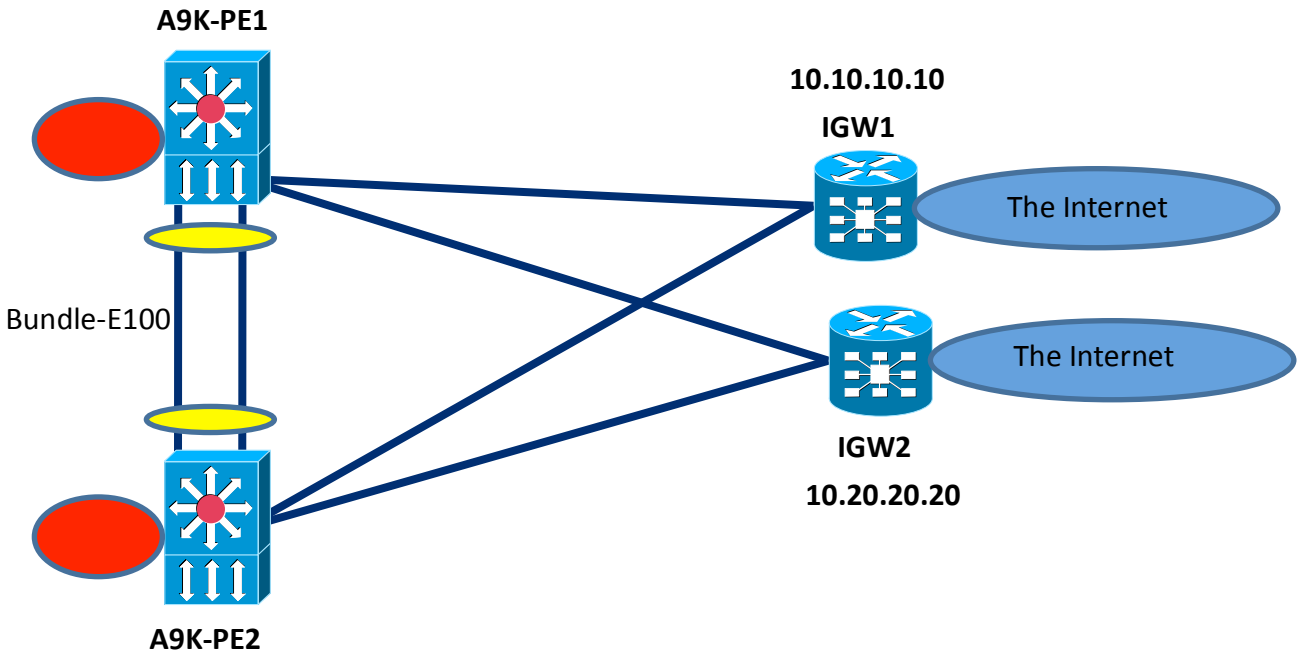
ABFv4 Example:

```
ipv4 access-list abf-1
10 permit ipv4 any 100.100.100.0/24 nexthop1 VRF RED
```

1.2.4 **ABF with track option**

Track option in ABF enables track object to be specified along with nexthop VRF and IP address. Status of track object is used along with reachability of the ABF Nexthop to determine if corresponding nexthop should be selected or not for forwarding. At present ABFv4 is supported with track option. ABFv6 track option will be supported from 5.1.0 release onwards.

The following simple topology will be used to show this interaction.



1.3 Description

From the above diagram the requirement is to have traffic coming into the RED interfaces/VLANs/VRFs on **A9K-PE1** to be forwarded with ABF to **IGW1,** if IGW1 is to fail or not become reachable, the traffic should then get forwarded to IGW2, the same requirement is present on **A9K-PE2**. There might be multiple Internet gateways in this network, so it's important to associate each PE with the primary and secondary IGW. In the examples below we will use the loopback addresses of the IGWs, equally one can use any IP address on the GW. It some designs it might be more suitable to use the IP address of the internet facing interface.

