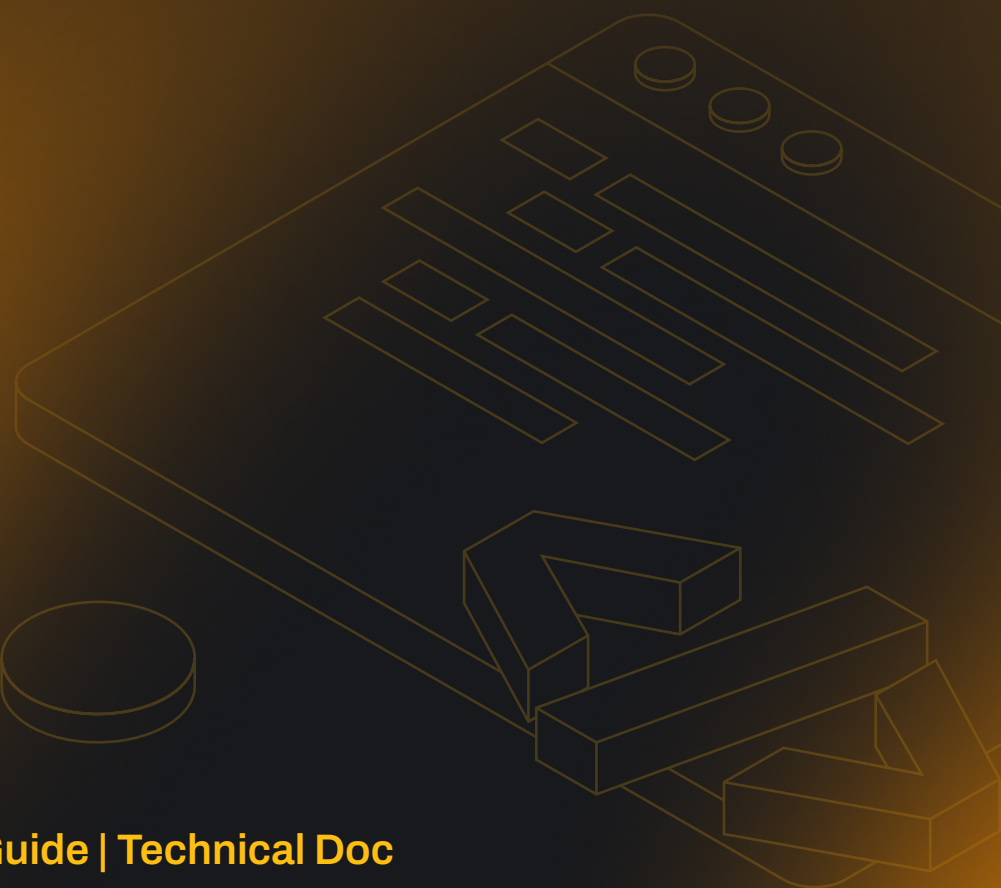




Deployment Guide | Technical Doc

SEGMENT ROUTING - TE USING SDN CONTROLLER - TRAFFIC DICTATOR

Index:

Introduction	3
Centralized SR-TE	3
SR-TE Considerations	3
Scenario/Topology	4
Configurations and Deployment	5
BGP-LU policy with dynamic path (BLUE/YELLOW)	5
Traffic Dictator Policies/Mapping traffic	12
Validation and Troubleshooting	14
What happens if the Traffic Dictator goes down?	18
Disjointness (Disjoint Group)	20
Conclusion	27



Introduction

In this document we are going to talk about SR-TE on VyOS, VyOS supports modern traffic engineering architectures based on Segment Routing (SR). Segment Routing Traffic Engineering (SR-TE) provides deterministic path control without the signaling complexity of traditional RSVP-TE, enabling simpler and more scalable network designs.

Centralized SR-TE

In centralized SR-TE, path computation and segment list generation are performed by an external controller. In this model, the router is responsible for the following functions:

- Advertising the IGP topology to the controller
- Receiving calculated SR-TE policies from the controller
- Steering traffic into the appropriate SR-TE policies

Although centralized SR-TE is simpler than RSVP-TE, the router must support several protocols and mechanisms, including:

- **BGP-LS** for topology advertisement
- **BGP SR-TE** or **PCEP** for receiving SR-TE policies
- Automated traffic steering

External controller, this case is important who defines our policies and give our routers the path to follow, this is covered by the traffic dictator, which we talked about later.

SR-TE Considerations

Segment Routing provides operational and scalability advantages over RSVP-TE; however, not all Segment Routing implementations fully support SR-TE. VyOS focuses on standards-based protocol support and interoperability, allowing SR-TE to be integrated into both distributed and controller-based traffic engineering architectures.

 **Note:** This guide was created and validated using the VyOS rolling release **VyOS 2026.09.03-1952-rolling**.



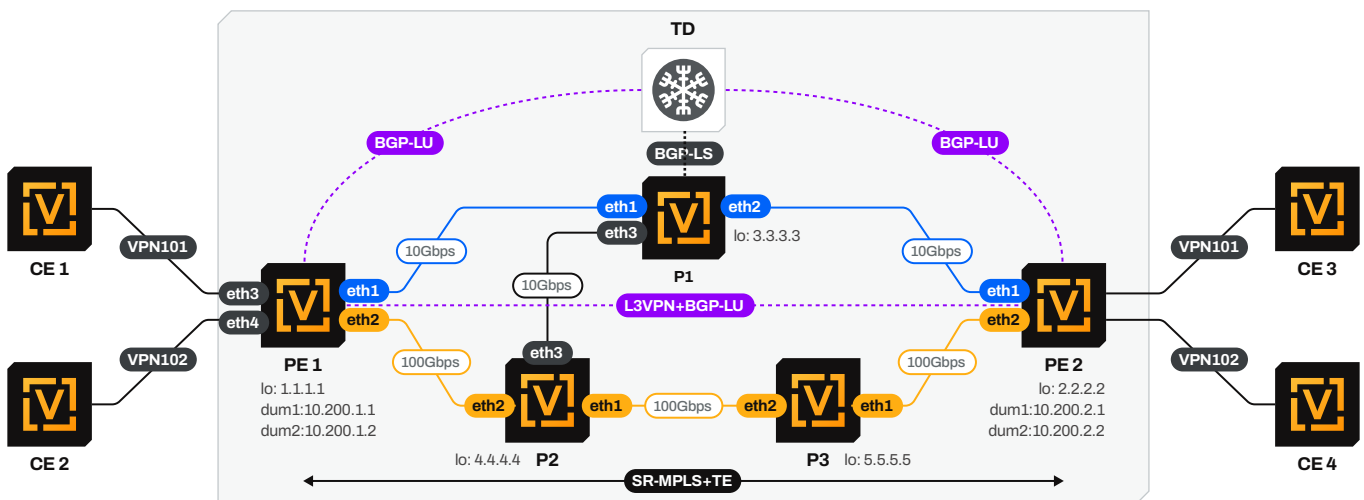
Scenario/Topology

This lab demonstrates a simple scenario in which multiple VRFs are configured on the same Provider Edge (PE) routers, a common use case in ISP environments.

On PE1 and PE2, two VRFs are configured: **VPN101** and **VPN102**.

- **CE1** and **CE3** belong to VRF **VPN101**
- **CE2** and **CE4** belong to VRF **VPN102**

Traffic engineering is used to steer traffic from each VRF over different network paths.



The primary objective of this design is to simplify the topology in order to support multiple types of use cases. As illustrated in the topology, there are two distinct paths in which affinity can be applied. Specifically, the BLUE path can be used to associate VPN101 with the service loopback interface (dum1), while the YELLOW path can be used to associate VPN102 with service loopback interface (dum2).

The goal is to deploy traffic engineering using affinity and bandwidth constraints, map different types of traffic to different policies, and ensure there is a predictable failure scenario if some links/router fail, or if the controller fails completely.

