# BGP Multipath and Load Balancing Techniques

This chapter describes the features that enable BGP multipath and load-balancing. These techniques allow you to optimize performance and resilience in complex, large-scale environments.

BGP multipath and load balancing techniques are a set of network solutions that

- enable traffic to use multiple equal-cost paths for improved reliability,
- increase scalability and efficiency by distributing traffic across iBGP and eBGP sessions, and
- support advanced policies and resilient hashing for granular control.


- EIBGP policy-based multipath with equal-cost multipath, on page 1
- Resilient hashing and flow auto-recovery, on page 8
- BGP selective multipath, on page 14
- iBGP multipath load sharing, on page 16
- BGP multipath next-hop and as-path control, on page 18


# EIBGP policy-based multipath with equal-cost multipath

EIBGP policy-based multipath with Equal-Cost Multipath (ECMP) is a BGP feature that

- enables routers to use multiple best paths for forwarding traffic,
- allows load balancing across both internal (iBGP) and external (eBGP) BGP sessions, and
- supports advanced policies like policy-based multipath, ECMP, resilient hashing, and selective multipath.

By using BGP communities, nexthops, path types, and consistent hashing, the solution increases flexibility and reliability in managing traffic distribution across large-scale networks

- ECMP: A routing strategy that allows load balancing over multiple paths with the same cost.
- eBGP/iBGP/eiBGP: External/Internal/External-Internal BGP sessions, defining the relationship between the BGP peers.
- BGP community: An attribute used to tag and group routes for policy application.

*Table 1: Feature History Table*

| Feature Name | Release Name | Description |
|---|---|---|
| **EIBGP policy-based multipath with equal-cost multipath** | Release 7.10.1 | You can control traffic distribution and load balancing in BGP by enabling policy-based multipath selection for iBGP, eBGP, and eiBGP sessions. This approach uses BGP communities, nexthops, and path types to define how paths are selected.<br>Additionally, by using the Equal-Cost Multipath (ECMP) option in eiBGP, you can balance traffic across multiple eligible iBGP paths chosen for eiBGP. This feature gives you greater flexibility and efficiency in managing BGP routing and load balancing.<br>The feature introduces these changes:<br>CLI:<br>The keywords **route-policy** and **equal-cost** are added to the **maximum-paths** command.<br>YANG Data Model:<br>`Cisco-IOS-XR-um-router-bgp-cfg`<br>(see GitHub, YANG Data Models Navigator) |

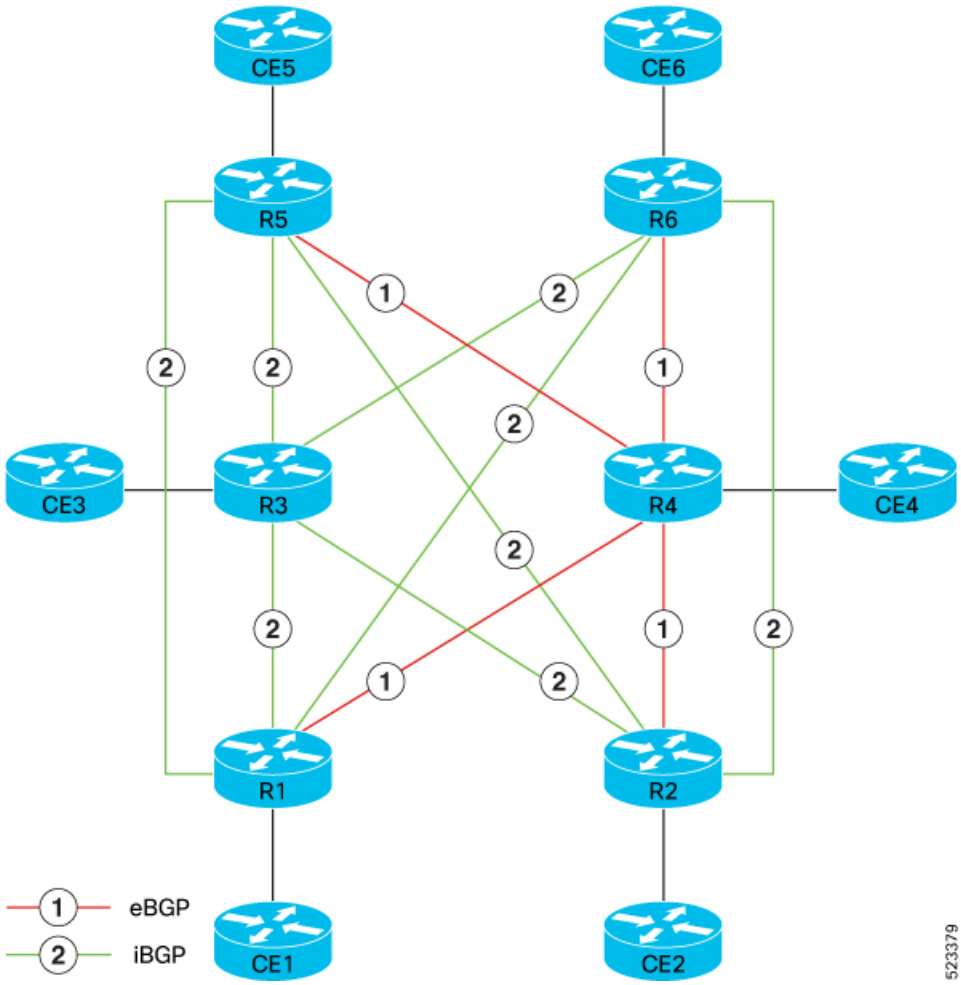
The enhanced policy-based multipath selection in BGP operates now at the default Virtual Routing and Forwarding (VRF) level for variations of BGP, such as iBGP, eBGP, and eiBGP. To improve this functionality, the policy-based multipath selection is now extended to include iBGP, eBGP, and eiBGP by utilizing communities as the underlying mechanism. By utilizing communities, the selection of multiple paths based on specific policy criteria becomes more elaborate. It enables better control over the routing decisions within the BGP network.