1.4 Solution

1.4.1 Configuring IPSLA

Pre-requisite for IPSLA is the MGBL pie, without this pie the IPSLA CLI will not be available, this is only required on the PEs, as the IGW is acting as a responder. Step one is to configure IPSLA tracking towards the IGWs from the PEs, this is an example from **A9K-PE1**.

```
RP/0/RSP0/CPU0:A9K-PE1#sh run ipsla
Thu Mar 28 21:54:21.934 UTC
ipsla
 operation 1
  type icmp echo
   destination address 10.10.10.10
```

```
  frequency 5
 !
!
operation 2
 type icmp echo
  destination address 10.20.20.20
  frequency 5
 !
!
reaction operation 1
 react timeout
  action logging
 !
!
reaction operation 2
 react timeout
  action logging
 !
!
schedule operation 1
 start-time now
 life forever
!
schedule operation 2
 start-time now
 life forever
!
!

RP/0/RSP0/CPU0:A9K-PE1#
```

1.4.2 **Verify the IPSLA operation with show ipsla statistics**

```
RP/0/RSP0/CPU0:A9K-PE1#show ipsla statistics
Thu Mar 28 21:54:53.052 UTC
Entry number: 1
   Modification time: 21:19:42.937 UTC Thu Mar 28 2013
   Start time      : 21:16:55.158 UTC Thu Mar 28 2013
   Number of operations attempted: 456
   Number of operations skipped  : 0
   Current seconds left in Life  : Forever
   Operational state of entry    : Active
   Operational frequency(seconds): 5
   Connection loss occurred      : FALSE
   Timeout occurred              : FALSE
   Latest RTT (milliseconds)     : 2
```

```
   Latest operation start time   : 21:54:50.367 UTC Thu Mar 28 2013
   Next operation start time     : 21:54:55.367 UTC Thu Mar 28 2013
   Latest operation return code  : OK
   RTT Values:
    RTTAvg  : 2        RTTMin: 2        RTTMax : 2
    NumOfRTT: 1        RTTSum: 2        RTTSum2: 4

Entry number: 2
   Modification time: 21:46:26.132 UTC Thu Mar 28 2013
   Start time      : 21:46:26.116 UTC Thu Mar 28 2013
   Number of operations attempted: 102
   Number of operations skipped  : 0
   Current seconds left in Life  : Forever
   Operational state of entry    : Active
   Operational frequency(seconds): 5
   Connection loss occurred      : FALSE
   Timeout occurred              : FALSE
   Latest RTT (milliseconds)     : 5
   Latest operation start time   : 21:54:51.325 UTC Thu Mar 28 2013
   Next operation start time     : 21:54:56.325 UTC Thu Mar 28 2013
   Latest operation return code  : OK
   RTT Values:
    RTTAvg  : 5        RTTMin: 5        RTTMax : 5
    NumOfRTT: 1        RTTSum: 5        RTTSum2: 25
RP/0/RSP0/CPU0:A9K-PE1#
```

1.5 Configuring Tracking

The next step is to configure tracking, so to have IPSLA associated with the tracker. Tracked objects could be Line protocol, Routes, IPSLA reachability, lists etc.. In below configuration we are associating the above IPSLA operation 1 through rtr 1.

```
RP/0/RSP0/CPU0:A9K-PE1#sh run track
Thu Mar 28 21:33:44.466 UTC
track IGW1
 type rtr 1 reachability
!
track IGW2
 type rtr 2 reachability
!

RP/0/RSP0/CPU0:A9K-PE1#
```

1.5.1 Verify the track operation status:

```
RP/0/RSP0/CPU0:A9K-PE1#show track brief
```

```
Thu Mar 28 21:55:53.004 UTC
Track                Object                    Parameter      Value
-------------------------------------------------------------------------------------------------
IGW2                 IPSLA Operation 2         reachability   Up
IGW1                 IPSLA Operation 1         reachability   Up
RP/0/RSP0/CPU0:A9K-PE1#
```

