

Enabling Segment Routing Flexible Algorithm

Segment Routing Flexible Algorithm allows operators to customize IGP shortest path computation according to their own needs. An operator can assign custom SR prefix-SIDs to realize forwarding beyond link-cost-based SPF. As a result, Flexible Algorithm provides a traffic engineered path automatically computed by the IGP to any destination reachable by the IGP.

The SR architecture associates prefix-SIDs to an algorithm which defines how the path is computed. Flexible Algorithm allows for user-defined algorithms where the IGP computes paths based on a user-defined combination of metric type and constraint.

- Feature History, on page 2
- Prerequisites for Flexible Algorithm, on page 3
- Restrictions for Flexible Algorithm, on page 3
- Building Blocks of Segment Routing Flexible Algorithm, on page 3
- Flexible Algorithm Prefix-SID Redistribution , on page 5
- Flexible Algorithm Prefix Metric Advertisement, on page 6
- Flexible Algorithm Configurations, on page 7
- Verifying the Flexible Algorithm Configuration, on page 13

Feature History

Table 1: Feature History

Feature Name	Release Information	Feature Description
TE Metric Support for IS-IS Flexible Algorithm	Cisco IOS XE Dublin 17.11.1a	Flexible algorithm allows user-defined algorithms where the Interior Gateway Protocol (IGP) computes paths based on a user-defined combination of metric type (path optimization objective) and constraint. This feature adds support for the TE metric as a metric type for the IS-IS Flexible Algorithm feature. This allows the TE metric along with the IGP and delay metrics to be used when running the shortest path computations.
Segment Routing Flexible Algorithm Prefix SID Redistribution	Cisco IOS XE Cupertino 17.8.1	With this feature, prefix SIDs are provided for all supported algorithms when a prefix is redistributed. This feature is enabled automatically when you configure redistribution of routes with strict or Flexible Algorithm SIDs.
IS-IS Flexible Algorithm Include Affinity Support	Cisco IOS XE Bengaluru 17.6.1	This feature supports **include-any** and **include-all** affinities in IS-IS. Prior to Cisco IOS XE Bengaluru 17.6.1 release, only Flexible Algorithm affinity **exclude-any** was supported.
Segment Routing Flexible Algorithm	Cisco IOS XE Bengaluru 17.4.1	TI LFA and uLoop Avoidance: Allows computation of Loop Free Alternate (LFA) paths. TI-LFA backup paths using the same constraints as the calculation of the primary paths for Flexible Algorithms, for IS-IS.

Feature Name	Release Information	Feature Description
Segment Routing Flexible Algorithm	Cisco IOS XE Amsterdam17.3.1	Segment Routing Flexible Algorithm allows operators to customize IGP shortest path computation according to their own needs. An operator can assign custom SR prefix-SIDs to realize forwarding beyond link-cost-based SPF. As a result, Flexible Algorithm provides a traffic engineered path automatically computed by the IGP to any destination reachable by the IGP Support for affinity **exclude-any**.

Prerequisites for Flexible Algorithm

Segment routing must be enabled on the router before the Flexible Algorithm functionality is activated.

Restrictions for Flexible Algorithm

- A maximum of 20 IS-IS flexible algorithm sessions are supported.
- In IS-IS, the flexible algorithm affinity "exclude-any", "include-any", and "include-all" are supported.

Building Blocks of Segment Routing Flexible Algorithm

This section describes the building blocks that are required to support the SR Flexible Algorithm functionality in IS-IS and OSPF.

Flexible Algorithm Definition

Many possible constraints may be used to compute a path over a network. Some networks are deployed with multiple planes. A simple form of constraint may be to use a particular plane. A more sophisticated form of constraint can include some extended metric, like delay, as described in [RFC 8570]. Even more advanced case could be to restrict the path and avoid links with certain affinities. Combinations of these are also possible. To provide a maximum flexibility, the mapping between the algorithm value and its meaning can be defined by the user. When all the routers in the domain have the common understanding what the particular algorithm value represents, the computation for such algorithm is consistent and the traffic is not subject to looping. Here, since the meaning of the algorithm is not defined by any standard, but is defined by the user, it is called as Flexible Algorithm.

Flexible Algorithm Support Advertisement

An algorithm defines how the best path is computed by IGP. Routers advertise the support for the algorithm as a node capability. Prefix-SIDs are also advertised with an algorithm value and are tightly coupled with the algorithm itself.

An algorithm is a one octet value. Values from 128 to 255 are reserved for user defined values and are used for Flexible Algorithm representation.