eiBGP traditionally implements the unequal-cost mutipath (UCMP) capability to enable the use of both iBGP and eBGP paths. This feature, utilizing the equal-cost multipath option (ECMP), ensures that the nexthop IGP metric remains consistent across the chosen iBGP paths. Hence the metric evaluation is not performed between eBGP and iBGP paths because they have distinct path types.

### Example topology and behavior

This topology illustrates a network comprising BGP peers denoted as R1 through R6.

*Figure 1: Example topology*




Consider a scenario where there is a specific need to transition from using eBGP multipaths to iBGP multipaths. Throughout this transition, you require the simultaneous operation of both eBGP and iBGP to facilitate a seamless migration.

### Topology setup

This topology showcases distinct path types, where eBGP paths are visually depicted using a red-colored line labeled as 1, and the iBGP paths are visually illustrated using a green-colored line labeled as 2.

### Classic eiBGP behavior

In the context of CE routers, CEI to CE6, the preferred path for prefixes will be from eBGP, specifically from the R4 router. The selection of best paths prioritizes eBGP multipaths from R4, although paths might exist from R5 and R6 routers and also from RI and R2 routers through iBGP. This is the classic behavior. In classic eiBGP, unequal cost paths are employed, leading to the disregard of metrics. However, you rely on the IGP metric for optimal performance.

### Change in eiBGP behavior

The iBGP paths with the shortest AS-PATH length are chosen for R5 and R6. The same iBGP multipath selection process applies to paths from R1 and R2. As a result, R1 and R2 establish iBGP peering sessions

with R3. This setup creates a combination of eBGP and iBGP paths, called eiBGP, which are available for prefixes advertised to hosts beyond the CE devices. The CE routers must balance prefixes between R3 and R4, and you should exclude paths from R5, R6, R1, and R2. To do this, configure the additive community attribute on R1 and R2 for routes advertised toward R5 and R6.

With this topology, you can run both eBGP and iBGP, allowing a smooth transition between eBGP and iBGP multipaths. Include the default VRF in policy-based multipath selection to apply route policies that control how your network distributes traffic. Use BGP attributes such as communities, next hops, and path types in these policies to select paths. For example, use BGP communities to prioritize routes or change next hops to direct traffic over specific paths. This approach helps you optimize routing decisions, control traffic distribution, and improve load balancing across all BGP types in your network.

Enable ECMP to distribute traffic evenly across multiple equal-cost paths. This prevents overloading any single path and improves load balancing. With the ECMP option in eiBGP, the router can use multiple iBGP paths with equal cost for traffic distribution.

#### Benefits of EIBGP policy-based multipath with ECMP

- Optimizes traffic distribution and network utilization across iBGP, eBGP, and eiBGP sessions.
- Enables seamless migration between eBGP and iBGP multipath scenarios.
- Provides granular control over path selection using communities, nexthops, and path types.
- Ensures efficient load balancing and prevents single-path overload by leveraging ECMP.

#### Caveats of multipath selection without BGP attributes and ECMP

- If BGP communities, nexthops, or path types are not used in policy-based multipath selection, control over routing is reduced, leading to suboptimal load balancing.
- Not enabling ECMP in eiBGP causes the router to rely on classic best-path selection and forgo the benefits of multipath load balancing.
- Avoid comparing IGP metrics between eBGP and iBGP paths, as ECMP only considers iBGP paths with equal cost.

# Restrictions and guidelines for eiBGP policy-based multipath with ECMP

## eiBGP multipath configuration restrictions

- You cannot configure eiBGP along with either eBGP or iBGP.
- The maximum-paths route policy checks only the community, next hop, and path type.
- The OpenConfig model is not supported.

