Understand BGP Dynamic Segment Routing Traffic Engineering

Contents

Introduction

This document describes how to understand, configure, and verify the BGP Dynamic Segment Routing Traffic Engineering (SR-TE) feature in Cisco IOS® XR.

Prerequisites

There are no prerequisites for this document.

Requirements

There are no specific requirements for this document.

Components Used

The information in this document is based on Cisco IOS XR and Cisco IOS XE.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Background Information

SR-TE provides the capabilities to steer traffic through an SR-enabled core without state creation and maintenance (stateless). An SR-TE policy is expressed as a list of segments that specifies a path, called Segment ID (SID) list. No signaling is required as state is in the packet and SID list are processed as a set of instructions by the transit routers.

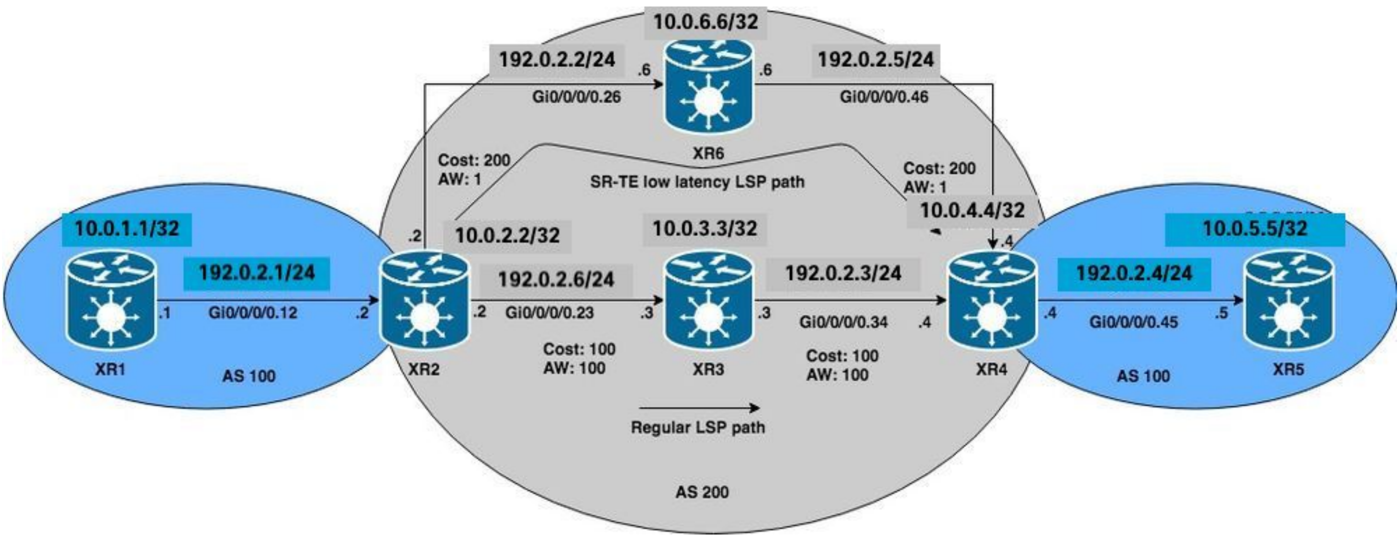
With Dynamic Border Gateway Protocol (BGP) SR-TE you can generate automatic SR-TE policies based on arbitrary criteria such as communities signaled by a router participating in a Segment Routing network. In order to be able to meet service level assurance (SLAs) of site's applications and compute paths based on specific requirements, you can generate automatic SR-TE policies for a given IP subnet or services by setting communities and triggering these policies .

> **Note**: Matching criteria other than communities is also supported to create dynamic SR-TE policies.

A common application for this feature is in MPLS L3VPN environments, where the network administrator can trigger automatic SR-TE tunnel policies to route traffic based on specific constraints (delay, bandwidth, and so on). For the demonstrations in this document, we create a L3VPN service connecting XR1 and XR5 and trigger auto-tunnels on XR2 (headend) based on a particular community set on XR4 (tail end) on MP-BGP.