Flexible Algorithm Definition Advertisement

To guarantee the loop free forwarding for paths computed for a particular Flexible Algorithm, all routers in the network must share the same definition of the Flexible Algorithm. This is achieved by dedicated router(s) advertising the definition of each Flexible Algorithm. Such advertisement is associated with the priority to make sure that all routers will agree on a single and consistent definition for each Flexible Algorithm.

Definition of Flexible Algorithm includes:

- Metric type
- Affinity constraints

To enable the router to advertise the definition for the particular Flexible Algorithm, **advertise-definition** command is used. At least one router in the area, preferably two for redundancy, must advertise the Flexible Algorithm definition. Without the valid definition being advertised, the Flexible Algorithm will not be functional.

Flexible Algorithm Prefix-SID Advertisement

To forward traffic on a Flexible Algorithm specific path, all routers participating in the Flexible Algorithm install an MPLS labeled path for the Flexible Algorithm specific prefix-SID. This Flexible Algorithm specific prefix-SID is advertised for the prefix. Only prefixes for which the Flexible Algorithm specific Prefix-SID is advertised, are subject to Flexible Algorithm specific forwarding.

Inter-Area Leaking

Effective Cisco IOS XE Bengaluru 17.4.1, Flexible Algorithm SIDs and prefixes are leaked between IS-IS areas. However, only the prefixes that are reachable by Level 1 or Level 2 paths are leaked. Similarly, only SIDs that are reachable in a given Flexible Algorithm are leaked.

For example, consider a prefix P:

- that is originated in Level 1 and leaked in to Level 2
- has SID value = 128 in Flexible Algorithm 128, and SID value = 129 in Flexible Algorithm 129
- for which Level 1 path exist only for SID value = 128, but not for SID value = 129

As a result of the above conditions, only SID 128 is leaked from Level 1 to Level 2 and not SID 129.

Calculation of Flexible Algorithm Path

A router may compute path for multiple Flexible Algorithms. A router must be configured to support particular Flexible Algorithm before it can compute any path for such Flexible Algorithm. A router must have a valid definition of the Flexible Algorithm before such Flexible Algorithm is used.

When computing the shortest path tree for particular Flexible Algorithm:

- All nodes that do not advertise support for such Flexible Algorithm will be pruned from the topology.
- If the Flexible Algorithm definition includes affinities that are excluded, then all links for which any of such affinities are advertised will be pruned from the topology.
- Router uses the metric that is part of the Flexible Algorithm definition. If the metric is not advertised for the particular link, such link will be pruned from the topology.

For OSPF and IS-IS, LoopFree Alternate (LFA) paths, and TI-LFA backup paths for a Flexible Algorithm are computed using the same constraints as the calculation of the primary paths for such Flexible Algorithm. These paths use Prefix-SIDs advertised specifically for such Flexible Algorithm to enforce a backup.

Installation of Forwarding Entries for Flexible Algorithm Paths

Flexible Algorithm paths to any prefix must be installed in the forwarding entries using the Prefix-SID that was advertised for such Flexible Algorithm. If the Prefix-SID for Flexible Algorithm is not known, such Flexible Algorithm path is not installed in forwarding for such prefix.

Only MPLS to MPLS entries are installed for a Flexible Algorithm path. No IP to IP or IP to MPLS entries are installed. These follow the native IPG paths computed based on the default algorithm and regular IGP metrics.

You can selectively filter the paths that are installed to the MFI by using the configuration command **distribute-list** *filter name* **in**. See Configuring Selective Path Filtering for configuration example. This feature is only supported for IS-IS Flexible Algorithm.

Flexible Algorithm Prefix-SID Redistribution

Prior to Cisco IOS XE 17.8, when prefixes were redistributed between protocols, only Prefix SIDs for SR algorithm 0 (regular SPF) were available.

In Cisco IOS XE 17.8, support for providing prefix SIDs for all supported algorithms when a prefix is redistributed is added. This feature is called the Segment Routing Flexible Algorithm Prefix SID Redistribution. This feature is enabled automatically when you configure redistribution of routes with strict or Flexible Algorithm SIDs.

When OSPF redistributes to ISIS, it redistributes all the algorithm prefixes and ISIS processes it. When ISIS redistributes to OSPF, only the base algorithm prefixes are processed by OSPF. Redistribution of other flexible algorithm prefixes are not supported in OSPF. For example, OSPF 10 is redistributed into ISIS 30, the strict SIDs and flexible algorithm SIDs are processed by ISIS. However, if ISIS 30 redistributes into OSPF 10, only the strict SIDs are processed by OSPF.