## eiBGP multipath configuration guidelines

- Use the Accumulated Interior Gateway Protocol (AIGP) metric attribute only with equal-cost eiBGP paths.
- When you configure both eBGP and iBGP multipath, you can assign the same or different route policies to each. The router automatically applies the policy for the best path type of each prefix:

- If iBGP provides the best path for a prefix, the iBGP route policy is applied.
- If eBGP provides the best path, the eBGP route policy is applied.

# How eiBGP policy-based multipath with ECMP works

### Summary

The key components involved in the process are:

- BGP communities: Used for tagging and grouping routes according to policy.
- Multipath selection policy: Determines which iBGP, eBGP, or eiBGP paths are included based on community, nexthop, and path type.
- ECMP option: Ensures that equal-cost iBGP paths are included in eiBGP multipath selection.

Enhanced policy-based multipath selection operates at the default VRF for iBGP, eBGP, and eiBGP. The solution uses BGP communities to enable flexible and granular selection of multiple paths according to policy.

### Workflow

These stages describe how eiBGP policy-based multipath with ECMP works:

1. The router receives routes from eBGP and iBGP peers.
2. Route policies using communities are applied to select eligible multipaths.
3. With ECMP enabled, the router balances traffic across all equal-cost iBGP paths that are chosen as part of the eiBGP multipath set.
4. The router installs the selected multipaths and forwards traffic accordingly.

### Result

Traffic distribution and load balancing are controlled and optimized, supporting seamless transitions between eBGP and iBGP multipath operation.

# Configure eiBGP policy-based multipath with ECMP

Enable eiBGP policy-based multipath routing with ECMP by applying route policies that use communities, path types, or next hops.

Use this task to configure ECMP for eiBGP, allowing your routers to consider multiple eligible paths for load balancing, based on policy.

### Before you begin

- Ensure that BGP and required address families are enabled on all participating routers.
- Have appropriate community values and route policies planned.

Follow these steps to configure eiBGP policy-based multipath with ECMP.

### Procedure

**Step 1** Define a community set for R1 and R2 routers.

**Example:**

```
Router(config)# community-set ABC
Router(config-comm)# 2:1
Router(config-comm)# end-set
```

**Step 2** Create a route policy named EIBGP on R1 and R2.

**Example:**

```
Router(config)# route-policy EIBGP
Router(config-rpl)# if community matches-any ABC then
Router(config-rpl-if)# pass
Router(config-rpl-if)# else
Router(config-rpl-else)# drop
Router(config-rpl-else)# endif
Router(config-rpl)# end-policy
```

This policy checks the BGP communities of received routes and takes action based on the community value.

- If the community matches "ABC", the route is eligible for multipath.
- If not, the route is dropped from multipath consideration.
- You can also match on path-type or next-hop in the route policy as needed.

**Step 3** Configure BGP to use ECMP for eiBGP and apply the route policy.

**Example:**

```
Router(config)# router bgp 100
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# maximum-paths eibgp 32 equal-cost route-policy EIBGP
Router(config-bgp-af)# commit
```

**Step 4** Verify the running configuration.

**Example:**

```
Router# show running-config
community-set ABC
 2:1
 end-set
!

route-policy EIBGP
  if community matches-any ABC then
    pass
  else
    drop
  endif
end-policy router bgp 100
 address-family ipv4 unicast
  maximum-paths eibgp 32 equal-cost route-policy EIBGP
!
```

**Step 5** Verify that eiBGP multipath and the route policy are working as intended.

**Example:**

```
Router# show bgp 203.0.113.99/32
BGP routing table entry for 203.0.113.99/32
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 27          27
Last Modified: Feb 23 16:08:54.000 for 04:12:23
Paths: (7 available, best #2)
  Advertised IPv4 Unicast paths to update-groups (with more than one peer):
    0.1 0.4
  Path #1: Received by speaker 0
  Not advertised to any peer
  200 300
    209.165.200.11  from 209.165.200.11 (192.168.0.3), -> From R4
   Origin IGP, localpref 100, valid, external, multipath
      Received Path ID 0, Local Path ID 0, version 0
      Community: 2:1
      Origin-AS validity: (disabled)

  Path #2: Received by speaker 0
  Advertised IPv4 Unicast paths to update-groups (with more than one peer):
    0.1 0.4
  200 300
    209.165.201.1  from 209.165.201.1  (209.165.201.1) -> From R4
   Origin IGP, localpref 100, valid, external, best, group-best, multipath
      Received Path ID 0, Local Path ID 1, version 27
      Community: 2:1
      Origin-AS validity: (disabled)

  Path #3: Received by speaker 0
  Not advertised to any peer
  200 300, (Received from a RR-client)
    192.168.2.6 (metric 2) from 198.51.100.1  (198.51.100.1)  -> From R3
   Origin IGP, localpref 100, valid, internal, multipath, backup, add-path
      Received Path ID 0, Local Path ID 2, version 6
      Community: 2:1
  Path #4: Received by speaker 0
  Not advertised to any peer
  200 300, (Received from a RR-client)
    192.168.0.6 (metric 2) from 192.0.2.1 (192.0.2.1)  -> From R5
      Origin IGP, localpref 100, valid, internal
      Received Path ID 0, Local Path ID 0, version 0
      Community: 11:11 99:99

  Path #5: Received by speaker 0
  Not advertised to any peer
  200 300, (Received from a RR-client)
    192.168.0.2 (metric 5) from 192.168.0.2 (192.168.0.2)  -> From R2
      Origin IGP, localpref 100, valid, internal
      Received Path ID 0, Local Path ID 0, version 0
      Community: 2:1 99:99
/* The router does not select Path 5, even though it satisfies the route-policy community constraint,
 because it has a higher metric (i.e., metric 5) than the best path of its path type (i.e., iBGP
metric 2). */

  Path #6: Received by speaker 0
  Not advertised to any peer
  200 300, (Received from a RR-client)
    192.168.0.4 (metric 2) from 192.168.0.4 (192.168.0.4) -> From R5
      Origin IGP, localpref 100, valid, internal
      Received Path ID 0, Local Path ID 0, version 0
      Community: 11:11 99:99
```

```
Path #7: Received by speaker 0
Not advertised to any peer
100 300, (Received from a RR-client)
  192.168.0.5 (metric 2) from 192.168.0.5 (192.168.0.5) -> From R3
 Origin IGP, localpref 100, valid, internal, multipath
    Received Path ID 0, Local Path ID 0, version 0
    Community: 2:1
```