Configure

Network Diagram



Initial Configurations

L3VPN, Segment Routing, and SR-TE basic configurations have been enabled.

```
XR1
hostname XR1
logging console debugging
interface Loopback0
 ipv4 address 10.0.1.1 255.255.255.255
!
interface GigabitEthernet0/0/0/0.12
 ipv4 address 192.0.2.1 255.255.255.0
 encapsulation dot1q 12
!
route-policy PASS
  pass
end-policy
!
```

```
router bgp 100
 bgp router-id 10.0.1.1
 address-family ipv4 unicast
  network 10.0.1.1/32
 !
 neighbor 192.0.2.7
  remote-as 200
  address-family ipv4 unicast
   route-policy PASS in
   route-policy PASS out
  !
 !
!
end
```

XR2

hostname XR2 logging console debugging vrf BLUE address-family ipv4 unicast import route-target 1:1 ! export route-target 1:1 ! ! ! interface Loopback0 ipv4 address 10.0.2.2 255.255.255.255 ! interface GigabitEthernet0/0/0/0.12 vrf BLUE ipv4 address 192.0.2.7 255.255.255.0 encapsulation dot1q 12 ! interface GigabitEthernet0/0/0/0.23 ipv4 address 192.0.2.8 255.255.255.0 encapsulation dot1q 23 ! interface GigabitEthernet0/0/0/0.26 ipv4 address 192.0.2.9 255.255.255.0 encapsulation dot1q 26 ! route-policy PASS pass end-policy ! ! router ospf 1 segment-routing mpls segment-routing forwarding mpls segment-routing sr-prefer address-family ipv4 area 0 mpls traffic-eng interface Loopback0 prefix-sid index 2 ! interface GigabitEthernet0/0/0/0.23 cost 100 network point-to-point ! interface GigabitEthernet0/0/0/0.26 cost 200 network point-to-point ! ! mpls traffic-eng router-id Loopback0 ! router bgp 100 bgp router-id 10.0.2.2 address-family vpnv4 unicast ! neighbor 10.0.4.4 remote-as 200 update-source Loopback0 address-family vpnv4 unicast ! ! vrf BLUE rd 1:1 address-family ipv4 unicast ! neighbor 192.0.2.10 remote-as 200 address-family ipv4 unicast route-policy PASS in route-policy PASS out as-override ! ! ! ! mpls oam ! mpls traffic-eng interface GigabitEthernet0/0/0/0.23 admin-weight 100 ! interface GigabitEthernet0/0/0/0.26 admin-weight 1 ! ! end

XR3

hostname XR3 logging console debugging interface Loopback0 ipv4 address 10.0.3.3 255.255.255.255 ! ! interface GigabitEthernet0/0/0/0.23 ipv4 address 192.0.2.11 255.255.255.0 encapsulation dot1q 23 ! interface GigabitEthernet0/0/0/0.34 ipv4 address 192.0.2.12 255.255.255.0 encapsulation dot1q 34 ! router ospf 1 segment-routing mpls segment-routing forwarding mpls segment-routing sr-prefer address-family ipv4 area 0 mpls traffic-eng interface Loopback0 prefix-sid index 3 ! interface GigabitEthernet0/0/0/0.23 cost 100 network point-to-point ! interface GigabitEthernet0/0/0/0.34 cost 100 network point-to-point ! ! mpls traffic-eng router-id Loopback0 ! mpls oam ! mpls traffic-eng interface GigabitEthernet0/0/0/0.23 admin-weight 100 ! interface GigabitEthernet0/0/0/0.34 admin-weight 100 ! ! end

XR4

hostname XR4 logging console debugging vrf BLUE address-family ipv4 unicast import route-target 1:1 ! export route-target 1:1 ! ! ! interface Loopback0 ipv4 address 10.0.4.4 255.255.255.255 ! interface GigabitEthernet0/0/0/0.34 ipv4 address 192.0.2.13 255.255.255.0 encapsulation dot1q 34 ! interface GigabitEthernet0/0/0/0.45 vrf BLUE ipv4 address 192.0.2.14 255.255.255.0 encapsulation dot1q 45 ! interface GigabitEthernet0/0/0/0.46 ipv4 address 192.0.2.15 255.255.255.0 encapsulation dot1q 46 ! route-policy PASS pass end-policy ! ! router ospf 1 segment-routing mpls segment-routing forwarding mpls segment-routing sr-prefer address-family ipv4 area 0 mpls traffic-eng interface Loopback0 prefix-sid index 4 ! interface GigabitEthernet0/0/0/0.34 cost 100 network point-to-point ! interface GigabitEthernet0/0/0/0.46 cost 200 network point-to-point ! ! mpls traffic-eng router-id Loopback0 ! router bgp 100 bgp router-id 10.0.4.4 address-family vpnv4 unicast ! neighbor 10.0.2.2 remote-as 200 update-source Loopback0 address-family vpnv4 unicast ! ! vrf BLUE rd 1:1 bgp unsafe-ebgp-policy address-family ipv4 unicast ! neighbor 192.0.2.16 remote-as 200 address-family ipv4 unicast route-policy PASS in route-policy PASS out as-override ! ! ! ! mpls oam ! mpls traffic-eng interface GigabitEthernet0/0/0/0.34 admin-weight 100 ! interface GigabitEthernet0/0/0/0.46 admin-weight 1 ! ! end