OSPF supports strict SPF and flexible algorithm. However, it does not support redistribution. For example, OSPF 10 and OSPF 20 are two instances having strict SPF and flexible algorithm. If OSPF 10 is redistributed into OSPF 20, then OSPF 20 will not process the strict SID and flexible algorithm SIDs of OSPF 10.

Displaying the Algorithm Information

You can use the **show mpls forwarding-table** command to display the non-zero algorithm specific prefix SID Label MPLS forwarding information. The command syntax is as follows:

show mpls forwarding <*ip*> <*mask*> [**algo** <*algo-number*>]

For more information, see Verifying the Flexible Algorithm Configuration, on page 13.

Flexible Algorithm Prefix Metric Advertisement

Segment Routing Flexible Algorithm Prefix Metric allows operators to associate metric computed inthe given Flexible Algorithm with a prefix during prefix inter-level leaking or inter-domain redistribution. It helps to compute the optimal inter-level or inter-domain path. When you configure Flexible Algorithm to support prefix-metric, the prefix-metric flag (M-flag) is advertised in ISIS Flexible Algorithm definition flags Sub-TLV. The sub-TLV is advertised only by the Level 1 and Level 2 routers. To view the prefix-metric flag (M-flag), use the **show isis database verbose** command. For more information, see Verifying the Flexible Algorithm Configuration, on page 13.

If a given flex algo algorithm (128-255) specifies the use of the Flex Algo Prefix Metric (FAPM), then the metric associated with the prefix *must* be advertised using the algorithm specific FAPM sub-TLV by the ABRs which advertise the prefix into other levels/areas. When the Flexible Algorithm Definition specifies the use of FAPM (M-flag) then only prefixes which have an algorithm specific FAPM advertisement will be considered reachable in the algorithm specific topology.



Note Cisco IOS XE supports flexible algorithm prefix-metric insertion only during prefix inter-level leaking and not during inter-domain redistribution.

The ISIS Flexible Algorithm Prefix Metric Sub-TLV supports the advertisement of a Flexible Algorithm specific prefix metric associated with a given prefix advertisement.

The following command is used to enable advertisement of the Flexible Algorithm Prefix Metric:

```
router isis 1
flex-algo 128
  advertise-definition
  prefix-metric

# show isis 1 rib redistribution level-2
IPv4 redistribution RIB for IS-IS process 1
IPV4 unicast base topology (TID 0, TOPOID 0x0) ==================
====== Level 2 ======
10.1.1.1/32
  [ISIS/20] isis prefix-SID index: 1, R:1 N:1 P:1 E:0 V:0 L:0
  flex-algo 128 SID index: 11, R:0 N:1 P:0 E:0 V:0 L:0 map 0x0
    prefix-metric: 20, advertised
```

In this command output, you can see that the prefix-metric is advertised only if it is enabled.

Flexible Algorithm Configurations

This section describes various configurations that are required to support the SR Flexible Algorithm functionality.

Table 2: Flexible Algorithm Configuration

Task	Protocol	Mode	Command
Configure Flexible Algorithm	IS-IS and OSPF	IS-IS and OSPFconfiguration sub-mode	**flex-algo** *algorithm number* *algorithm number* —value from 128 to 255
Setting metric type	IS-IS and OSPF	Flex-algo sub-mode	**[IS-IS]** **metric type {delay | te}** **Note** By default, the regular IGP metric is used. If the delay metric is enabled, the advertised delay on the link is used as the metric for flexible algorithm computation. If TE metric is enabled, the advertised TE metric in the link is used as the metric for flexible algorithm computation. **[OSPF]** **metric-type {delay | te-metric | igp-metric}**

Task	Protocol	Mode	Command
Setting affinity	IS-IS and OSPF	Flex-algo sub-mode	**[IS-IS]** **affinity {exclude-any | include-any | include-all} name** *affinity-name* **[OSPF]** **affinity {exclude-any | include-any | include-all} name** *affinity-name* *affinity-name*: Name of the affinity map
Setting priority	IS-IS and OSPF	Flex-algo sub-mode	**[IS-IS and OSPF]** **priority** *priority value* *priority-value*: Priority used during the Flexible Algorithm definition election
Enable advertisement of the Flexible Algorithm definition in IS-IS and OSPF: Affinity-map associates the name with the particular bit positions in the Extended Admin Group bitmask	IS-IS and OSPF		**[IS-IS]** **affinity-map** *affinity-name* **bit-position***bit number* **[OSPF]** **affinity-map name** *affinity-name* **bit-position***bit number* *affinity-name*: Name of the affinity map *bit number*: Bit position in the Extended Admin Group bitmask

Task	Protocol	Mode	Command
Associate affinity with an interface	IS-IS and OSPF		**[IS-IS]** **isis affinity flex-algo name** *affinity-name* **[OSPF]** **ip ospf affinity flex-algo name** *affinity-name* *affinity-name*: Name of the affinity map

From Cisco IOS XE Release 17.11.1a, a new metric, **TE**, is introduced for IS-IS Flexible Algorithm. This metric includes a new keyword for the **isis flex-algo metric-type** command.

isis *instance* **flex-algo** *algo* **metric-type {delay | te}**

This keyword is available for Cisco ASR 1000 series platforms.