Traffic Dictator is used as a controller. It is easy to deploy and configure, and supports the most basic SR implementations.

⚠ This guide is not focused on installation of Traffic Dictator, we are to apply specific configuration to our use cases. This guide focuses only on applying specific configurations for our use cases and does not cover the installation of Traffic Dictator.

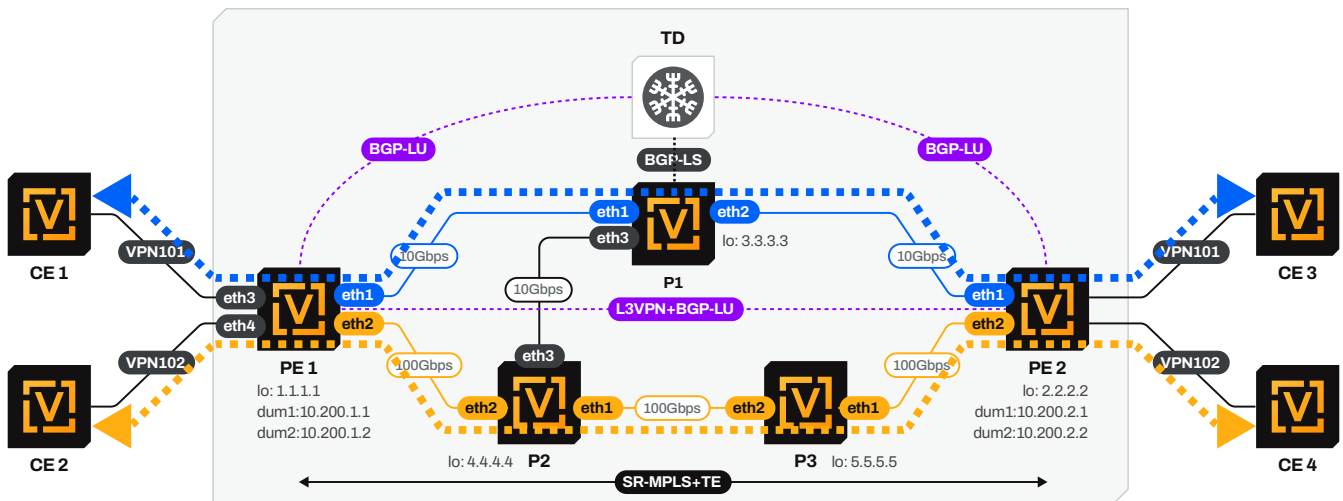
For more information about the deployment of Traffic Dictator, refer to

<https://vegvisir.ie/installation-and-upgrade/>



Configurations and Deployment

BGP-LU policy with dynamic path (BLUE/YELLOW)



In this topology, PE2 has a “service-loopback” 10.200.2.1. It is not advertised into IGP but advertised into BGP-LU with nexthop of 2.2.2.2 which is the main PE2 loopback reachable via IGP.

Traffic Dictator sends a BGP-LU route to PE1, with prefix 10.200.2.1. and nexthop of P1 interface address. This way, the same behaviour as automated steering can be achieved even if routers do not support BGP SR-TE.

This example prefix that we received from CE3, it will use the next-hop 10.200.2.1 goes through eth1 which belongs to P1

In the yellow scenario, PE2 has a **service loopback** at 10.200.2.2. This address is not injected into the IGP; instead, it is announced via BGP-LU with a next hop of 2.2.2.2, which corresponds to PE2’s primary loopback and is reachable through the IGP. As a result, VPN102 traffic will follow the **YELLOW** path.

Basic IGP configuration (CORE):

VyOS:

```
P2:

set system host-name P2
set interfaces ethernet eth1 address '10.100.4.4/24'
set interfaces ethernet eth2 address '10.100.2.4/24'
set interfaces ethernet eth3 address '10.100.10.4/24'
set interfaces loopback lo address '4.4.4.4/32'
set protocols traffic-engineering admin-group YELLOW bit-position '1'
set protocols traffic-engineering interface eth1 admin-group 'YELLOW'
set protocols traffic-engineering interface eth1 max-bandwidth '100000'
set protocols traffic-engineering interface eth1 max-reservable-bandwidth '100000'
set protocols traffic-engineering interface eth2 admin-group 'YELLOW'
set protocols traffic-engineering interface eth2 max-bandwidth '100000'
set protocols traffic-engineering interface eth2 max-reservable-bandwidth '100000'
```

```
set protocols traffic-engineering interface eth3 max-bandwidth '10000'  
set protocols traffic-engineering interface eth3 max-reservable-bandwidth '10000'  
set protocols isis interface eth1 network point-to-point  
set protocols isis interface eth2 network point-to-point  
set protocols isis interface eth3 network point-to-point  
set protocols isis interface lo passive  
set protocols isis level 'level-2'  
set protocols isis lsp-mtu '1300'  
set protocols isis net '49.0004.0004.0004.00'  
set protocols isis segment-routing global-block high-label-value '865534'  
set protocols isis segment-routing global-block low-label-value '800000'  
set protocols isis segment-routing prefix 4.4.4.4/32 index value '4'  
set protocols isis traffic-engineering enable  
set protocols isis traffic-engineering export  
set protocols mpls interface 'eth2'  
set protocols mpls interface 'eth3'  
set protocols mpls interface 'eth1'  
set protocols mpls interface 'lo'
```

P3:

```
set system host-name 'P3'  
set interfaces ethernet eth1 address '10.100.5.5/24'  
set interfaces ethernet eth2 address '10.100.4.5/24'  
set interfaces loopback lo address '5.5.5.5/32'  
set protocols traffic-engineering admin-group YELLOW bit-position '1'  
set protocols traffic-engineering interface eth1 admin-group 'YELLOW'  
set protocols traffic-engineering interface eth1 max-bandwidth '100000'  
set protocols traffic-engineering interface eth1 max-reservable-bandwidth '100000'  
set protocols traffic-engineering interface eth2 admin-group 'YELLOW'  
set protocols traffic-engineering interface eth2 max-bandwidth '100000'  
set protocols traffic-engineering interface eth2 max-reservable-bandwidth '100000'  
set protocols isis interface eth1 network point-to-point  
set protocols isis interface eth2 network point-to-point  
set protocols isis interface lo passive  
set protocols isis level 'level-2'  
set protocols isis lsp-mtu '1300'  
set protocols isis net '49.0005.0005.0005.00'  
set protocols isis segment-routing global-block high-label-value '865534'  
set protocols isis segment-routing global-block low-label-value '800000'  
set protocols isis segment-routing prefix 5.5.5.5/32 index value '5'  
set protocols isis traffic-engineering enable  
set protocols isis traffic-engineering export  
set protocols mpls interface 'eth2'  
set protocols mpls interface 'eth1'  
set protocols mpls interface 'lo'
```