XR5

hostname XR5

```
logging console debugging
interface Loopback0
description REGULAR LSP PATH ipv4 address 10.0.5.5 255.255.255.255 ! interface Loopback1
description DELAY SENSITIVE - LOW LATENCY PATH (1:1) ipv4 address 10.0.5.55 255.255.255.255 !
interface GigabitEthernet0/0/0/0.45 ipv4 address 192.0.2.16 255.255.255.0 encapsulation dot1q 45
! route-policy PASS pass end-policy ! router bgp 100 bgp router-id 10.0.5.5 bgp unsafe-ebgp-
policy address-family ipv4 unicast network 10.0.5.5/32 network 10.0.5.55/32 ! neighbor
192.0.2.14 remote-as 200 address-family ipv4 unicast route-policy PASS in route-policy PASS out
! ! ! mpls oam ! end
```

XR6

```
hostname XR6 logging console debugging interface Loopback0 ipv4 address 10.0.6.6 255.255.255.255
! interface GigabitEthernet0/0/0/0.26 ipv4 address 192.0.2.17 255.255.255.0 encapsulation dot1q
26 ! interface GigabitEthernet0/0/0/0.46 ipv4 address 192.0.2.18 255.255.255.0 encapsulation
dot1q 46 ! router ospf 1 segment-routing mpls segment-routing forwarding mpls segment-routing
sr-prefer address-family ipv4 area 0 mpls traffic-eng interface Loopback0 prefix-sid index 6 !
interface GigabitEthernet0/0/0/0.26 cost 200 network point-to-point ! interface
GigabitEthernet0/0/0/0.46 cost 200 network point-to-point ! ! mpls traffic-eng router-id
Loopback0 ! mpls oam ! mpls traffic-eng interface GigabitEthernet0/0/0/0.26 admin-weight 1 !
interface GigabitEthernet0/0/0/0.46 admin-weight 1 ! ! end
```

XR2 and XR4 (PEs) have built an LSP using Segment Routing, this can be verified by using MPLS ping for the corresponding Segment Routing FEC. For this scenario, there are two possible paths to transport the L3VPN traffic from XR1 to XR5:

Regular LSP path: XR1 > XR2 > **XR3** > XR4 > XR5

Low latency LSP path: XR1 > XR2 >**XR6** > XR4 > XR5

Initially, all traffic between XR1 and XR5 is routed through XR3 via the regular LSP path due to lower IGP cost, we can confirm both LSPs and connectivity as per theseverifications. IGP cost to reach XR4 from XR2 via XR3 is 201 versus 401 via XR6. Even though the path via XR3 has a better path metric, low latency services on VRF BLUE must be routed through the path via XR6.

```
RP/0/0/CPU0:XR2#ping mpls ipv4 10.0.4.4/32 fec-type generic verbose

Sending 5, 100-byte MPLS Echos to 10.0.4.4/32,
      timeout is 2 seconds, send interval is 0 msec:

Codes: '!' - success, 'Q' - request not sent, '.' - timeout,
  'L' - labeled output interface, 'B' - unlabeled output interface,
  'D' - DS Map mismatch, 'F' - no FEC mapping, 'f' - FEC mismatch,
  'M' - malformed request, 'm' - unsupported tlvs, 'N' - no rx label,
  'P' - no rx intf label prot, 'p' - premature termination of LSP,
  'R' - transit router, 'I' - unknown upstream index,
  'X' - unknown return code, 'x' - return code 0

Type escape sequence to abort.

!       size 100, reply addr 192.0.2.13, return code 3
!       size 100, reply addr 192.0.2.13, return code 3
!       size 100, reply addr 192.0.2.13, return code 3
!       size 100, reply addr 192.0.2.13, return code 3
!       size 100, reply addr 192.0.2.13, return code 3

Success rate is 100 percent (5/5), round-trip min/avg/max = 1/4/10 ms
```

Note: When using ping MPLS application in Segment Routing we must use Nil-FEC or generic FEC.