Note By default, the IGP metric is used for flexible algorithm computation. If either the delay or the TE metric is enabled, the advertised delay or the TE metric in the link is used as the metric for flexible algorithm computation.

Command for Prefix SID in Flexible Algorithm Configuration

To define a prefix SID associated with a specific flexible algorithm, a new command is added under segment routing, for both connected prefix SID map, as well as the mapping server:

```
segment-routing mpls
connected-prefix-sid-map
address-family ipv4algorithmflex-algo
ip addressmask [index | absolute] sid range range of SIDs

segment-routing mpls
mapping-server
prefix-sid-map
address-family ipv4algorithmflex-algo
ip addressmask [index | absolute] sid range range of SIDs
```

Configuring IS-IS Flexible Algorithm

The following is an example of how to configure the IS-IS flexible algorithm:

```
router isis 1
 net 49.0002.0000.0001.00
 is-type level-1
 metric-style wide
 log-adjacency-changes
 nsf cisco
 distribute link-state
 segment-routing mpls
 segment-routing prefix-sid-map advertise-local
```

```
 affinity-map blue bit-position 8
 affinity-map green bit-position 201
 affinity-map red bit-position 65

 fast-reroute per-prefix level-1 all
 fast-reroute tie-break level-1 node-protecting 100
 fast-reroute tie-break level-1 srlg-disjoint 50
 fast-reroute ti-lfa level-1
 fast-reroute ti-lfa level-2
 microloop avoidance segment-routing
 microloop avoidance rib-update-delay 10000

flex-algo 129
  advertise-definition
  metric-type delay
  priority 120
  affinity
   exclude-any
    name red
   !
```



Note Use the **fast-reroute disable** command to disable TI LFA.

The following example shows how to configure the IS-IS flexible algorithm with metric-type as TE:

```
router isis 1
net 49.0002.0000.0001.00
is-type level-1
metric-style wide
log-adjacency changes
nsf cisco
distribute link-state
segment-routing mpls
segment-routing prefix-sid-map advertise-local
affinity-map blue bit-position 8
affinity-map green bit-position 201
affinity-map red bit-position 65

fast-reroute per-prefix level-1 all
fast-reroute tie-break level-1 node-protecting 100
fast-reroute tie-break level-1 srlg-disjoint 50
fast-reroute ti-lfa level-1
fast-reroute ti-lfa level-2
microloop avoidance segment-routing
microloop avoidance rib-update-delay 10000

flex-algo 129 advertise-definition
metric-type te
  priority 120
affinity exclude-any name red
!
```

The following example shows how to configure IS-IS TE metric on an interface:

```
interface Ethernet0/0
ip address 10.12.12.1 255.255.255.0
ip router isis 1
ipv6 address 2001:20::1/112
ipv6 router isis 1
isis network point-to-point
isis te-metric flex-algo 500
```

Redistributing IS-IS

The following example shows how to redistribute IS-IS:

```
router isis 2
router-id Loopback0
metric-style wide
segment-routing mpls
segment-routing prefix-sid-map advertise-local
flex-algo 128
advertise-definition
redistribute isis 1 ip level2 <-----
passive-interface Loopback0
mpls traffic-eng level-1
mpls traffic-eng level-2
```

Configuring SRTE-ODN Association

The following example shows how to configure an SR traffic engineering - ODN association:

```
segment-routing traffic-eng
 on-demand color 100
  authorize
  candidate-paths
   preference 100
    constraints
     segments
      dataplane mpls
      algorithm 129
     !
    !
    dynamic
     metric
      type delay
    !
   !
```

Configuring the Interface for Flexible Algorithm

The following example shows how to configure an interface for flexible algorithm:

```
interface GigabitEthernet0/0/6
 ip address 10.11.11.1 255.255.255.0
 ip router isis 1
 mpls ip
 mpls traffic-eng tunnels
 bfd template pw_bfd
 isis network point-to-point
 isis affinity flex-algo
  name red
  !
```