P1(BGP-LS):

```
set system host-name P1  
  
set interfaces ethernet eth1 address '10.100.1.3/24'  
set interfaces ethernet eth2 address '10.100.3.3/24'  
set interfaces ethernet eth3 address '10.100.10.3/24'  
set interfaces ethernet eth5 address '192.168.0.103/24'  
set interfaces loopback lo address '3.3.3.3/32'  
set protocols bgp neighbor 192.168.0.1 address-family link-state  
set protocols bgp neighbor 192.168.0.1 remote-as '65001'  
set protocols bgp parameters router-id '3.3.3.3'
```



```

set protocols bgp system-as '65002'
set protocols isis interface eth1 network point-to-point
set protocols isis interface eth2 network point-to-point
set protocols isis interface eth3 network point-to-point
set protocols isis interface eth5 network point-to-point
set protocols isis interface lo passive
set protocols isis level 'level-2'
set protocols isis lsp-mtu '1300'
set protocols isis net '49.0003.0003.0003.00'
set protocols isis segment-routing global-block high-label-value '865534'
set protocols isis segment-routing global-block low-label-value '800000'
set protocols isis segment-routing maximum-label-depth '13'
set protocols isis segment-routing prefix 3.3.3.3/32 index value '3'
set protocols isis traffic-engineering enable
set protocols isis traffic-engineering export
set protocols mpls interface 'eth1'
set protocols mpls interface 'eth2'
set protocols mpls interface 'eth3'
set protocols mpls interface 'eth5'
set protocols mpls interface 'lo'
set protocols traffic-engineering admin-group BLUE bit-position '0'
set protocols traffic-engineering interface eth1 admin-group 'BLUE'
set protocols traffic-engineering interface eth1 max-bandwidth '10000'
set protocols traffic-engineering interface eth1 max-reservable-bandwidth '10000'
set protocols traffic-engineering interface eth2 admin-group 'BLUE'
set protocols traffic-engineering interface eth2 max-bandwidth '10000'
set protocols traffic-engineering interface eth2 max-reservable-bandwidth '10000'
set protocols traffic-engineering interface eth3 max-bandwidth '10000'
set protocols traffic-engineering interface eth3 max-reservable-bandwidth '10000'
set system host-name 'P1'

```

We are going to start with our PE1/2 configuration

VyOS-PE1

```

### PE1

### interfaces

set interfaces dummy dum1 address '10.200.1.1/32'
set interfaces dummy dum2 address '10.200.1.2/32'
set interfaces ethernet eth1 address '10.100.1.1/24'
set interfaces ethernet eth2 address '10.100.2.1/24'
set interfaces ethernet eth3 address '10.100.6.1/24'
set interfaces ethernet eth3 vrf 'VPN101'
set interfaces ethernet eth4 address '10.100.7.1/24'
set interfaces ethernet eth4 vrf 'VPN102'
set interfaces ethernet eth5 address '192.168.0.101/24'
set interfaces loopback lo address '1.1.1.1/32'

## BGP policy :

set policy prefix-list VPN101 rule 10 action 'permit'
set policy prefix-list VPN101 rule 10 prefix '192.168.1.1/32'
set policy prefix-list VPN102 rule 10 action 'permit'
set policy prefix-list VPN102 rule 10 prefix '192.168.2.2/32'
set policy route-map DENY_ALL rule 10 action 'deny'
set policy route-map LP500 rule 10 action 'permit'
set policy route-map LP500 rule 10 set local-preference '500'

```



```
set policy route-map PASS rule 10 action 'permit'
set policy route-map VPN_EXPORT rule 10 action 'permit'
set policy route-map VPN_EXPORT rule 10 match ip address prefix-list 'VPN101'
set policy route-map VPN_EXPORT rule 10 set ip-next-hop '10.200.1.1'
set policy route-map VPN_EXPORT rule 20 action 'permit'
set policy route-map VPN_EXPORT rule 20 match ip address prefix-list 'VPN102'
set policy route-map VPN_EXPORT rule 20 set ip-next-hop '10.200.1.2'
set policy route-map VPN_EXPORT rule 30 action 'permit'

set policy route-map VPN_EXPORT rule 30 set

### BGP-LU configuration:

set protocols bgp address-family ipv4-labeled-unicast network 10.200.1.1/32
set protocols bgp address-family ipv4-labeled-unicast network 10.200.1.2/32
set protocols bgp address-family ipv4-unicast network 10.200.1.1/32
set protocols bgp address-family ipv4-unicast network 10.200.1.2/32
set protocols bgp address-family ipv4-vpn
set protocols bgp interface eth5 mpls forwarding
set protocols bgp neighbor 2.2.2.2 address-family ipv4-labeled-unicast
set protocols bgp neighbor 2.2.2.2 address-family ipv4-vpn route-map export 'VPN_EXPORT'
set protocols bgp neighbor 2.2.2.2 remote-as '65002'
set protocols bgp neighbor 2.2.2.2 update-source 'lo'
set protocols bgp neighbor 192.168.0.1 address-family ipv4-labeled-unicast route-map export 'DENY_ALL'
set protocols bgp neighbor 192.168.0.1 address-family ipv4-labeled-unicast route-map import 'LP500'
set protocols bgp neighbor 192.168.0.1 remote-as '65001'
set protocols bgp parameters router-id '1.1.1.1'
set protocols bgp system-as '65002'

## ISIS and SR-TE

set protocols traffic-engineering admin-group BLUE bit-position '0'
set protocols traffic-engineering admin-group YELLOW bit-position '1'
set protocols traffic-engineering interface eth1 admin-group 'BLUE'
set protocols traffic-engineering interface eth1 max-bandwidth '10000'
set protocols traffic-engineering interface eth1 max-reservable-bandwidth '10000'
set protocols traffic-engineering interface eth2 admin-group 'YELLOW'
set protocols traffic-engineering interface eth2 max-bandwidth '100000'
set protocols traffic-engineering interface eth2 max-reservable-bandwidth '100000'
set protocols isis interface eth1 network point-to-point
set protocols isis interface eth2 network point-to-point
set protocols isis interface lo passive
set protocols isis level 'level-2'
set protocols isis lsp-mtu '1300'
set protocols isis net '49.0001.0001.0001.00'
set protocols isis segment-routing global-block high-label-value '865534'
set protocols isis segment-routing global-block low-label-value '800000'
set protocols isis segment-routing prefix 1.1.1.1/32 index value '1'
set protocols isis traffic-engineering enable
set protocols isis traffic-engineering export
set protocols mpls interface 'eth1'
set protocols mpls interface 'eth2'
set protocols mpls interface 'eth5'
set protocols mpls interface 'lo'
set protocols mpls interface 'dum1'
set protocols mpls interface 'dum2'
set system host-name 'PE1'
```



VyOS-PE2

```

### PE2

### interfaces

set interfaces dummy dum1 address '10.200.2.1/32'
set interfaces dummy dum2 address '10.200.2.2/32'
set interfaces ethernet eth1 address '10.100.3.2/24'
set interfaces ethernet eth2 address '10.100.5.2/24'
set interfaces ethernet eth3 address '10.100.8.2/24'
set interfaces ethernet eth3 vrf 'VPN101'
set interfaces ethernet eth4 address '10.100.9.2/24'
set interfaces ethernet eth4 vrf 'VPN102'
set interfaces ethernet eth5 address '192.168.0.102/24'
set interfaces loopback lo address '2.2.2.2/32'

# BGP policy;

set policy prefix-list VPN101 rule 10 action 'permit'
set policy prefix-list VPN101 rule 10 prefix '192.168.3.3/32'
set policy prefix-list VPN102 rule 10 action 'permit'
set policy prefix-list VPN102 rule 10 prefix '192.168.4.4/32'
set policy route-map DENY_ALL rule 10 action 'deny'
set policy route-map LP500 rule 10 action 'permit'
set policy route-map LP500 rule 10 set local-preference '500'
set policy route-map PASS rule 10 action 'permit'
set policy route-map VPN_EXPORT rule 10 action 'permit'
set policy route-map VPN_EXPORT rule 10 match ip address prefix-list 'VPN101'
set policy route-map VPN_EXPORT rule 10 set ip-next-hop '10.200.2.1'
set policy route-map VPN_EXPORT rule 20 action 'permit'
set policy route-map VPN_EXPORT rule 20 match ip address prefix-list 'VPN102'
set policy route-map VPN_EXPORT rule 20 set ip-next-hop '10.200.2.2'
set policy route-map VPN_EXPORT rule 30 action 'permit'
set policy route-map VPN_EXPORT rule 30 set

### BGP-LU configuration:

set protocols bgp address-family ipv4-labeled-unicast network 10.200.2.1/32
set protocols bgp address-family ipv4-labeled-unicast network 10.200.2.2/32
set protocols bgp address-family ipv4-unicast network 10.200.2.1/32
set protocols bgp address-family ipv4-unicast network 10.200.2.2/32
set protocols bgp address-family ipv4-vpn
set protocols bgp interface eth5 mpls forwarding
set protocols bgp neighbor 1.1.1.1 address-family ipv4-labeled-unicast
set protocols bgp neighbor 1.1.1.1 address-family ipv4-vpn route-map export 'VPN_EXPORT'
set protocols bgp neighbor 1.1.1.1 remote-as '65002'
set protocols bgp neighbor 1.1.1.1 update-source 'lo'
set protocols bgp neighbor 192.168.0.1 address-family ipv4-labeled-unicast route-map export 'DENY_ALL'
set protocols bgp neighbor 192.168.0.1 address-family ipv4-labeled-unicast route-map import 'LP500'
set protocols bgp neighbor 192.168.0.1 remote-as '65001'
set protocols bgp parameters router-id '2.2.2.2'
set protocols bgp system-as '65002'

## ISIS and SR-TE

set protocols traffic-engineering admin-group BLUE bit-position '0'
set protocols traffic-engineering admin-group YELLOW bit-position '1'
set protocols traffic-engineering interface eth1 admin-group 'BLUE'
set protocols traffic-engineering interface eth1 max-bandwidth '10000'
set protocols traffic-engineering interface eth1 max-reservable-bandwidth '10000'
set protocols traffic-engineering interface eth2 admin-group 'YELLOW'