Configuring ABF on IOS XR

The next step is to associate the Tracker with the ABF statement; it's done through the standard ABF CLI. The nexthop ip address does not need to be the same as the tracked ip address, as showing in this example.

```
RP/0/RSP0/CPU0:A9K-PE1#sh run ipv4 access-list
Thu Mar 28 21:50:23.672 UTC
ipv4 access-list ABF_IGW
10 permit ipv4 any any nexthop1 track IGW1 ipv4 10.10.10.1 nexthop2 track IGW2 ipv4 10.20.20.2
!
```

Bind the ABF statement to an ingress interface.

```
RP/0/RSP0/CPU0:A9K-PE1#sh run int bundle-e1
Thu Mar 28 21:50:28.751 UTC
interface Bundle-Ether1.100
 ipv4 address 192.168.100.1 255.255.255.0
ipv4 access-group ABF_IGW ingress
!

RP/0/RSP0/CPU0:A9K-PE1#
```

Notes, the above configuration can be applied to an interface in a VRF or a global routing table.

2 ABF Support Matrix

Following provide the support matrix for ABF IPv4 and ABF IPv6 on different linecards on the ASR9k platform. Classification is done based on combination of different interface types on ingress and egress linecards for ABFv4 and ABFv6. Aforementioned, ABF is ingress only feature, i.e., ABF rule is always applied on the ingress interface. TCAM resources are only allocated on line ingress cards where ABF is configured which is used for classification of incoming traffic. For virtual interface such as GRE TCAM resources will be allocated on all LCs where virtual interface is present. In the following figure ingress LC determines that packet will be routed using ABF rule. Packet will be sent to egress LC with ABF type in the fabric header, which will be used by Egress LC to route packet based on ABF nexthop instead of packet DA. As a result both ingress and egress LCs need to be aware of the ABF feature that can be supported on a particular LC.

Ingress Egress

2.1 Line Cards Support

ABFv4 is supported on Ethernet, Enhanced Ethernet, and SIP-700 Linecards. ABFv6 is supported only on Enhanced Ethernet and SIP-700. Following support matrix provide the different combinations of ingress and egress LCs that can be supported for ABFv4 and ABFv6 with different interface types.

- **Ethernet: ETH**
- **Enhanced Ethernet: E-ETH**
- **SIP-700: SI**

2.1.1 Enhanced Ethernet as Ingress LC

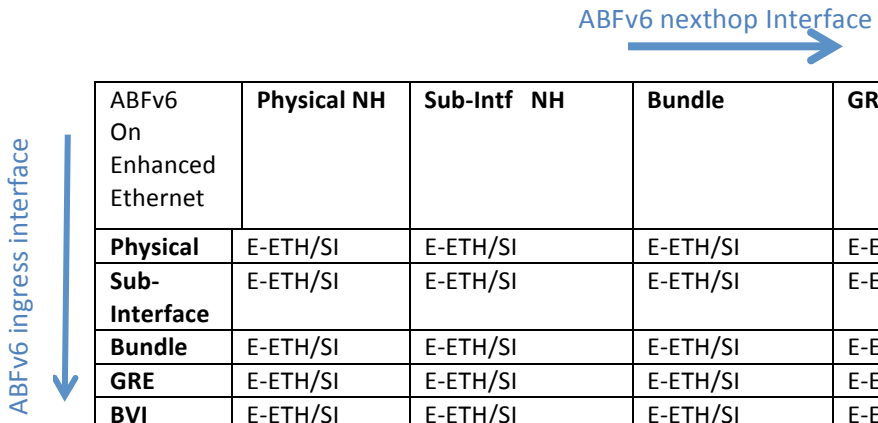
ABFv4

ABFv4 nexthop Interface →

ABFv4 ingress Interface ↓

ABFv4 On Enhanced Ethernet LC	**Physical NH LC**	**Sub-Intf NH LC**	**Bundle NH LC**	**GRE NH LC**	**BVI NH LC**	**PW-HE NH LC**
Physical	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH	E-ETH
Sub-Interface	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH	E-ETH
Bundle	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH	E-ETH
GRE	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH	E-ETH
BVI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH	E-ETH
PW-HE	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH	E-ETH