Configuring BGP

The following example shows how to configure BGP:

```
router bgp 100
 bgp router-id 10.1.1.1
```

```
 bgp log-neighbor-changes
 bgp graceful-restart
 neighbor 10.2.2.2 remote-as 100
 neighbor 10.2.2.2 ha-mode sso
 neighbor 10.2.2.2 update-source Loopback1
 !
 address-family ipv4
  neighbor 10.2.2.2 activate
 exit-address-family
 !
address-family vpnv4
  neighbor 10.2.2.2 activate
  neighbor 10.2.2.2 send-community both
  neighbor 10.2.2.2 route-map BGP_TE_MAP out
 exit-address-family
 !
 address-family ipv4 vrf SR
  redistribute connected
  neighbor 10.132.1.1 remote-as 101
  neighbor 10.132.1.1 activate
 exit-address-family
!
```

Configuring Selective Path Filtering

The following example shows how to selectively filter the paths that are installed in the MPLS Forwarding Infrastructure (MFI):

```
Prefix-source
=====================
interface Loopback1
ip address 10.1.1.1 255.255.255.255
ip router isis
 isis tag 111

Remote router configured for selective path filtering
 ================================================
route-map block deny 10
match tag 111
!
route-map block permit 100
!
router isis 1
!
flex-algo 135
!
distribute-list route-map block in
```

Configuring SR Policy with PCE Delegation

The following example shows how to configure SR policy with Path Computation Element (PCE) delegation:

```
policy p-delay
  color 1111 end-point 10.6.6.6
  candidate-paths
   preference 1
    constraints
     segments
      dataplane mpls
```

```
  algorithm 128
 !
!
dynamic
 pcep
```

Verifying the Flexible Algorithm Configuration

The following is a sample output of the **show isis flex-algo** *value* command showing all the information regarding the IS-IS flexible algorithm:

```
show isis flex-algo 129
Tag 1:
IS-IS Flex-Algo Database
 Flex-Algo count: 7

Flex-Algo 129:
    IS-IS Level-1
      Definition Priority: 222
      Definition Source: R2-RSP3-2015.00, (Local)
      Definition Equal to Local: Yes
      Definition Metric Type: Delay
      Definition Flex-Algo Prefix Metric: No
      Disabled: No
      Microloop Avoidance Timer Running: No
   Local Priority: 222
   FRR Disabled: No
   Microloop Avoidance Disabled: No
```

The following is a sample output for the **show isis flex-algo** command showing the metric type TE:

```
show isis flex-algo 129 Tag 1:
IS-IS Flex-Algo Database Flex-Algo count: 7

Flex-Algo 129:
IS-IS Level-1
Definition Priority: 222
Definition Source: R2-RSP3-2015.00, (Local) Definition Equal to Local: Yes
Definition Metric Type: TE
Definition Flex-Algo Prefix Metric: No Disabled: No
Microloop Avoidance Timer Running: No Local Priority: 222
FRR Disabled: No
Microloop Avoidance Disabled: No
```