Review the output to confirm that multiple eligible paths are installed and used according to the route policy and ECMP configuration.

The router selects paths for multipath if they match the community criteria and the metric of the best path within the same path type, iBGP or eBGP. A path with a higher metric than the best path of its type is not selected, even if it meets the community constraint.

---

The router uses ECMP for eiBGP routes that meet your route policy conditions, improving traffic distribution and load balancing.

# Resilient hashing and flow auto-recovery

Resilient hashing and flow auto-recovery is a network reliability feature that

- maintains stable traffic flows during Equal-Cost Multipath (ECMP) path failures by only rerouting traffic affected by the failed path,
- prevents unnecessary rebalancing of existing flows to new links, and
- automatically restores original flow distribution when a failed path or server returns to service.
- ECMP: A routing technique that balances traffic across multiple equal-cost paths to the same destination.
- Bucket: A logical mapping of flows to paths in a hashing algorithm.

*Table 2: Feature History Table*

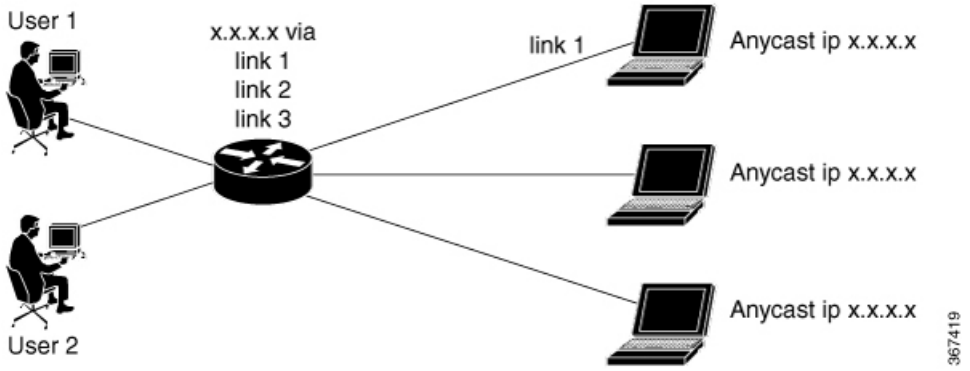
| Feature Name | Release Name | Description |
|---|---|---|
| **Resilient hashing and flow auto-recovery** | Release 24.4.1 | Introduced in this release on: Fixed Systems (8700 [ASIC:K100])(select variants only*).<br><br>You can ensure no packet loss and optimal load distribution across available paths by automatically rerouting data flows during link failures. This feature enhances network reliability by maintaining continuous service and dynamically adjusting to network topology changes without manual intervention. It seamlessly integrates with existing configurations, offering high availability and reducing downtime, thus keeping network operations uninterrupted and efficient.<br><br>*Previously this feature was supported on Q200 and Q100. It is now extended to Cisco 8712-MOD-M routers. |

Resilient hashing and flow auto-recovery let you selectively override the default equal cost multipath (ECMP) behavior during an ECMP path failure. This feature redirects only the flows on failed links and prevents all existing flows from being rehashed to a new link. It also allows a recovered link or server to be reused for sessions when it becomes available.

### Impact of ECMP path failures on traffic flows

Prior to the implementation of resilient hashing and flow auto-recovery feature, ECMP load balances traffic across all available paths to a destination. If one path fails, ECMP rehashes the traffic and selects new next hops for each flow.