If you verify the L3VPN services on XR1, you can confirm reachability to XR5 loopback 10.0.5.5/32 and 10.0.5.55/32 respectively via the regular LSP path. Basic L3VPN services are enabled in the SR MPLS core.

```
RP/0/0/CPU0:XR1#ping 10.0.5.5 source 10.0.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.5.5, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/7/9 ms

RP/0/0/CPU0:XR1#ping 10.0.5.55 source 10.0.1.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.0.5.55, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/7/9 ms


RP/0/0/CPU0:XR1#traceroute 10.0.5.5 source 10.0.1.1

Type escape sequence to abort.
Tracing the route to 10.0.5.5

 1  192.0.2.7 9 msec  0 msec  0 msec
 2  192.0.2.11 [MPLS: Labels 16004/24002 Exp 0] 0 msec  0 msec  0 msec
 3  192.0.2.13 [MPLS: Label 24002 Exp 0] 0 msec  0 msec  0 msec
 4  192.0.2.16 0 msec  *  0 msec

RP/0/0/CPU0:XR1#traceroute 10.0.5.55 source 10.0.1.1

Type escape sequence to abort.
Tracing the route to 10.0.5.55

 1  192.0.2.7 9 msec  0 msec  0 msec
 2  192.0.2.11 [MPLS: Labels 16004/24005 Exp 0] 0 msec  0 msec  0 msec
 3  192.0.2.13 [MPLS: Label 24005 Exp 0] 0 msec  0 msec  0 msec
 4  192.0.2.16 0 msec  *  0 msec
```

As observed, all traffic on VRF BLUE goes through the regular LSP path XR1 > XR2 > XR3 > XR4 > XR5.

Configure BGP Dynamic SR-TE

For this example, configure XR4 (tail end) to insert community 1:1 and send it to XR2 to signal the creation of an SR-TE policy for prefix 10.0.5.55/32 on VRF BLUE. SR-TE policy path selection will be set to take the low latency path instead of the regular LSP, do this by selecting the lowest TE metric (Admin Weight) via XR6. Total TE metric (admin weight) via XR6 is 2, as admin weights have been set to 1 on outgoing interfaces towards XR4 (tail end) via XR6 as seen in the reference topology diagram and initial configurations.

In order to create the dynamic SR-TE policies, we need to configure what loopback will be used as source and what is the dynamic tunnel range that the headend will use generate the tunnels, this configuration is required at the headend of the SR-TE policy XR2. set the tunnel range to a minimum of 500 and a maximum of 500, effectively creating a single SR-TE tunnel and the source loopback to loopback 0 at the headend for the tunnel.

```
XR2
ipv4 unnumbered mpls traffic-eng Loopback0
mpls traffic-eng
 auto-tunnel p2p
  tunnel-id min 500 max 500
 !
!
end
```

On XR4, set the community 1:1 and apply it on the VRF BLUE prefix 10.0.5.55/32, this will allow it to insert the community in the BGP update.

```
XR4
route-policy COMMUNITY_1:1
  # 1:1 Community
  if destination in (10.0.5.55/32) then
    set community (1:1)
  endif
  pass
end-policy
!
router bgp 100
 vrf BLUE
 !
  neighbor 192.0.2.16
  address-family ipv4 unicast
    route-policy COMMUNITY_1:1 in
 !
!
end
```

Verifying XR2 (headend) we can see it has the community 1:1 set on the VPNv4 updates received from XR4.

```
RP/0/0/CPU0:XR2#show bgp vrf BLUE 10.0.5.55/32 detail
BGP routing table entry for 10.0.5.55/32, Route Distinguisher: 1:1 Versions: Process bRIB/RIB
SendTblVer Speaker 36 36 Flags: 0x00043001+0x00000200; Last Modified: Nov 23 17:50:59.798 for
00:02:53 Paths: (1 available, best #1) Advertised to CE peers (in unique update groups):
192.0.2.10 Path #1: Received by speaker 0 Flags: 0x4000000085060005, import: 0x9f Advertised to
CE peers (in unique update groups): 192.0.2.10 200 10.0.4.4 (metric 201) from 10.0.4.4
(10.0.4.4) Received Label 24005 Origin IGP, metric 0, localpref 100, valid, internal, best,
group-best, import-candidate, imported Received Path ID 0, Local Path ID 0, version 36
Community: 1:1
      Extended community: RT:1:1
      Source AFI: VPNv4 Unicast, Source VRF: BLUE, Source Route Distinguisher: 1:1
```