The following is a sample output of the **show isis rib flex-algo** *value* command showing all the IS-IS local RIB information:

```
show isis rib flex-algo 129
IPv4 local RIB for IS-IS process 1

IPV4 unicast topology base (TID 0, TOPOID 0x0) ================= Repair path attributes:
DS - Downstream, LC - Linecard-Disjoint, NP - Node-Protecting PP - Primary-Path, SR -
SRLG-Disjoint

Flex-algo 129

10.1.1.1/32 prefix attr X:0 R:0 N:1 source router id: 10.1.1.1 SID index 38 - Bound
[115/L1/113] via 10.11.11.1(GigabitEthernet0/4/6) R1-ASR920-2011.00-00, from 10.1.1.1, tag
 0
LSP 6/6/351(351), prefix attr: X:0 R:0 N:1 Source router id: 10.1.1.1
Prefix-SID index: 38, R:0 N:1 P:0 E:0 V:0 L:0
```

```
label: implicit-null
repair path: 10.20.20.2 (GigabitEthernet0/4/7) metric: 117 (DS,SR) local LFA
label: implicit-null
repair source: R1-ASR920-2011, LSP 6

10.2.2.2/32 prefix attr X:0 R:0 N:1 source router id: 10.2.2.2 SID index 39 - Bound
[115/L1/24] via 10.13.13.2(GigabitEthernet0/1/5) R4-RSP3-2036.00-00, from 10.2.2.2, tag 0
LSP 2/3/345(345), prefix attr: X:0 R:0 N:1 Source router id: 10.2.2.2
Prefix-SID index: 39, R:0 N:1 P:0 E:0 V:0 L:0
label: 17039
repair path: 10.4.4.4 (MPLS-SR-Tunnel4) metric: 170 (DS,NP,SR) next-hop: 10.20.20.2
(GigabitEthernet0/4/7)
TI-LFA node/SRLG-protecting, SRLG-protecting
SRGB: 17000, range: 7000 prefix-SID index: 39, R:0 N:1 P:0 E:0 V:0 L:0
label: 17039
P node: R3-RSP2-2013[10.4.4.4], label: 17221
repair source: R6-RSP3-2038, LSP 3

10.4.4.4/32 prefix attr X:0 R:0 N:1 source router id: 10.4.4.4 SID index 221 - Bound



[115/L1/172] via 10.13.13.2(GigabitEthernet0/1/5) R4-RSP3-2036.00-00, from 10.4.4.4, tag
0
LSP 2/7/24(24), prefix attr: X:0 R:0 N:1 Source router id: 10.4.4.4
Prefix-SID index: 221, R:0 N:1 P:0 E:0 V:0 L:0
label: 17221
repair path: 10.20.20.2 (GigabitEthernet0/4/7) metric: 184 (DS,NP,SR) local LFA
label: 17221
repair source: R3-RSP2-2013, LSP 7

10.5.5.5/32 prefix attr X:0 R:0 N:1 source router id: 10.5.5.5 SID index 222 - Bound
[115/L1/17] via 10.13.13.2(GigabitEthernet0/1/5) R4-RSP3-2036.00-00, from 10.5.5.5, tag 0
LSP 2/2/347(347), prefix attr: X:0 R:0 N:1 Source router id: 10.5.5.5
Prefix-SID index: 222, R:0 N:1 P:0 E:0 V:0 L:0
label: implicit-null
repair path: 10.4.4.4 (MPLS-SR-Tunnel4) metric: 170 (DS,SR) next-hop: 10.20.20.2
(GigabitEthernet0/4/7)
TI-LFA SRLG-protecting
SRGB: 17000, range: 7000 prefix-SID index: 222, R:0 N:1 P:0 E:0 V:0 L:0
label: 17222
P node: R3-RSP2-2013[10.4.4.4], label: 17221
repair source: R4-RSP3-2036, LSP 2

10.6.6.6/32 prefix attr X:0 R:0 N:1 source router id: 10.6.6.6 SID index 333 - Bound
[115/L1/122] via 10.13.13.2(GigabitEthernet0/1/5) R4-RSP3-2036.00-00, from 10.6.6.6, tag
0
LSP 2/4/351(351), prefix attr: X:0 R:0 N:1 Source router id: 10.6.6.6
Prefix-SID index: 333, R:0 N:1 P:0 E:0 V:0 L:0
label: 17333
repair path: 10.4.4.4 (MPLS-SR-Tunnel4) metric: 170 (DS,NP,SR) next-hop: 10.20.20.2
(GigabitEthernet0/4/7)
TI-LFA node/SRLG-protecting, SRLG-protecting
SRGB: 17000, range: 7000 prefix-SID index: 333, R:0 N:1 P:0 E:0 V:0 L:0
label: 17333
P node: R3-RSP2-2013[10.4.4.4], label: 17221
repair source: R5-ASR920-2012, LSP 4
```

The following is a sample output of the **show isis topo flex-algo** *value* command showing information regarding the IS-IS paths to intermediate systems:

```
show isis topo flex-algo 129
Tag 1:
IS-IS TID 0 paths to level-1 routers
```

```
Flex-algo 129
System Id            Metric     Next-Hop            Interface   SNPA
920_1                3          RSP2_2              Gi0/15/0    e8ed.f3b8.f804
RSP3_R1              **
RSP2_1               2          RSP2_2              Gi0/15/0    e8ed.f3b8.f804
RSP3_R2              **
RSP2_2               1          RSP2_2              Gi0/15/0    e8ed.f3b8.f804
RSP3_R3              --
```

The following is a sample output of the **show isis fast-reroute ti-lfa tunnel** command showing information regarding the IS-IS TI-LFA tunnels:

```
show isis fast-reroute ti-lfa tunnel
Tag null:
Fast-Reroute TI-LFA Tunnels:
Tunnel  Interface  Next Hop         End Point        Label     End Point Host
Tag 1:
Fast-Reroute TI-LFA Tunnels:

Tunnel  Interface  Next Hop         End Point        Label     End Point Host

MP2     Gi0/0/6    10.12.12.2       10.2.2.2          17019     RSP3_R3

MP5     Gi0/0/5    10.11.11.2       10.2.2.2          17019     RSP3_R3

MP3     Gi0/0/6    10.12.12.2       10.6.6.6          17333     RSP2_2

                                    10.2.2.2          16        RSP3_R3

MP9     Gi0/0/5    10.11.11.2       10.2.2.2          17039     RSP3_R3

MP1     Gi0/0/6    10.12.12.2       10.6.6.6          20333     RSP2_2

                                    10.2.2.2          16        RSP3_R3

MP6     Gi0/0/5    10.11.11.2       10.2.2.2          17049     RSP3_R3
```