*Figure 2: ECMP Path Failure*




For example, if you have three links—link 1, link 2, and link 3—a traffic flow that originally uses link 1 may switch to link 3 after a failure, even if only link 2 fails.

This redistribution of traffic flows does not cause issues in traditional core networks because end-to-end connectivity is maintained and users are not affected. However, in data center environments, load balancing caused by this redistribution can create problems.

In data centers where multiple servers connect through ECMP, rehashing may cause active flows to move, which can reset TCP sessions and disrupt applications.

### Benefits of resilient hashing and flow auto-recovery

- Maintains uninterrupted network operations and high availability
- Minimizes traffic disruption and session resets during link failures or recoveries, and
- Supports dynamic adjustment to topology changes without manual intervention.

## How resilient hashing and flow auto-recovery work

### Summary

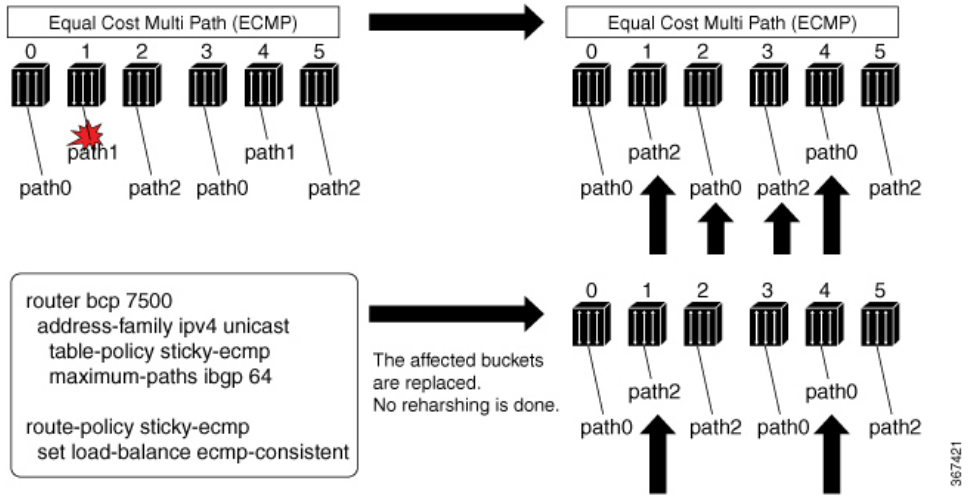
The key components involved in the process are:

- Resilient hashing configuration: Determines how flows are distributed and reassigned when a path fails.
- Route policy language (RPL): Used to specify the prefixes that require resilient hashing and flow auto-recovery.
- ECMP path list: Represents the set of available equal-cost paths for a given prefix.

Resilient hashing and flow auto-recovery help maintain consistent traffic distribution and minimize disruption during ECMP path failures and recoveries. This process ensures only affected traffic flows are redirected, while existing flows remain stable.

## Workflow

***Figure 3: Resilient hashing and flow auto-recovery***




These stages describe how resilient hashing and flow auto-recovery work.

1. The router uses an RPL to define prefixes with associated ECMP path lists, such as path 0, path 1, and path 2 for prefix X.
2. If a path fails, for example, path 1:
   - Without resilient hashing: The router performs a full rehashing, redistributing all flows across the remaining available paths, for example, path 0, path 2, and path 0.
   - With resilient hashing and flow auto-recovery enabled: Only the affected traffic buckets are reassigned, for example, the new path list becomes path 0, path 0, and path 2, and no complete rehash occurs.
3. When the failed path becomes active again, for example, path 1:
   - Without resilient hashing and flow auto-recovery: The path is not reused until one of the following actions happens:
     - Addition of a new path to ECMP
     - Use of the **clear route** command
     - Removal and reapplication of a table policy followed by a **commit**, or
     - Configuration of the **cef consistent-hashing auto-recovery** command
   - With resilient hashing and flow auto-recovery enabled: Sessions previously moved to other paths during the failure are automatically rehashed back to the restored path. Only these specific sessions are disrupted, while others remain unaffected.

### Result

Resilient hashing and flow auto-recovery provide stable traffic flows during ECMP path failures and recoveries, minimizing service disruption and ensuring efficient use of all available paths.

# Configure resilient hashing and flow auto-recovery

To realize resilient hashing and flow auto-recovery, you can use persistent load balancing, also known as sticky ECMP. Sticky ECMP ensures that when a path failure occurs, only the flows that relied on the failed path are reassigned, while all other flows continue on their original routes.

Traditional ECMP load balances traffic across multiple available paths. When a path fails, ECMP redistributes all flows, which can disrupt established sessions. Persistent load balancing, also known as sticky ECMP, ensures that only flows using the failed path are reassigned, while all other flows remain unchanged.

**Before you begin**

- Make sure BGP and ECMP are already configured in your network.
- Identify the prefixes that require sticky ECMP.

Follow these steps to configure resilient hashing and flow auto-recovery using sticky ECMP.