On XR2 (headend) create an RPL route policy matching the community 1:1 and setting the corresponding attribute-set for MPLS traffic-engineering. After the policy is set, we can go to the MPLS-TE configuration stanza and set the corresponding attribute-set for the SR-TE policy and indicate what is the path selection criteria, which are Segment Routing and TE metric in this case since we want to choose the path via the lowest administrative weight via XR6.

```
XR2
route-policy DYN_BGP_SR-TE
  # Matches community 1:1
  if community matches-every (1:1) then
    set mpls traffic-eng attributeset DYN_SR-TE_POLICIES
  endif
```

```
  pass
end-policy
!
router bgp 100
!
 neighbor 10.0.4.4
 address-family vpnv4 unicast
   route-policy DYN_BGP_SR-TE in
  !
mpls traffic-eng
 attribute-set p2p-te DYN_SR-TE_POLICIES
  path-selection
   metric te
   segment-routing adjacency unprotected
  !
end
```

Verify

Once completed, you can observe that tunnel-te 500 interface has been dynamically created for the specified range.

```
RP/0/0/CPU0:XR2#show mpls traffic-eng tunnels segment-routing tabular

           Tunnel   LSP     Destination          Source    Tun    FRR  LSP  Path
             Name    ID         Address         Address  State  State Role  Prot
----------------- ----- --------------- --------------- ------ ------ ---- -----
    ^tunnel-te500     2         10.0.4.4         10.0.2.2     up  Inact Head Inact
^ = automatically created P2P/P2MP tunnel
```

BGP RIB indicates that the "DYN_SR-TE_POLICIES" policy is attached to the prefix, which means traffic must be routed according to the policy.

```
RP/0/0/CPU0:XR2#show bgp vrf BLUE

Status codes: s suppressed, d damped, h history, * valid, > best
              i - internal, r RIB-failure, S stale, N Nexthop-discard
Origin codes: i - IGP, e - EGP, ? - incomplete
   Network            Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 1:1 (default for vrf BLUE)
*> 10.0.1.1/32        192.0.2.10                0             0 200 i
*>i10.0.5.5/32        10.0.4.4                 0    100      0 200 i
*>i10.0.5.55/32       10.0.4.4 T:DYN_SR-TE_POLICIES
                                             0    100      0 200 i
```

If we verify the BGP RIB for the prefix 10.0.5.55/32 in detail we can see the control plane information that will be referenced to generate the SR-TE tunnel.

```
RP/0/0/CPU0:XR2#show bgp vrf BLUE 10.0.5.55/32 detail

BGP routing table entry for 10.0.5.55/32, Route Distinguisher: 1:1
Versions:
  Process           bRIB/RIB  SendTblVer
  Speaker                 39          39
    Flags: 0x00041001+0x00000200;
```

```
Last Modified: Nov 23 17:55:22.798 for 00:04:43
Paths: (1 available, best #1)
  Advertised to CE peers (in unique update groups):
    192.0.2.10
  Path #1: Received by speaker 0
  Flags: 0x4000000085060005, import: 0x9f
  Advertised to CE peers (in unique update groups):
    192.0.2.10
  200
    10.0.4.4 T:DYN_SR-TE_POLICIES (metric 201) from 10.0.4.4 (10.0.4.4)
      Received Label 24005
      Origin IGP, metric 0, localpref 100, valid, internal, best, group-best, import-candidate,
imported
      Received Path ID 0, Local Path ID 0, version 39
      Community: 1:1
      Extended community: RT:1:1
      TE tunnel attribute-set DYN_SR-TE_POLICIES, up, registered, binding-label 24000, if-handle
0x00000130

      Source AFI: VPNv4 Unicast, Source VRF: BLUE, Source Route Distinguisher: 1:1
```

We can see that the tunnel policy is in **up** state and **registered**. The binding SID assigned is 24000, this binding SID can be used to verify what tunnel is used for this particular prefix. As observed before, tunnel-te500 was created and installed in the LFIB.

```
RP/0/0/CPU0:XR2#show mpls forwarding labels 24000 detail
Local Outgoing Prefix Outgoing Next Hop Bytes Label Label or ID Interface Switched ------ ------
----- ------------------ ------------ --------------- ------------ 24000  Pop         No ID
tt500        point2point     0
     Updated: Nov 23 17:55:23.267
     Label Stack (Top -> Bottom): { }
     MAC/Encaps: 0/0, MTU: 0
     Packets Switched: 0
```

> **Note**: Binding SID has many use cases, for this particular document, limit its use for local verification, but its application is much broader.