The following is a sample output of the **show ip ospf topology** command showing the node and link information compiled from the link-state advertisements (LSAs):

```
R1#show ip ospf topology
        Process OSPF-10

  Instance : global
   Router ID : 10.1.1.1
    Area :  (8 nodes)
      Node : 10.2.0.2 (pseudo) (2 links)
        Link : 10.1.1.1 10.0.0.0 Transit
        Link : 10.1.1.2 10.0.0.0 Transit
      Node : 10.1.1.1 (root) (3 links) ABR
        Algos supported: 128, 129
        Flex Algo Definition: 128
        Flex Algo Definition: 129
        Link : 10.1.1.6 10.0.0.2 Point-to-point
        Link : 10.1.1.6 10.6.1.1 Point-to-point
        Link : 10.2.0.2 10.2.0.1 Transit
      Node : 10.1.1.2  (3 links)
        Algos supported: 128
        Link : 10.1.1.3 10.3.0.2 Point-to-point
        Link : 10.1.1.54 10.5.0.2 Point-to-point
        Link : 10.2.0.2 10.2.0.2 Transit
      Node : 10.1.1.3  (2 links)
        Algos supported: 128
        Link : 10.1.1.2 10.3.0.3 Point-to-point
```

```
      Link : 10.1.1.4 10.4.0.3 Point-to-point
    Node : 10.1.1.4  (3 links) ABR, ASBR
      Algos supported: 128, 129
      Link : 10.1.1.3 10.4.0.4 Point-to-point
      Link : 10.1.1.9 10.0.0.3 Point-to-point
      Link : 10.1.1.54 10.5.0.4 Point-to-point
    Node : 10.1.1.6  (4 links)
      Algos supported: 129
      Link : 10.1.1.1 10.0.0.2 Point-to-point
      Link : 10.1.1.1 10.6.1.6 Point-to-point
      Link : 10.1.1.54 10.6.0.6 Point-to-point
      Link : 10.1.1.54 10.6.1.6 Point-to-point
    Node : 10.1.1.9  (1 links) ABR
      Link : 10.1.1.4 10.0.0.3 Point-to-point
    Node : 10.1.1.54  (4 links)
      Algos supported: 129
      Link : 10.1.1.2 10.5.0.5 Point-to-point
      Link : 10.1.1.4 10.5.0.5 Point-to-point
      Link : 10.1.1.6 10.6.0.5 Point-to-point
      Link : 10.1.1.6 10.6.1.5 Point-to-point
  Area :  (2 nodes)
    Node : 10.1.1.1 (root) (1 links) ABR
      Algos supported: 128, 129
      Flex Algo Definition: 128
      Flex Algo Definition: 129
      Link : 10.1.1.8 10.8.0.1 Point-to-point
    Node : 10.1.1.8  (1 links)  ASBR
      Link : 10.1.1.1 10.8.0.8 Point-to-point
```

The following is a sample output of the **show ip ospf topology prefix** command showing the node and prefix information compiled from the LSAs:

```
R1#show ip ospf topology prefix
        Process OSPF-10

  Instance : global
   Router ID : 10.1.1.1
    Area :  (8 nodes)
      Node : 10.2.0.2 (pseudo) (2 links)
      Node : 10.1.1.1 (root) (3 links) ABR
        Algos supported: 128, 129
        Flex Algo Definition: 128
        Flex Algo Definition: 129
      Node : 10.1.1.2  (3 links)
        Algos supported: 128
      Node : 10.1.1.3  (2 links)
        Algos supported: 128
        Prefix : 10.1.1.34/32
      Node : 10.1.1.4  (3 links) ABR, ASBR
        Algos supported: 128, 129
        Prefix : 10.1.1.4/32
        Prefix : 10.1.1.34/32
        Prefix : 10.1.1.45/32
      Node : 10.1.1.6  (4 links)
        Algos supported: 129
      Node : 10.1.1.9  (1 links) ABR
      Node : 10.1.1.54  (4 links)
        Algos supported: 129
        Prefix : 10.1.1.54/32
    Area :  (2 nodes)
      Node : 10.1.1.1 (root) (1 links) ABR
        Algos supported: 128, 129
        Flex Algo Definition: 128
        Flex Algo Definition: 129
      Node : 10.1.1.8  (1 links)  ASBR
```