**Procedure**

**Step 1** Enter BGP configuration mode and set up the table policy for sticky ECMP.

**Example:**

```
Router(config)# router bgp 7500
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# table-policy sticky-ecmp
Router(config-bgp-af)# bgp attribute-download
Router(config-bgp-af)# maximum-paths ebgp 64
Router(config-bgp-af)# maximum-paths ibgp 32
Router(config-bgp-af)# exit
Router(config-bgp)# exit
```

**Step 2** Define a route policy that applies sticky ECMP to the required destination prefix.

**Example:**

```
Router(config)# route-policy sticky-ecmp
Router(config-rpl)# if destination in (192.1.1.1/24) then
Router(config-rpl-if)# set load-balance ecmp-consistent
Router(config-rpl-if)# else
Router(config-rpl-else)# pass
Router(config-rpl-else)# endif
RP/0/0/CPU0:ios(config-rpl)# end-policy
RP/0/0/CPU0:ios(config)#
```

**Step 3** Enable automatic recovery to restore the original hashing state after failed paths become active.

**Example:**

```
Router(config)# cef consistent-hashing auto-recovery
```

**Step 4** Clear the route to recover the original hashing state after failed paths come back up and avoid affecting new flows.

**Example:**

```
Router(config)# clear route 192.0.2.0/24
```

**Step 5** Verify the running configuration.

**Example:**

```
Router#show running-config
router bgp 7500
 address-family ipv4 unicast
  table-policy sticky-ecmp
  bgp attribute-download
  maximum-paths ebgp 64
  maximum-paths ibgp 32


cef consistent-hashing auto-recovery

clear route 192.0.2.0/24
```

**Step 6** Verify that persistent load balancing is working as expected by checking the path distribution before and after a link failure.

a) Run the **show cef** command to check the path distribution before a link failure.

**Example:**

```
Router# show cef 192.0.2.0/24
 LDI Update time Sep  5 11:22:38.201
   via 10.1.0.1/32, 3 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 0 NHID 0x0 [0x57ac4e74 0x0]
    next hop 10.1.0.1/32 via 10.1.0.1/32
   via 10.2.0.1/32, 3 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 1 NHID 0x0 [0x57ac4a74 0x0]
    next hop 10.2.0.1/32 via 10.2.0.1/32
   via 10.3.0.1/32, 3 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 2 NHID 0x0 [0x57ac4f74 0x0]
    next hop 10.3.0.1/32 via 10.3.0.1/32


    Load distribution (consistent): 0 1 2 (refcount 1)

    Hash  OK  Interface                 Address
    0     Y   GigabitEthernet0/0/0/0    10.1.0.1
    1     Y   GigabitEthernet0/0/0/1    10.2.0.1
    2     Y   GigabitEthernet0/0/0/2    10.3.0.1
```

b) Run the **show cef** command to check the path distribution after a link failure.

**Example:**

```
Router# show cef 192.0.2.0/24
 LDI Update time Sep  5 11:23:13.434
   via 10.1.0.1/32, 3 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 0 NHID 0x0 [0x57ac4e74 0x0]
    next hop 10.1.0.1/32 via 10.1.0.1/32
   via 10.3.0.1/32, 3 dependencies, recursive, bgp-multipath [flags 0x6080]
    path-idx 1 NHID 0x0 [0x57ac4f74 0x0]
    next hop 10.3.0.1/32 via 10.3.0.1/32

    Load distribution (consistent) : 0 1 2 (refcount 1)
    Hash  OK  Interface                 Address
    0     Y   GigabitEthernet0/0/0/0    10.1.0.1
 1*    Y   GigabitEthernet0/0/0/0    10.1.0.1
    2     Y   GigabitEthernet0/0/0/2    10.3.0.1
```

The reassignment of bucket 1 to GigabitEthernet 0/0/0/0, indicated by the "*" symbol, shows that this path is being used as a replacement for a failed path.

Persistent load balancing ensures that only flows on failed paths are reassigned, maintaining session stability for all other traffic.

# BGP selective multipath

BGP selective multipath is a routing feature that

- allows you to install multiple parallel paths to the same destination in the routing table
- applies the multipath function only to specified BGP peers instead of all configured peers, and
- enables fine-grained control over which neighbor routes are eligible for multipath installation.

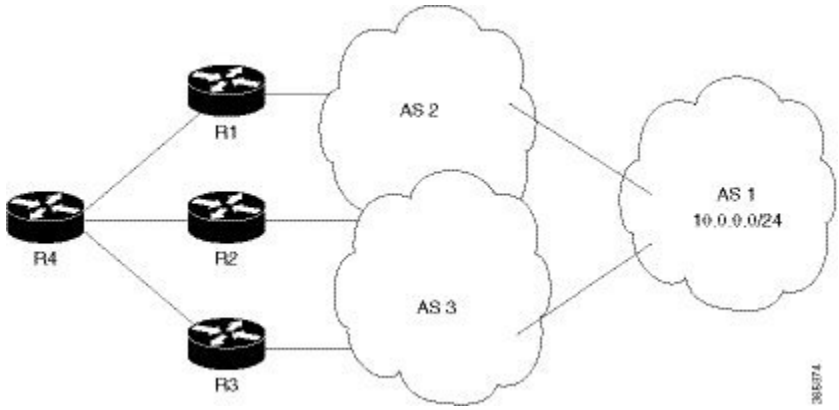
Traditional BGP multipath installs multiple paths from all configured peers by default. With selective multipath, you choose specific neighbors that participate in multipath routing.