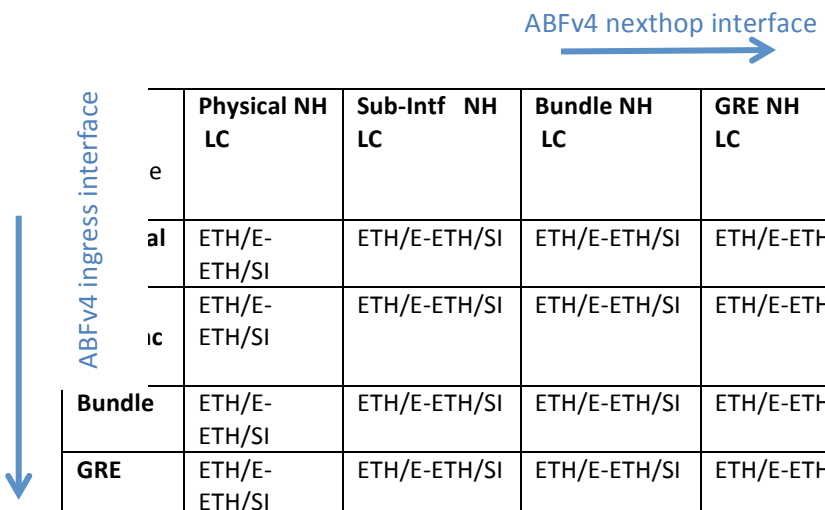
2.1.1.1 ABFv6



ABFv6 On Enhanced Ethernet	Physical NH	Sub-Intf NH	Bundle	GRE	BVI	PW-HE
Physical	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH	E-ETH
Sub-Interface	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH	E-ETH
Bundle	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH	E-ETH
GRE	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH	E-ETH
BVI	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH	E-ETH
PW-HE	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH	E-ETH

2.1.2 Ethernet as Ingress LC

2.1.2 BFv4



e	Physical NH LC	Sub-Intf NH LC	Bundle NH LC	GRE NH LC	BVI NH LC	PW-HE NH LC
al	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	Not supported	Not supported
ıc	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	Not supported	Not supported
Bundle	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	Not supported	Not supported
GRE	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	Not supported	Not supported
BVI	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
PW-HE	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported

2.1.2.2 ABFv6 (not supported on Ethernet LC as ingress or egress)

For ABFv6 Ethernet LC cannot be ingress LC or Egress LC, i.e., we can not configure ABFv6 on Ethernet LC interfa\(\)ABFv6 nexthop cannot be Ethernet LC interface.

ABFv6 nexthop interface

ABFv6 ingress interface

	Physical NH	Sub-Intf NH	Bundle	GRE	BVI	PW-HE
	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
GRE	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
BVI	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
PW-HE	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported

2.1.3 SIP-700 as Ingress LC

2.1.3 ABFv4

ABFv4 nexthop interface

ABFv4 ingress interface

	Physical NH LC supported	Sub-Intf NH LC supported	Bundle NH LC supported	GRE NH LC supported	BVI NH LC supported	PW-HE NH LC supported
	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	Not supported	Not supported
	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	Not supported	Not supported
	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	Not supported	Not supported
	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	ETH/E-ETH/SI	Not supported	Not supported
BVI	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
PW-HE	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported

2 ABFv6

ABFv6 nexthop interface

ABFv6 ingress interface

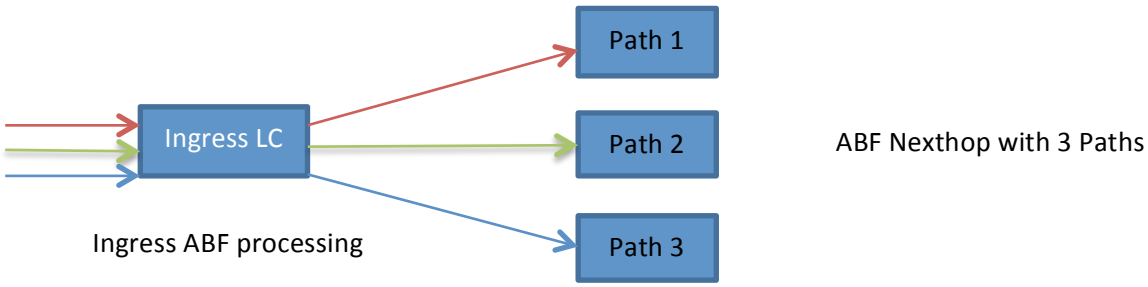
5 0	Physical NH LC supported	Sub-Intf NH LC supported	Bundle NH LC supported	GRE NH LC supported	BVI NH LC supported	PW-HE NH LC supported
cal	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH/SI	Not supported	Not supported
ace	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH/SI	Not supported	Not supported
le	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH/SI	Not supported	Not supported
	E-ETH/SI	E-ETH/SI	E-ETH/SI	E-ETH/SI	Not supported	Not supported
BVI	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported
PW-HE	Not supported	Not supported	Not supported	Not supported	Not supported	Not supported

2.2 Scale Support

We have tested 1000 unique ABF nexthop (IPv4 and IPv6 combined). Note that this does not include duplicate nexthops, we can use same nexthops across multiple ACLs, ACEs and it will be counted once towards the supported scale. System does not enforce this limitation through configuration, user can configure more than 1000 unique nexthops but any such use-case should be tested thoroughly by the customer before the deployment.

2.3 Load Balancing

If ABF nexthop has multiple ECMP path then multiple streams matching the corresponding ACE rule will get load balanced across the ECMP paths. The raw-hash will be computed on the ingress LC based on the incoming packet and that will be used to select the outgoing path for the ABF nexthop like regular IPv4 or IPv6 forwarding. This will ensure that all the packets from a given stream will always take the same path when packet is routed using ABF. As a result for multiple streams using same ABF nexthop we should expect to see load balancing across multiple ECMP paths. Figure 2 shows load balancing for 3 incoming streams matching same ACE rule with ABF nexthop.



2.4 Troubleshooting

- **Verify ABF is programmed correctly in the hardware.**

RP/0/RSP0/CPU0:ios#show access-lists ipv4 abf-1 hardware ingress location 0/1/cpu0

Mon May 13 06:06:58.609 UTC

ipv4 access-list abf-1

10 permit ipv4 any 20.20.20.20 (next-hop: addr=**40.40.40.2**, vrf name=**default**)

If nexthop is programmed in the hardware then verify that nexthop is resolved, you can confirm that by pinging nexthop.

- **Verify that traffic is matching the ACL for which ABF rule need to be applied.**

ABF rule is applied if the traffic matches corresponding ACE entry

RP/0/RSP0/CPU0:ios#show access-lists ipv4 abf-perf-1 hardware ingress location$

Mon May 13 06:12:16.203 UTC

ipv4 access-list abf-perf-1

10 permit ipv4 any 20.20.20.20 (**294096734 hw matches**) (next-hop: addr=40.40.40.2, vrf name=default)

Note that if incoming traffic is IPv4 or IPv6 then that will be only matched with ACL rules. However, if incoming traffic is labeled traffic then that traffic won't match ACL rules and hence no ABF forwarding will take place for that scenario.

- **Verify that Ingress and Egress interface and line card are supported as per support matrix given in Section 2**

  For e.g. ABFv6 is not supported on Ethernet Linecard. Ingress and Egress cannot be Ethernet LC for ABF v6.

- **Verify that ABF NH is pingable and is in resolved state.**

If ABF is programmed correctly in the hardware and traffic is also matching the ACE. Verify that ABF NH is in resolved state. Confirm that you are able to ping ABF nexthop which is programmed in the hardware.

- ## Verify path using traceroute

Traceroute can be used to determine the end-to-end path taken by the packet stream. If ABF match happen for the traceroute probe packet then you should expect to see ABF nexthop in the traceroute output.

Full credit goes to Tarun Banka for authoring this paper