Alternatively you can use the given **if-handle 0x00000130** from the BGP RIB output to check the SR-TE policy assigned for prefix 10.0.5.55/32.

```
RP/0/0/CPU0:XR2#show mpls forwarding tunnels ifh 0x00000130 detail
Tunnel Outgoing Outgoing Next Hop Bytes Name Label Interface Switched ------------- -----------
------------ --------------- ------------ tt500        (SR) 24003       Gi0/0/0/0.26 192.0.2.17
0
     Updated: Nov 23 17:55:23.267
     Version: 138, Priority: 2
     Label Stack (Top -> Bottom): { 24003 }
     NHID: 0x0, Encap-ID: N/A, Path idx: 0, Backup path idx: 0, Weight: 0
     MAC/Encaps: 18/22, MTU: 1500
     Packets Switched: 0

  Interface Name: tunnel-te500, Interface Handle: 0x00000130, Local Label: 24001
  Forwarding Class: 0, Weight: 0
  Packets/Bytes Switched: 0/0
```

SR-TE policy on XR2 (headend) will have these properties from a control plane and data plane perspective to forward traffic. Also state information of the SR-TE tunnel can be seen as per output, which must match with previous verifications.

```
RP/0/0/CPU0:XR2#show mpls traffic-eng tunnels segment-routing p2p 500

Name: tunnel-te500  Destination: 10.0.4.4  Ifhandle:0x130 (auto-tunnel for BGP default)
  Signalled-Name: auto_XR2_t500
  Status:
    Admin:    up Oper:   up   Path:  valid   Signalling: connected

    path option 10, (Segment-Routing) type dynamic  (Basis for Setup, path weight 2)
    G-PID: 0x0800 (derived from egress interface properties)
    Bandwidth Requested: 0 kbps  CT0
    Creation Time: Fri Nov 23 17:55:23 2018 (00:09:01 ago)
  Config Parameters:
    Bandwidth:        0 kbps (CT0) Priority:  7  7 Affinity: 0x0/0x0
    Metric Type: TE (interface)
    Path Selection:
      Tiebreaker: Min-fill (default)
      Protection: Unprotected Adjacency
    Hop-limit: disabled
    Cost-limit: disabled
    Path-invalidation timeout: 10000 msec (default), Action: Tear (default)
    AutoRoute: disabled  LockDown: disabled   Policy class: not set
    Forward class: 0 (default)
    Forwarding-Adjacency: disabled
    Autoroute Destinations: 0
    Loadshare:          0 equal loadshares
    Auto-bw: disabled
    Path Protection: Not Enabled
    Attribute-set: DYN_SR-TE_POLICIES (type p2p-te)
    BFD Fast Detection: Disabled
    Reoptimization after affinity failure: Enabled
    SRLG discovery: Disabled
  History:
    Tunnel has been up for: 00:09:01 (since Fri Nov 23 17:55:23 UTC 2018)
    Current LSP:
      Uptime: 00:09:01 (since Fri Nov 23 17:55:23 UTC 2018)
    Reopt. LSP:
      Last Failure:
        LSP not signalled, identical to the [CURRENT] LSP
        Date/Time: Fri Nov 23 17:56:53 UTC 2018 [00:07:31 ago]

  Segment-Routing Path Info (OSPF 1 area 0)
    Segment0[Link]: 192.0.2.9 - 192.0.2.17, Label: 24005
    Segment1[Link]: 192.0.2.18 - 192.0.2.15, Label: 24003
Displayed 1 (of 1) heads, 0 (of 0) midpoints, 0 (of 0) tails
Displayed 1 up, 0 down, 0 recovering, 0 recovered heads
```