```



```
set protocols traffic-engineering interface eth2 max-bandwidth '100000'
set protocols traffic-engineering interface eth2 max-reservable-bandwidth '100000'
set protocols isis interface eth1 network point-to-point
set protocols isis interface eth2 network point-to-point
set protocols isis interface lo passive
set protocols isis level 'level-2'
set protocols isis lsp-mtu '1300'
set protocols isis net '49.0002.0002.0002.00'
set protocols isis segment-routing global-block high-label-value '865534'
set protocols isis segment-routing global-block low-label-value '800000'
set protocols isis segment-routing prefix 2.2.2.2/32 index value '2'
set protocols isis traffic-engineering enable
set protocols isis traffic-engineering export
set protocols mpls interface 'eth1'
set protocols mpls interface 'eth2'
set protocols mpls interface 'eth5'
set protocols mpls interface 'lo'
set protocols mpls interface 'dum1'
set protocols mpls interface 'dum2'
```

VRF+L3VPN in PE1/PE2

#VyOS-PE1:

```
set vrf name VPN101 protocols bgp address-family ipv4-unicast export vpn
set vrf name VPN101 protocols bgp address-family ipv4-unicast import vpn
set vrf name VPN101 protocols bgp address-family ipv4-unicast label vpn export 'auto'
set vrf name VPN101 protocols bgp address-family ipv4-unicast network 10.100.6.0/24
set vrf name VPN101 protocols bgp address-family ipv4-unicast rd vpn export '65002:101'
set vrf name VPN101 protocols bgp address-family ipv4-unicast route-target vpn both '65002:101'
set vrf name VPN101 protocols bgp neighbor 10.100.6.6 address-family ipv4-unicast
set vrf name VPN101 protocols bgp neighbor 10.100.6.6 remote-as '65101'
set vrf name VPN101 protocols bgp parameters router-id '1.1.1.1'
set vrf name VPN101 protocols bgp system-as '65002'
set vrf name VPN101 table '101'
set vrf name VPN102 protocols bgp address-family ipv4-unicast export vpn
set vrf name VPN102 protocols bgp address-family ipv4-unicast import vpn
set vrf name VPN102 protocols bgp address-family ipv4-unicast label vpn export 'auto'
set vrf name VPN102 protocols bgp address-family ipv4-unicast network 10.100.7.0/24
set vrf name VPN102 protocols bgp address-family ipv4-unicast rd vpn export '65002:102'
set vrf name VPN102 protocols bgp address-family ipv4-unicast route-target vpn both '65002:102'
set vrf name VPN102 protocols bgp neighbor 10.100.7.7 address-family ipv4-unicast
set vrf name VPN102 protocols bgp neighbor 10.100.7.7 remote-as '65102'
set vrf name VPN102 protocols bgp parameters router-id '1.1.1.1'
set vrf name VPN102 protocols bgp system-as '65002'
set vrf name VPN102 table '102'
```

#VyOS-PE2:

```
set vrf name VPN101 protocols bgp address-family ipv4-unicast export vpn
set vrf name VPN101 protocols bgp address-family ipv4-unicast import vpn
set vrf name VPN101 protocols bgp address-family ipv4-unicast label vpn export 'auto'
set vrf name VPN101 protocols bgp address-family ipv4-unicast network 10.100.8.0/24
set vrf name VPN101 protocols bgp address-family ipv4-unicast rd vpn export '65002:101'
set vrf name VPN101 protocols bgp address-family ipv4-unicast route-target vpn both '65002:101'
set vrf name VPN101 protocols bgp neighbor 10.100.8.8 address-family ipv4-unicast
set vrf name VPN101 protocols bgp neighbor 10.100.8.8 remote-as '65111'
set vrf name VPN101 protocols bgp parameters router-id '2.2.2.2'
set vrf name VPN101 protocols bgp system-as '65002'
```



```
set vrf name VPN101 table '101'
set vrf name VPN102 protocols bgp address-family ipv4-unicast export vpn
set vrf name VPN102 protocols bgp address-family ipv4-unicast import vpn
set vrf name VPN102 protocols bgp address-family ipv4-unicast label vpn export 'auto'
set vrf name VPN102 protocols bgp address-family ipv4-unicast network 10.100.9.0/24
set vrf name VPN102 protocols bgp address-family ipv4-unicast rd vpn export '65002:102'
set vrf name VPN102 protocols bgp address-family ipv4-unicast route-target vpn both '65002:102'
set vrf name VPN102 protocols bgp neighbor 10.100.9.9 address-family ipv4-unicast
set vrf name VPN102 protocols bgp neighbor 10.100.9.9 remote-as '65112'
set vrf name VPN102 protocols bgp parameters router-id '2.2.2.2'
set vrf name VPN102 protocols bgp system-as '65002'
set vrf name VPN102 table '102'
```

CE1/CE2/CE3/CE4 Configurations

```
#CE1

#interfaces
set interfaces ethernet eth1 address '10.100.6.6/24'
set interfaces loopback lo address '6.6.6.6/32'
set interfaces dummy dum1 address '192.168.1.1/32'

#bgp
set policy route-map PASS rule 10 action 'permit'
set protocols bgp address-family ipv4-unicast network 192.168.1.1/32
set protocols bgp neighbor 10.100.6.1 address-family ipv4-unicast route-map import 'PASS'
set protocols bgp neighbor 10.100.6.1 address-family ipv4-unicast route-map export 'PASS'
set protocols bgp neighbor 10.100.6.1 remote-as '65002'
set protocols bgp parameters router-id '6.6.6.6'
set protocols bgp system-as '65101'

set system host-name CE1

#CE2

#interfaces
set interfaces ethernet eth1 address '10.100.7.7/24'
set interfaces loopback lo address '7.7.7.7/32'
set interfaces dummy dum1 address '192.168.2.2/32'

#bgp
set policy route-map PASS rule 10 action 'permit'
set protocols bgp address-family ipv4-unicast network 192.168.2.2/32
set protocols bgp neighbor 10.100.7.1 address-family ipv4-unicast route-map import 'PASS'
set protocols bgp neighbor 10.100.7.1 address-family ipv4-unicast route-map export 'PASS'
set protocols bgp neighbor 10.100.7.1 remote-as '65002'
set protocols bgp parameters router-id '7.7.7.7'
set protocols bgp system-as '65102'

set system host-name CE2

#CE3

#interfaces
set interfaces ethernet eth1 address '10.100.8.8/24'
set interfaces loopback lo address '8.8.8.8/32'
set interfaces dummy dum1 address '192.168.3.3/32'

#bgp
set policy route-map PASS rule 10 action 'permit'
set protocols bgp address-family ipv4-unicast network 192.168.3.3/32
set protocols bgp neighbor 10.100.8.2 address-family ipv4-unicast route-map import 'PASS'
```



```

set protocols bgp neighbor 10.100.8.2 address-family ipv4-unicast route-map export 'PASS'
set protocols bgp neighbor 10.100.8.2 remote-as '65002'
set protocols bgp parameters router-id '8.8.8.8'
set protocols bgp system-as '65111'

set system host-name CE3

#CE4

#interfaces
set interfaces ethernet eth1 address '10.100.9.9/24'
set interfaces loopback lo address '9.9.9.9/32'
set interfaces dummy dum1 address '192.168.4.4/32'

#bgp
set policy route-map PASS rule 10 action 'permit'
set protocols bgp address-family ipv4-unicast network 192.168.4.4/32
set protocols bgp neighbor 10.100.2 address-family ipv4-unicast route-map import 'PASS'
set protocols bgp neighbor 10.100.9.2 address-family ipv4-unicast route-map export 'PASS'
set protocols bgp neighbor 10.100.9.2 remote-as '65002'
set protocols bgp parameters router-id '9.9.9.9'
set protocols bgp system-as '65112'

set system host-name CE4