## Guidelines for BGP selective multipath

- Always include the best path in the set of multipaths; BGP selective multipath does not affect best path calculations.
- Ensure provider edge (PE) paths for VPN prefixes are treated as eligible multipaths.
- Use the **next-hop-unchanged multipath** command to avoid overwriting next-hop information before advertising multipaths.
- Configure the multipath option on neighbors that should be eligible for parallel path installation in the routing table.
- Set the maximum number of selective multipaths for iBGP and eBGP neighbors by using the **maximum-paths ... selective** command.

# Topology for BGP selective multipath

*Figure 4: Topology for BGP selective multipath*




In this sample topology, router R4 receives parallel paths to the same destination from routers R1, R2, and R3. If only R1 and R2 are configured as selective multipath neighbors on R4, then only the parallel paths from R1 and R2 are installed in R4 routing table. Routes from R3 are not used for multipath unless configured.

# Configure BGP selective multipath

Enable BGP selective multipath so that only selected neighbors provide multipath routes.

Use this task to control which BGP neighbors contribute multipath routes in your network, improving path management and stability.

**Before you begin**

Configure your network topology with iBGP or eBGP before enabling selective multipath.

- Make sure iBGP or eBGP are already running on your routers.
- Design your network topology and identify neighbor IP addresses for selective multipath.

Follow these steps to configure BGP selective multipath on router R4.

**Procedure**

**Step 1** Enter BGP address-family configuration and set maximum selective multipaths.

a) Configure selective multipath for iBGP and eBGP.

**Example:**

```
Router(config)# router bgp 1
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# maximum-paths ibgp 4 selective
Router(config-bgp-af)# maximum-paths ebgp 5 selective
Router(config-bgp-af)# commit
```

b) Configure selective multipath for eiBGP.

**Example:**

```
Router(config)# router bgp 1
Router(config-bgp)# address-family ipv4 unicast
Router(config-bgp-af)# maximum-paths eibgp 6 selective
Router(config-bgp-af)# commit
```

Choose the right configuration based on your network topology.

**Step 2** Configure neighbors for selective multipath.

a) For neighbors eligible for multipath, R1 and R2:

**Example:**

```
Router(config-bgp)# neighbor 1.1.1.1
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# multipath
Router(config-bgp-nbr-af)# commit

Router(config-bgp-nbr)# neighbor 2.2.2.2
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# multipath
Router(config-bgp-nbr-af)# commit
```

b) For a neighbor not eligible for multipath, R3:

**Example:**

```
Router(config-bgp-nbr)# neighbor 3.3.3.3
Router(config-bgp-nbr)# address-family ipv4 unicast
Router(config-bgp-nbr-af)# commit
```

Only the specified neighbors, R1 and R2, provide parallel multipath routes in the routing table of router R4.

# iBGP multipath load sharing

The iBGP multipath load sharing is a BGP capability that:

- allows a router to install multiple iBGP paths to the same destination in the routing table,
- enables load sharing across these eligible paths, and
- improves network efficiency by leveraging path diversity.

By default, when a BGP router without a local policy receives multiple iBGP routes to the same destination, it selects only one best path and installs it in the routing table. With the iBGP multipath load sharing feature, the router can install multiple iBGP paths as best paths for the same destination, allowing traffic to be shared across these paths.

### Benefits of iBGP multipath load sharing

These are some of the benefits that iBGP multipath load sharing provides:

- Increases bandwidth utilization by distributing traffic over multiple iBGP paths

- Improves network resiliency and redundancy
- Enhances path diversity, reducing the risk of single points of failure, and
- Optimizes traffic flow and balances network load.

# Restrictions for iBGP multipath load sharing

- You cannot use per-VRF label mode with Carrier Supporting Carrier (CSC) networks that have both internal and external BGP multipath.
- Do not use per-VRF label mode for BGP PIC edge with eBGP multipath, because it may cause routing loops. Use per-prefix label mode instead.
- Per-VRF label mode does not support BGP best external within the same address family, as it can cause routing loops during network re-convergence.

# Configure iBGP multipath load sharing

Enable iBGP multipath load sharing to use multiple iBGP paths for the same destination and improve load distribution.

Use this task to configure the maximum number of iBGP paths for load sharing in the BGP routing process.