Check the prefix directly on VRF BLUE RIB, we can confirm that binding SID 24000 was assigned to the prefix.

```
RP/0/0/CPU0:XR2#show route vrf BLUE 10.0.5.55/32 detail

Routing entry for 10.0.5.55/32
  Known via "bgp 100", distance 200, metric 0
  Tag 200, type internal
  Installed Nov 23 17:55:23.267 for 00:10:38
  Routing Descriptor Blocks
    10.0.4.4, from 10.0.4.4
      Nexthop in Vrf: "default", Table: "default", IPv4 Unicast, Table Id: 0xe0000000
```

```
      Route metric is 0
      Label: 0x5dc5 (24005)
      Tunnel ID: None
      Binding Label: 0x5dc0 (24000)
      Extended communities count: 0
      Source RD attributes: 0x0000:1:1
      NHID:0x0(Ref:0)
  Route version is 0x5 (5)
  No local label
  IP Precedence: Not Set
  QoS Group ID: Not Set
  Flow-tag: Not Set
  Fwd-class: Not Set
  Route Priority: RIB_PRIORITY_RECURSIVE (12) SVD Type RIB_SVD_TYPE_REMOTE
  Download Priority 3, Download Version 27
  No advertising protos.
```

FIB for VRF BLUE indicates that forwarding for this prefix is done via tunnel-te 500 according to our BGP dynamic SR-TE policy.

```
RP/0/0/CPU0:XR2#show cef vrf BLUE 10.0.5.55/32 detail
10.0.5.55/32, version 27, internal 0x1000001 0x0 (ptr 0xa142a574) [1], 0x0 (0x0), 0x208
(0xa159d208) Updated Nov 23 17:55:23.287 Prefix Len 32, traffic index 0, precedence n/a,
priority 3 gateway array (0xa129f23c) reference count 1, flags 0x4038, source rib (7), 0 backups
[1 type 1 flags 0x48441 (0xa15b780c) ext 0x0 (0x0)] LW-LDI[type=0, refc=0, ptr=0x0, sh-ldi=0x0]
gateway array update type-time 1 Nov 23 17:55:23.287 LDI Update time Nov 23 17:55:23.287 via
local-label 24000, 3 dependencies, recursive [flags 0x6000]     path-idx 0 NHID 0x0 [0xa1605bf4
0x0]
    recursion-via-label
    next hop VRF - 'default', table - 0xe0000000
    next hop via 24000/0/21
     next hop tt500        labels imposed {ImplNull 24005}


    Load distribution: 0 (refcount 1)

    Hash  OK  Interface                 Address
    0     Y   Unknown                   24000/0
```

On XR1 we can verify connectivity and confirm that traffic is going through tunnel-te 500 via low latency path via XR6.

```
RP/0/0/CPU0:XR1#traceroute 10.0.5.55 source 10.0.1.1

Type escape sequence to abort.
Tracing the route to 10.0.5.55

 1  192.0.2.7 0 msec  0 msec  0 msec
 2  192.0.2.17 [MPLS: Labels 24003/24005 Exp 0] 0 msec  0 msec  0 msec
 3  192.0.2.15 [MPLS: Label 24005 Exp 0] 0 msec  0 msec  0 msec
 4  192.0.2.16 0 msec  *  9 msec
```

XR2 counters increase for the tunnel-te500 which corresponds to our SR-TE policy.

```
RP/0/0/CPU0:XR2#show mpls forwarding tunnels

Tunnel         Outgoing    Outgoing     Next Hop        Bytes
```

```
Name          Label       Interface                     Switched
------------- ----------- ------------ -------------- ------------
tt500        (SR) 24003       Gi0/0/0/0.26 192.0.2.17        2250
```

Path for prefix 10.0.5.5/32 is still going through the regular LSP path via XR3 as seen below.

```
RP/0/0/CPU0:XR1#traceroute 10.0.5.5 source 10.0.1.1

Type escape sequence to abort.
Tracing the route to 10.0.5.5

 1  192.0.2.7 0 msec  0 msec  0 msec
 2  192.0.2.11 [MPLS: Labels 16004/24002 Exp 0] 0 msec  0 msec  0 msec
 3  192.0.2.13 [MPLS: Label 24002 Exp 0] 0 msec  0 msec  0 msec
 4  192.0.2.16 0 msec  *  0 msec
```

Troubleshoot

There is currently no specific troubleshooting information available for this configuration.

Summary

BGP Dynamic SR-TE offers granularity and automatic enforcing of routing policies for the purpose of traffic engineering in the SR enabled core. Automatic tunnel creation can be triggered based on arbitrary criteria, which can allow network administrators to easily create traffic patterns that meet site's application requirements.