```

Traffic Dictator Policies/Mapping traffic

In this section, after completing the basic environment setup, we will configure the traffic dictator parameters to define traffic engineering policies and establish color-coded paths for each service loopback.

```

### Basic BGP-LU/LS configuration:

router general
  spf delay 200
  spf holddown 5000
  spf rapid-runs 3
  srte distinguisher 1
  reoptimization interval 3600
!
router bgp 65001
  router-id 192.168.0.1
  !
  neighbor 192.168.0.101
    remote-as 65002
    address-family ipv4-labeled-unicast
  !
  neighbor 192.168.0.102
    remote-as 65002
    address-family ipv4-labeled-unicast
  !
  neighbor 192.168.0.103
    remote-as 65002
    address-family link-state
!

## defined affinities groups.
traffic-eng affinities
  affinity-map
    name BLUE bit-position 0

```



```

    name YELLOW bit-position 1
!
srlg-map
!
affinity-set BLUE_ONLY
    include all
    !
    include-affinities
        name BLUE
    !
    exclude-affinities
    !
    include-srlg
    !
    exclude-srlg
!
affinity-set YELLOW_ONLY
    include all
    !
    include-affinities
        name YELLOW
    !
    exclude-affinities
    !
    include-srlg
    !
    exclude-srlg
!
# define traffic engineering policies:

traffic-eng policies
!
policy PE1_PE2_BLUE
    headend 1.1.1.1 topology-id 0
    endpoint 2.2.2.2 service-loopback 10.200.2.1
    install direct labeled-unicast 192.168.0.101
    !
    candidate-path preference 100
        affinity-set BLUE_ONLY
!
policy PE1_PE2_YELLOW
    headend 1.1.1.1 topology-id 0
    endpoint 2.2.2.2 service-loopback 10.200.2.2
    install direct labeled-unicast 192.168.0.101
    !
    candidate-path preference 100
        affinity-set YELLOW_ONLY
!
policy PE2_PE1_BLUE
    headend 2.2.2.2 topology-id 0
    endpoint 1.1.1.1 service-loopback 10.200.1.1
    install direct labeled-unicast 192.168.0.102
    !
    candidate-path preference 100
        affinity-set BLUE_ONLY
!
policy PE2_PE1_YELLOW
    headend 2.2.2.2 topology-id 0
    endpoint 1.1.1.1 service-loopback 10.200.1.2
    install direct labeled-unicast 192.168.0.102

```



```

!
candidate-path preference 100
  affinity-set YELLOW_ONLY
!
traffic-eng mesh-templates
!
traffic-eng peer-groups
!
traffic-eng explicit-paths
!

```

Validation and troubleshooting

This section validates our approach by illustrating how the network operator can design and adjust network paths using service loopbacks.

Traffic Dictactor

```
TD#show bgp summary
```

```
BGP summary information
```

```
Router identifier 192.168.0.1, local AS number 65001
```

Neighbor	V	AS	MsgRcvd	MsgSent	InQ	OutQ	Up/Down	State	Received NLRI	Active AF
192.168.0.101	4	65002	63	68	0	0	1:00:17	Established	0	IPv4-LU
192.168.0.102	4	65002	63	62	0	0	1:00:17	Established	0	IPv4-LU
192.168.0.103	4	65002	81	62	0	0	1:00:17	Established	35	LS

```
TD#show bgp link-state
```

```
BGP-LS routing table information
```

```
Router identifier 192.168.0.1, local AS number 65001
```

```
Status codes: * valid, > best
```

```
Origin codes: i - IGP, e - EGP, ? - incomplete
```

```
Prefix codes: E link, V node, T IP reachable route, S SRv6 SID, SP SRTE Policy, u/U unknown,
```

```
I Identifier, N local node, R remote node, L link, P prefix, S SID,
```

```
L1/L2 ISIS level-1/level-2, O OSPF, D direct, S static/peer-node,
```

```
a area-ID, l link-ID, t topology-ID, s ISO-ID,
```

```
c confed-ID/ASN, b bgp-identifier, r router-ID, s SID,
```

```
i if-address, n nbr-address, o OSPF Route-type, p IP-prefix,
```

```
d designated router address
```

	Network	Next Hop	Metric	LocPref	Weigth	Path
	[V][L2][I0][N[c0][b0][s0003.0003.0003.00]]					
*>	192.168.0.103	0	100	0	65002	?
	[T][L2][I0][N[c0][b0][s0003.0003.0003.00]][P[p192.168.0.0/24]]					
*>	192.168.0.103	0	100	0	65002	?
	[T][L2][I0][N[c0][b0][s0002.0002.0002.00]][P[p10.100.5.0/24]]					
*>	192.168.0.103	0	100	0	65002	?
	[T][L2][I0][N[c0][b0][s0002.0002.0002.00]][P[p10.100.3.0/24]]					
*>	192.168.0.103	0	100	0	65002	?
	[T][L2][I0][N[c0][b0][s0001.0001.0001.00]][P[p10.100.1.0/24]]					
*>	192.168.0.103	0	100	0	65002	?
	[T][L2][I0][N[c0][b0][s0005.0005.0005.00]][P[p10.100.5.0/24]]					
*>	192.168.0.103	0	100	0	65002	?
	[T][L2][I0][N[c0][b0][s0004.0004.0004.00]][P[p10.100.4.0/24]]					
*>	192.168.0.103	0	100	0	65002	?
	[T][L2][I0][N[c0][b0][s0004.0004.0004.00]][P[p10.100.2.0/24]]					



```

*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0004.0004.0004.00]][P[p10.100.10.0/24]]
*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0005.0005.0005.00]][P[p10.100.4.0/24]]
*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0003.0003.0003.00]][P[p10.100.1.0/24]]
*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0003.0003.0003.00]][P[p10.100.3.0/24]]
*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0003.0003.0003.00]][P[p10.100.10.0/24]]
*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0001.0001.0001.00]][P[p10.100.2.0/24]]
*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0003.0003.0003.00]][P[p3.3.3.3/32]]
*>          192.168.0.103          0          100          0 65002 ?
[V][L2][I0][N[c0][b0][s0002.0002.0002.00]]
*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0002.0002.0002.00]][P[p2.2.2.2/32]]
*>          192.168.0.103          0          100          0 65002 ?
[V][L2][I0][N[c0][b0][s0001.0001.0001.00]]
*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0001.0001.0001.00]][P[p1.1.1.1/32]]
*>          192.168.0.103          0          100          0 65002 ?
[V][L2][I0][N[c0][b0][s0005.0005.0005.00]]
*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0005.0005.0005.00]][P[p5.5.5.5/32]]
*>          192.168.0.103          0          100          0 65002 ?
[V][L2][I0][N[c0][b0][s0004.0004.0004.00]]
*>          192.168.0.103          0          100          0 65002 ?
[T][L2][I0][N[c0][b0][s0004.0004.0004.00]][P[p4.4.4.4/32]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0005.0005.0005.00]][R[c0][b0][s0002.0002.0002.00]][L[i10.100.5.5][n10.100.5.2]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0004.0004.0004.00]][R[c0][b0][s0001.0001.0001.00]][L[i10.100.2.4][n10.100.2.1]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0002.0002.0002.00]][R[c0][b0][s0005.0005.0005.00]][L[i10.100.5.2][n10.100.5.5]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0005.0005.0005.00]][R[c0][b0][s0004.0004.0004.00]][L[i10.100.4.5][n10.100.4.4]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0004.0004.0004.00]][R[c0][b0][s0005.0005.0005.00]][L[i10.100.4.4][n10.100.4.5]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0001.0001.0001.00]][R[c0][b0][s0004.0004.0004.00]][L[i10.100.2.1][n10.100.2.4]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0003.0003.0003.00]][R[c0][b0][s0001.0001.0001.00]][L[i10.100.1.3][n10.100.1.1]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0001.0001.0001.00]][R[c0][b0][s0003.0003.0003.00]][L[i10.100.1.1][n10.100.1.3]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0003.0003.0003.00]][R[c0][b0][s0002.0002.0002.00]][L[i10.100.3.3][n10.100.3.2]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0002.0002.0002.00]][R[c0][b0][s0003.0003.0003.00]][L[i10.100.3.2][n10.100.3.3]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0003.0003.0003.00]][R[c0][b0][s0004.0004.0004.00]][L[i10.100.10.3][n10.100.10.4]]
*>          192.168.0.103          0          100          0 65002 ?
[E][L2][I0][N[c0][b0][s0004.0004.0004.00]][R[c0][b0][s0003.0003.0003.00]][L[i10.100.10.4][n10.100.10.3]]
*>          192.168.0.103          0          100          0 65002 ?

