ECMP Convergence Acceleration in sonic for i/f down events

Table of Contents

[1 Document History](#_30j0zll)

[2 Abbreviations](#_1fob9te)

[3 References](#_3znysh7)

[4 Problem Overview](#_2et92p0)

[5 Proposed Solutions](#_tyjcwt)

[5.1 BGP next hop tracking](#_3dy6vkm)

[5.2 Removing the ecmp path](#_1t3h5sf)

[5.3 Marking the ecmp path as stale/non-forwarding](#_4d34og8)

[6 Dependencies](#_2s8eyo1)

[7 CLI](#_17dp8vu)

# Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description |
| A | 06/06/2017 | Nikos | Initial version |
|  |  |  |  |
|  |  |  |  |

# Abbreviations

|  |  |
| --- | --- |
| **Term** | **Definition** |
| BGP | Border Gateway Protocol |
| ECMP | Equal Cost Multi Path |
| IGP | Interior Gateway Protocol |
| RIB | Routing Information Base |

# References

# Problem Overview

It is standard practice in today’s networks to distribute traffic across multiple paths. This is achieved either statically or dynamically with protocols such as BGP, or IGPs. By default, BGP will choose only one path during its bestpath selection but in today’s data centers, network nodes are connected to multiple other nodes with the intend to provide increased bandwidth as well as multiple equal cost paths for redundancy purposes and load balancing. As such, BGP is configured to take advantage of the multiple available paths and selects them, if available, for a prefix during bestpath calculation for installation into the RIB and forwarding.

When one of those paths becomes unavailable due to an interface going down or administratively shut, BGP will either detect that or be notified of changes in its next hops. It will then determine which paths are affected by this change, usually via a walk of the prefix table, and for each prefix it will rerun its bestpath selection algorithm and modify the ecmp paths calculated for that prefix accordingly. It will then proceed downloading all the affected prefixes into RIB which in turn it will download into sonic.

This process is not instantaneous and until BGP converges from such an event and the hardware is updated, forwarding is still using the unavailable path which results in black holing traffic partially for any prefix that makes use of that path.

It is therefore desirable to reduce significantly the amount of time traffic is being dropped over the unavailable path and accelerate at a lower level the ecmp groups convergence by removing from forwarding any paths being affected due to the interface state change.

# Proposed Solutions

Before outlining the two proposed solutions, it is important to understand the different ways BGP may track next hop changes and how it reacts to them. This will provide more insight as to what advantages and disadvantages each solution has and decide what is best suited for the current sonic architecture and data center deployments as well as what may be desirable in the future.

## BGP next hop tracking

Different BGP implementations, may have different ways of tracking and validating BGP next hops.

Older BGP implementations are using a polling/periodic mechanism. In those cases, BGP uses a timer and upon timer expiration, the RIB is polled and next hops are validated. In between walks, next hop changes due to an interface flap that doesn’t affect a BGP session, are not detected. Similarly, between scan cycles, network failures are not detected and routing loops or traffic loss may occur.

Newer BGP implementations, are using an event driven mechanism for next hop address tracking. Usually this involves BGP registering with the RIB the prefixes it needs RIB to track. Next hop changes are rapidly reported to BGP for the registered prefixes as they are updated in the RIB. This event driven optimization improves overall BGP convergence by reducing the response time to next hop changes for routes installed in the RIB.

## Removing the ecmp path

The simplest solution to solve the problem from implementation’s perspective, is to perform the following steps **inline** in the orachagent when the i/f state change is detected:

* Obtain the i/f index.
* Walk all ecmp groups.
* For each path in the ecmp group, compare the i/f id with that in the path. If they match, remove the path from the ecmp group and also call the remove\_next\_hop\_group\_member SAI API to remove the path from the ecmp group in the SAI db.

This solution can only work reliably under the assumption that BGP is using an event driven mechanism for next hop address tracking and it doesn’t suppress events such as an interface flap for optimization purposes.

If BGP is employing a polling/periodic mechanism then with an interface flap in between scans we may run into a state where BGP will have more ecmp paths in its ecmp group than the orchagent and the hw. There will also be no way for BGP to add those ecmp paths back since as far as BGP is concerned, nothing has changed for the prefix regarding its ecmp members. In that case only a subsequent change in the missing ecmp path will cause them to be added back or if the BGP session is cleared, or routes are re-downloaded.

## Marking the ecmp path as stale/non-forwarding

A more reliable solution, but one that is more involved when it comes to implementation, is to go through the following steps inline in the orchagent when the i/f transitions into the down state:

* Obtain the i/f index/name.
* Walk all ecmp groups.
* For each path in the ecmp group, compare the i/f index/name with that in the path. If they match, mark the path as stale/non-forwarding and also call the remove\_next\_hop\_group\_member SAI API to remove the path from the ecmp group in the SAI db.

When the i/f transitions into the up state, go through the following steps **inline** in the orchagent:

* Obtain the i/f index/name.
* Walk all ecmp groups.
* For each path that is marked as stale/non-forwarding in the ecmp group, compare the i/f index/name with that in the path. If they match, clear the stale/non-forwarding flag and also call the add\_next\_hop\_group\_member SAI API to add the path back to the ecmp group in the SAI db.

This approach ensures that BGP and sonic will always be in sync and consistent with each other regarding ecmp group memberships.

However, there is a possibility that this approach when interfaces flap, can result in increased number of ecmp groups and reduced sharing amongst prefixes. This can be solved by merging the ecmp groups but given the number of prefixes and ecmp groups in today’s sonic deployments, it is considered outside the scope of this solution. It is not expected however to encounter such a problem since matching an ecmp path will only consider IP address and its i/f index/name.

# Dependencies

There are only software dependencies in delivering either solution. They both depend on the implementation of the add/remove\_next\_hop\_group\_member SAI APIs.

# CLI

It is desirable to have this behavior by default since it improves convergence and not controlled via a CLI option. This is also in line with existing vendor implementations.