**Before you begin**

- Confirm you have the correct autonomous system (AS) number for your router.
- Determine the required address family, IPv4 or IPv6, unicast or multicast.
- Decide the maximum number of iBGP paths you want to configure.

**Procedure**

**Step 1** Configure the AS number for your router.

**Example:**

```
Router# config
Router(config)# router bgp 100
```

Specifies the autonomous system number and enters the BGP configuration mode, allowing you to configure the BGP routing process.

**Step 2** Run the **address-family** {**ipv4**|**ipv6**} {**unicast**|**multicast**} command to enter the appropriate address family submode.

**Example:**

```
Router(config-bgp)# address-family ipv4 multicast
```

**Step 3** Run the **maximum-paths ibgp** *number* command to configures the maximum number of iBGP paths for load sharing.

**Example:**

```
Router(config-bgp-af)# maximum-paths ibgp 30
Router(config-bgp-af)# commit
```

**Step 4** Verify the running configuration.

**Example:**

```
Router# show running-config
router bgp 100
 address-family ipv4 multicast
  maximum-paths ibgp 30
 !
!
end
```

This sample configuration uses 30 iBGP paths for load sharing.

# BGP multipath next-hop and as-path control

BGP multipath next-hop and as-path control is a set of enhancements that:

- prevents overwriting of **next-hop** values for multipath prefixes,
- allows you to control how **as-path** information is used when selecting multipath routes, and
- helps maintain predictable and loop-free routing behavior.

You can use the **next-hop-unchanged multipath** command to ensure that BGP does not recalculate the next hop for multipath prefixes.

The **bgp multipath as-path ignore onwards** command enables you to ignore **as-path** attributes beyond a certain point when computing multipath, allowing for more flexible path selection.

## Guidelines for BGP multipath next-hop and as-path control

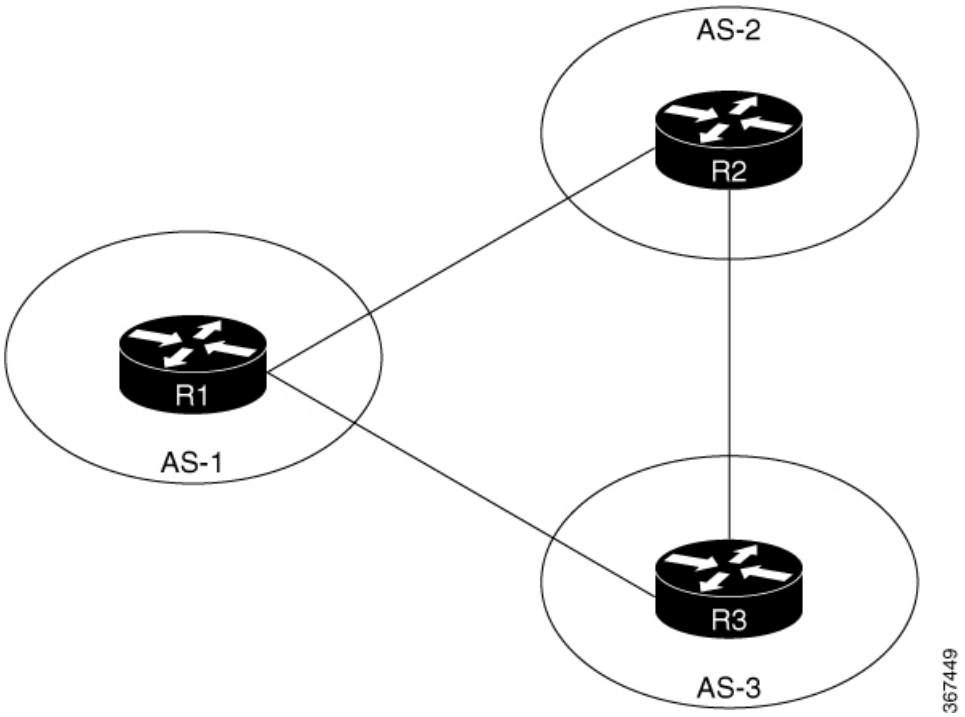
Use the **next-hop-unchanged multipath** command to disable next-hop recalculation for multipath prefixes and preserve the original next-hop values.

**Caution** Use the **bgp multipath as-path ignore onwards** command with caution. If you enable this command on multiple interconnected routers, it may cause routing loops.

## Topology illustrating routing loop formation

Illustrate how specific BGP multipath configurations can create routing loops between connected routers.

*Figure 5: Topology example: routing loop formation*




Consider three routers named R1 in AS-1, R2 in AS-2, and R3 in AS-3, with each router in a different autonomous system and all interconnected. R1 announces a prefix to both R2 and R3. If R2 and R3 are both configured for multipath and have the **bgp multipath as-path ignore onwards** command enabled, each router may forward part of its traffic to the other. This behavior creates a forwarding loop between R2 and R3. To avoid such loops, do not enable the **as-path ignore onwards** option on interconnected routers.

Topology illustrating routing loop formation