```

This section validates our approach by illustrating how the network operator can design and adjust network paths using service loopbacks.



#Basic check:

TD#show traffic-eng policy PE1_PE2_BLUE

Traffic-eng policy information

Status codes: * active, > installed, r - RSVP-TE, e - EPE only, s - admin down, m - multi-topology

Endpoint codes: * active override

Policy name	Headend	Endpoint	Color/Service loopback	Protocol
Reserved bandwidth	Priority	Status/Reason		
*> PE1_PE2_BLUE	1.1.1.1	2.2.2.2	10.200.2.1	LU/direct
N/A	7/7	Active/Installed		

TD#

TD#show traffic-eng policy PE1_PE2_YELLOW

Traffic-eng policy information

Status codes: * active, > installed, r - RSVP-TE, e - EPE only, s - admin down, m - multi-topology

Endpoint codes: * active override

Policy name	Headend	Endpoint	Color/Service loopback	Protocol
Reserved bandwidth	Priority	Status/Reason		
*> PE1_PE2_YELLOW	1.1.1.1	2.2.2.2	10.200.2.2	LU/direct
N/A	7/7	Active/Installed		

TD#

In this output, we extend the policy to demonstrate how the **path option**, **affinity set**, and **label stack** are applied in **BGP-LU**. This shows the active policy and includes all the relevant details required for verification.

#BLUE Policy

TD#show traffic-eng policy PE1_PE2_BLUE detail

Detailed traffic-eng policy information:

Traffic engineering policy "PE1_PE2_BLUE"

Valid config, Active, Installed

Headend 1.1.1.1, topology-id 0

Endpoint 2.2.2.2, service-loopback 10.200.2.1

Endpoint type: Node, Topology-id: 101, Protocol: isis, Router-id: 0002.0002.0002.00

Setup priority: 7, Hold priority: 7

Reserved bandwidth bps: 0

Install direct, protocol labeled-unicast, peer 192.168.0.101

Policy index: 0, SR-TE distinguisher: 16777216

Route-key: [16777216][10.200.2.1/32]

Candidate paths:

Candidate-path preference 100

Path config valid

Metric: igp

Path-option: dynamic

Affinity-set: BLUE_ONLY

Constraint: include-all

List: ['BLUE']

Value: 0x1

This path is currently active

Calculation results:

Aggregate metric: 20

Topologies: ['101']

Segment lists:

[800002]

BGP-LU next-hop: 10.100.1.3



Policy statistics:

Last config update: 2025-12-17 21:18:25,471

Last recalculation: 2025-12-30 13:26:16.752

Policy calculation took 0 miliseconds

YELLOW Policy:

TD#show traffic-eng policy PE1_PE2_YELLOW detail

Detailed traffic-eng policy information:

Traffic engineering policy "PE1_PE2_YELLOW"

Valid config, Active, Installed

Headend 1.1.1.1, topology-id 0

Endpoint 2.2.2.2, service-loopback 10.200.2.2

Endpoint type: Node, Topology-id: 101, Protocol: isis, Router-id: 0002.0002.0002.00

Setup priority: 7, Hold priority: 7

Reserved bandwidth bps: 0

Install direct, protocol labeled-unicast, peer 192.168.0.101

Policy index: 1, SR-TE distinguisher: 16777217

Route-key: [16777217][10.200.2.2/32]

Candidate paths:

Candidate-path preference 100

Path config valid

Metric: igp

Path-option: dynamic

Affinity-set: YELLOW_ONLY

Constraint: include-all

List: ['YELLOW']

Value: 0x2

This path is currently active

Calculation results:

Aggregate metric: 30

Topologies: ['101']

Segment lists:

[800005, 800002]

BGP-LU next-hop: 10.100.2.4

Policy statistics:

Last config update: 2025-12-17 21:18:25,570

Last recalculation: 2025-12-30 13:26:16.752

Policy calculation took 0 miliseconds

Now, let's check in our PE1 if our VPN101/VPN102 are taking the path that we've assigned by our controller.

#VPN101:

vyos@PE1:~\$ show ip route vrf VPN101

Codes: K - kernel route, C - connected, L - local, S - static,

R - RIP, O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,

T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,

f - OpenFabric, t - Table-Direct,

> - selected route, * - FIB route, q - queued, r - rejected, b - backup

t - trapped, o - offload failure

VRF VPN101:

C>* 10.100.6.0/24 is directly connected, eth3, weight 1, 01w5d16h



```

K * 10.100.6.0/24 [0/0] is directly connected, eth3, weight 1, 01w5d16h
L * 10.100.6.1/32 is directly connected, eth3, weight 1, 01w5d16h
L>* 10.100.6.1/32 is directly connected, eth3, weight 1, 01w5d16h
B> 10.100.8.0/24 [200/0] via 2.2.2.2 (vrf default) (recursive), label 81, weight 1, 6d19h05m
*
   via 10.100.1.3, eth1 (vrf default), label 800002/81, weight 1, 6d19h05m
B>* 192.168.1.1/32 [20/0] via 10.100.6.6, eth3, weight 1, 6d18h19m
B> 192.168.3.3/32 [200/0] via 10.200.2.1 (vrf default) (recursive), label 81, weight 1, 6d18h13m
*
   via 10.100.1.3, eth1 (vrf default), label 800002/81, weight 1, 6d18h13m

vyos@PE1:~$ show ip route 10.200.2.1
Routing entry for 10.200.2.1/32
  Known via "bgp", distance 20, metric 0, best
  Last update 6d19h06m ago
  * 10.100.1.3, via eth1, label 800002, weight 1

#VPN102

vyos@PE1:~$ show ip route vrf VPN102
Codes: K - kernel route, C - connected, L - local, S - static,
       R - RIP, O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric, t - Table-Direct,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF VPN102:
C>* 10.100.7.0/24 is directly connected, eth4, weight 1, 01w5d16h
K * 10.100.7.0/24 [0/0] is directly connected, eth4, weight 1, 01w5d16h
L * 10.100.7.1/32 is directly connected, eth4, weight 1, 01w5d16h
L>* 10.100.7.1/32 is directly connected, eth4, weight 1, 01w5d16h
B> 10.100.9.0/24 [200/0] via 2.2.2.2 (vrf default) (recursive), label 80, weight 1, 6d19h09m
*
   via 10.100.1.3, eth1 (vrf default), label 800002/80, weight 1, 6d19h09m
B>* 192.168.2.2/32 [20/0] via 10.100.7.7, eth4, weight 1, 6d18h12m
B> 192.168.4.4/32 [200/0] via 10.200.2.2 (vrf default) (recursive), label 80, weight 1, 6d18h07m
*
   via 10.100.2.4, eth2 (vrf default), label 800005/800002/80, weight 1, 6d18h07m

vyos@PE1:~$ show ip route 10.200.2.2
Routing entry for 10.200.2.2/32
  Known via "bgp", distance 20, metric 0, best
  Last update 6d19h13m ago
  * 10.100.2.4, via eth2, label 800005/800002, weight 1