The following is a sample output of the **show ip ospf topology route** command showing the path information of routes computed based on route calculation:

```
R1#show ip ospf topology route
 Route Table of OSPF-10 with router ID 10.1.1.1 (VRF global)

  10.1.1.4/32
    Algo 128, Metric 31, SID 132, Label 16132
      10.2.0.2, from 10.1.1.2, via Ethernet0/1
    Algo 129, Metric 31, SID 133, Label 16133
      10.1.1.6, from 10.1.1.6, via Ethernet0/0
      10.6.1.6, from 10.1.1.6, via Ethernet0/3
  10.1.1.34/32
    Algo 128, Metric 21, SID 43, Label 16043
      10.2.0.2, from 10.1.1.2, via Ethernet0/1
  10.1.1.45/32
    Algo 129, Metric 31, SID 4294967295, Label 1048577
      10.1.1.6, from 10.1.1.6, via Ethernet0/0
      10.6.1.6, from 10.1.1.6, via Ethernet0/3
  10.1.1.54/32
    Algo 129, Metric 21, SID 45, Label 16045
      10.1.1.6, from 10.1.1.6, via Ethernet0/0
      10.6.1.6, from 10.1.1.6, via Ethernet0/3
```

The following is a sample output of the **show mpls forwarding-table** command showing the non-zero algorithm specific prefix SID Label MPLS forwarding information:

```
#show mpls forwarding-table 10.23.23.23 255.255.255.255 algo 20
Local      Outgoing   Prefix           Bytes Label   Outgoing   Next Hop
Label      Label      or Tunnel Id     Switched      interface
18         16023      0-10.23.23.23/32-4 (10:30:20:1)   \
                                       0             Et1/1      10.1.1.2
```

The prefix or tunnel ID column provides information about the metric, for example, 0-10.6.6.6/32-4 (4:50:128:0).

The four parts next to the prefix are as follows:

- pdb-index=4
- metric=50
- algo=128
- via-srms=0

The **via-srm**s field indicates whether the source of the label came from a prefix reachability advertisement (0) or from a mapping server advertisement (1). Labels derived from mapping server advertisements should not be advertised when a redistributed route is advertised by the destination protocol for redistribution.

The **pdb-index** field indicates the protocol instance. The following command output shows the different protocols and their values:

```
# show ip protocols summary
Index Process Name
0 connected
1 static
2 application
3 nat-route
4 isis 1
```

The following is a sample output of the **show isis rib redistribution** command showing the redistributed prefix:

```
# show isis rib redistribution

IPv4 redistribution RIB for IS-IS process 1

IPV4 unicast base topology (TID 0, TOPOID 0x0) =================
====== Level 1 ======
====== Level 2 ======
10.3.3.3/32
  [Connected/0]  prefix-SID index: 31, R:0 N:1 P:0 E:0 V:0 L:0
  strict-SPF SID index: 32, R:0 N:1 P:0 E:0 V:0 L:0
  flex-algo 128 SID index: 33, R:0 N:1 P:0 E:0 V:0 L:0 map 0x1
    prefix-metric: 0, not advertised
10.4.4.4/32
  [ISIS/0] external interarea  prefix-SID index: 41, R:1 N:0 P:1 E:0 V:0 L:0
  strict-SPF SID index: 42, R:1 N:0 P:1 E:0 V:0 L:0
  flex-algo 128 SID index: 43, R:1 N:0 P:1 E:0 V:0 L:0 map 0x0
    prefix-metric: 40, not advertised
  prefix attr: X:1 R:0 N:0
```

In this example, you can see the strict SID or flexible algorithm prefix SIDs. The redistributed prefix is noted as the inter area route, and the X flag is set.

The following is a sample output of the **show isis database verbose** command showing the prefix-metric flag (M-flag) that is advertised in ISIS flexible algorithm definition flags sub-TLV:

```
# show isis database verbose
..
Router CAP:   10.1.1.1, D:0, S:0
    Segment Routing: I:1 V:0, SRGB Base: 16000 Range: 8000
    Segment Routing Local Block: SRLB Base: 15000 Range: 1000
    Node-MSD
      MSD: 16
    Flex algorithm: 150 Metric-Type: IGP Alg-type: SPF Priority: 128
    Segment Routing Algorithms: SPF, Strict-SPF, Flex-algo 128
    Segment Routing Algorithms: Flex-algo 150
    Flex algorithm: 128 Metric-Type: IGP Alg-type: SPF Priority: 128
      Flex-Algo Definition Flags:
       M:1.
```