```

As we see in our output, VRFs are taking different paths as we defined in our SDN controller.

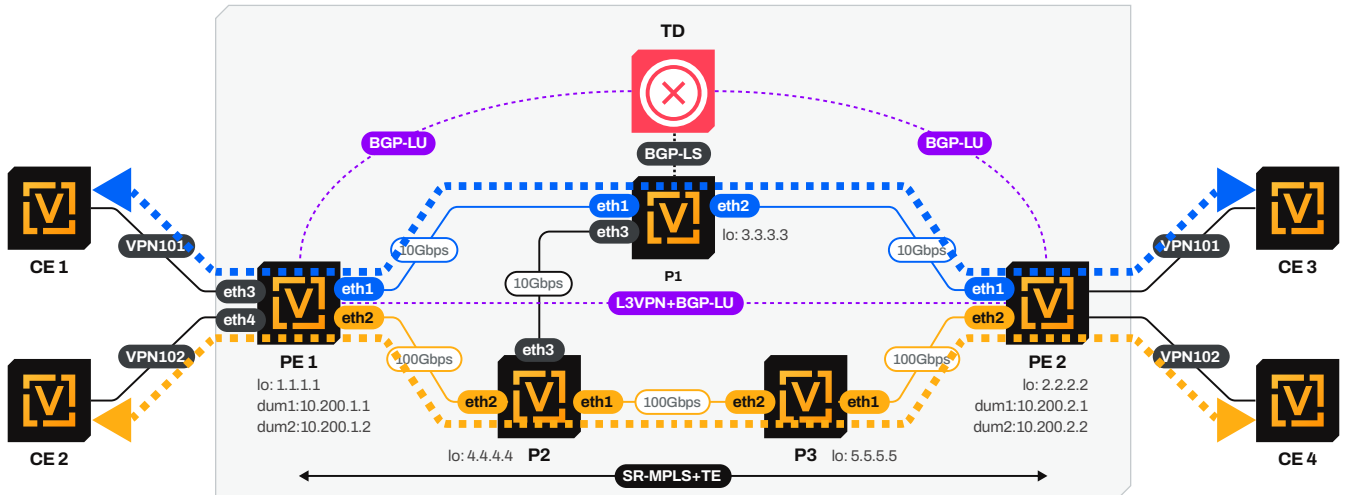
What happens if the Traffic Dictator goes down?

Traffic Dictator supports controller redundancy, so if the primary controller fails, the backup controller will take over and all traffic will keep using the same traffic engineering policies.

Now let's take a look at a critical failure scenario, when all controllers fail at the same time.

In regular SR-TE, color will be ignored and BGP routes will just resolve via IGP instead of the SR-TE policy, and follow the shortest path.





This scenario occurs when connectivity to the controller is lost. If the controller fails, BGP routes are resolved using the BGP-LU route, which in turn is resolved via the IGP. As a result, traffic follows the shortest path determined by the IGP

Validation:

```
vyos@PE1:~$ show bgp summary
```

IPv4 VPN Summary:

BGP router identifier 1.1.1.1, local AS number 65002 VRF default vrf-id 0

BGP table version 0

RIB entries 3, using 384 bytes of memory

Peers 1, using 24 KiB of memory

Neighbor	V	AS	MsgRcvd	MsgSent	TblVer	InQ	OutQ	Up/Down	State/PfxRcd	PfxSnt Desc
2.2.2.2	4	65002	10050	10050	4	0	0	6d23h21m	4	4 FRRouting/10.2.4

Total number of neighbors 1

IPv4 Labeled Unicast Summary:

BGP router identifier 1.1.1.1, local AS number 65002 VRF default vrf-id 0

BGP table version 2

RIB entries 3, using 384 bytes of memory

Peers 2, using 47 KiB of memory

Neighbor	V	AS	MsgRcvd	MsgSent	TblVer	InQ	OutQ	Up/Down	State/PfxRcd	PfxSnt Desc
2.2.2.2	4	65002	10050	10050	8	0	0	6d23h21m	2	2 FRRouting/10.2.4
192.168.0.1	4	65001	10387	10052	0	0	0	00:03:32	Active	0 N/A

Total number of neighbors 2

```
vyos@PE1:~$
```

Our BGP-LU sessions established with our controller are down.

```
vyos@PE1:~$ show ip route vrf VPN101
```

Codes: K - kernel route, C - connected, L - local, S - static,
 R - RIP, O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
 T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
 f - OpenFabric, t - Table-Direct,



```
> - selected route, * - FIB route, q - queued, r - rejected, b - backup
t - trapped, o - offload failure
```

```
VRF VPN101:
```

```
C>* 10.100.6.0/24 is directly connected, eth3, weight 1, 01w5d21h
K * 10.100.6.0/24 [0/0] is directly connected, eth3, weight 1, 01w5d21h
L * 10.100.6.1/32 is directly connected, eth3, weight 1, 01w5d21h
L>* 10.100.6.1/32 is directly connected, eth3, weight 1, 01w5d21h
B> 10.100.8.0/24 [200/0] via 2.2.2.2 (vrf default) (recursive), label 81, weight 1, 6d23h19m
*
   via 10.100.1.3, eth1 (vrf default), label 800002/81, weight 1, 6d23h19m
B>* 192.168.1.1/32 [20/0] via 10.100.6.6, eth3, weight 1, 6d22h32m
B> 192.168.3.3/32 [200/0] via 10.200.2.1 (vrf default) (recursive), label 81, weight 1, 00:01:49
*
   via 10.100.1.3, eth1 (vrf default), label 800002/implicit-null/81, weight 1, 00:01:49
```

```
vyos@PE1:~$ show ip route vrf VPN102
```

```
Codes: K - kernel route, C - connected, L - local, S - static,
R - RIP, O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
f - OpenFabric, t - Table-Direct,
> - selected route, * - FIB route, q - queued, r - rejected, b - backup
t - trapped, o - offload failure
```

```
VRF VPN102:
```

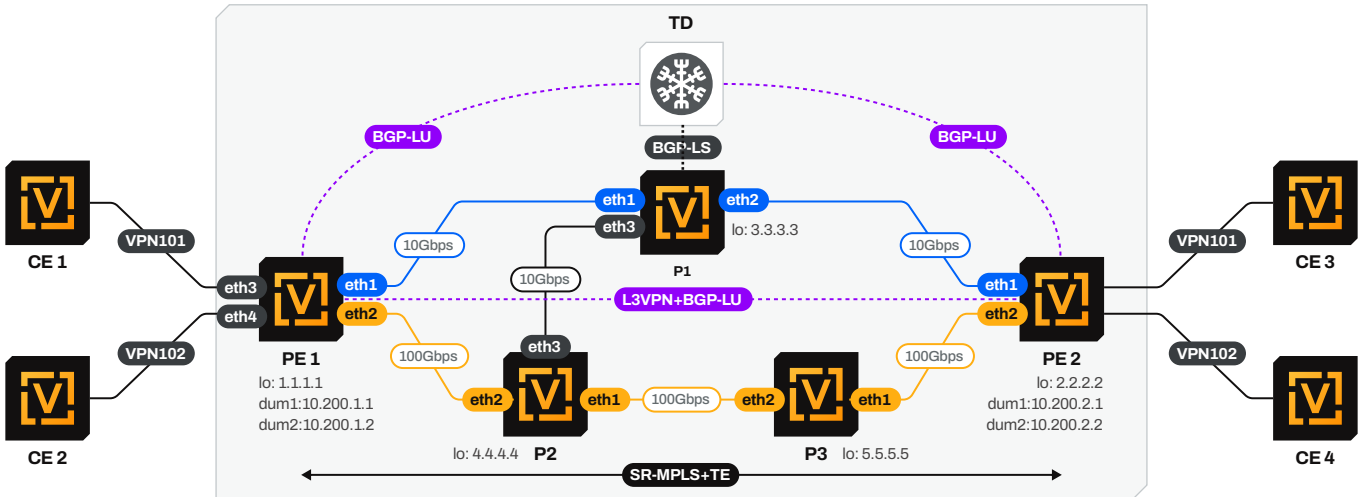
```
C>* 10.100.7.0/24 is directly connected, eth4, weight 1, 01w5d21h
K * 10.100.7.0/24 [0/0] is directly connected, eth4, weight 1, 01w5d21h
L * 10.100.7.1/32 is directly connected, eth4, weight 1, 01w5d21h
L>* 10.100.7.1/32 is directly connected, eth4, weight 1, 01w5d21h
B> 10.100.9.0/24 [200/0] via 2.2.2.2 (vrf default) (recursive), label 80, weight 1, 6d23h19m
*
   via 10.100.1.3, eth1 (vrf default), label 800002/80, weight 1, 6d23h19m
B>* 192.168.2.2/32 [20/0] via 10.100.7.7, eth4, weight 1, 6d22h22m
B> 192.168.4.4/32 [200/0] via 10.200.2.2 (vrf default) (recursive), label 80, weight 1, 00:01:55
*
   via 10.100.1.3, eth1 (vrf default), label 800002/implicit-null/80, weight 1, 00:01:55
```

As we see in our output, both VRFs services are taking the same path when the controller goes down.



Disjointness (Disjoint Group)

Based on our previous configuration and diagram, we will use a different use case: the **Disjoint Group** in Traffic Dictator.



Traffic Dictator uses the disjoint policy to compute two lists of segments that steer traffic from two source nodes to two destination nodes along disjoint paths. The disjoint paths can originate from the same head-end or different head-ends. Disjoint level refers to the type of resources that should not be shared by the two computed paths. The following attribute is used for our test.

- **Link** – Specifies that links are not shared on the computed paths.

Our controllers calculate this attribute and use different paths for our services.

Let's start with our basic configuration:

```
### Add a candidate-path with more preference in both paths

traffic-eng policies
!
policy PE1_PE2_BLUE
  headend 1.1.1.1 topology-id 0
  endpoint 2.2.2.2 service-loopback 10.200.2.1
  install direct labeled-unicast 192.168.0.101
  !
  candidate-path preference 200
  disjoint-group 100 link
!
policy PE1_PE2_YELLOW
  headend 1.1.1.1 topology-id 0
  endpoint 2.2.2.2 service-loopback 10.200.2.2
  install direct labeled-unicast 192.168.0.101
  !
  candidate-path preference 200
  disjoint-group 100 link
```

Let's check how it works with our current setup:

```
## BLUE PATH

TD#show traffic-eng policy PE1_PE2_BLUE detail
Detailed traffic-eng policy information:

Traffic engineering policy "PE1_PE2_BLUE"

Valid config, Active, Installed
Headend 1.1.1.1, topology-id 0
Endpoint 2.2.2.2, service-loopback 10.200.2.1
  Endpoint type: Node, Topology-id: 101, Protocol: isis, Router-id: 0002.0002.0002.00

Setup priority: 7, Hold priority: 7
Reserved bandwidth bps: 0
Install direct, protocol labeled-unicast, peer 192.168.0.101
Policy index: 0, SR-TE distinguisher: 16777216
Route-key: [16777216][10.200.2.1/32]

Candidate paths:
  Candidate-path preference 200
    Path config valid
    Metric: igp
    Disjoint-group: 100
    Link disjoint: True
    Node disjoint: False
    SRLG disjoint: False
    Path-option: dynamic
    This path is currently active
  Candidate-path preference 100
    Path config valid
    Metric: igp
    Path-option: dynamic
    Affinity-set: BLUE_ONLY
      Constraint: include-all
      List: ['BLUE']
      Value: 0x1
    This path has not been checked

Calculation results:
  Aggregate metric: 30
  Topologies: ['101']
  Segment lists:
    [800005, 800002]
  BGP-LU next-hop: 10.100.2.4

Policy statistics:
  Last config update: 2025-12-30 19:29:31.602
  Last recalculation: 2025-12-30 19:41:37.358
  Policy calculation took 0 milliseconds

TD#show traffic-eng policy PE1_PE2_BLUE extensive
Extensive traffic-eng policy information:

Traffic engineering policy "PE1_PE2_BLUE"

Valid config, Active, Installed
Headend 1.1.1.1, topology-id 100
Endpoint 2.2.2.2, service-loopback 10.200.2.1
  Endpoint type: Node, Topology-id: 101, Protocol: isis, Router-id: 0002.0002.0002.00
```



```
Setup priority: 7, Hold priority: 7
Reserved bandwidth bps: 0
Install direct, protocol labeled-unicast, peer 192.168.0.101
Policy index: 0, SR-TE distinguisher: 16777216
Route-key: [16777216][10.200.2.1/32]

-----
IOS-XR commands:
-----
show bgp ipv4 labeled-unicast 10.200.2.1/32
show cef 10.200.2.1/32
-----
JUNOS commands:
-----
*JUNOS doesn't support BGP-LU policies due to nexthop resolution limitations
-----
EOS commands:
-----
show bgp ipv4 labeled-unicast 10.200.2.1/32
show bgp labeled-unicast tunnel | grep 10.200.2.1/32
show tunnel rib 10.200.2.1/32 brief
-----
OcNOS commands:
-----
show ip bgp labeled-unicast 10.200.2.1/32
-----
FRR commands:
-----
show ip bgp 10.200.2.1/32
show mpls table
-----

Candidate paths:
  Candidate-path preference 200
    Path config valid
    Metric: igp
    Disjoint-group: 100
      Link disjoint: True
      Node disjoint: False
      SRLG disjoint: False
    Path-option: dynamic
    This path is currently active
  Candidate-path preference 100
    Path config valid
    Metric: igp
    Path-option: dynamic
    Affinity-set: BLUE_ONLY
      Constraint: include-all
      List: ['BLUE']
      Value: 0x1
    This path has not been checked

Calculation results:
  Aggregate metric: 30
  Topologies: ['101']
  Segment lists:
    -----
  SID list 1:
    Segment type:  PREFIX
    Address family: ipv4
    Prefix:        5.5.5.5
```



```

SID value:      800005

Segment type:   PREFIX
Address family: ipv4
Prefix:         2.2.2.2
SID value:      800002
BGP-LU next-hop: 10.100.2.4

Path details:
Topology-id 101
-----
Nexthop list 1:
0001.0001.0001.00
  [E][L2][I101][N[c65002][b101][s0001.0001.0001.00]][R[c65002][b101][s0004.0004.0004.00]][L[i10.100.2.1][n10.100.2.4]]
0004.0004.0004.00
  [E][L2][I101][N[c65002][b101][s0004.0004.0004.00]][R[c65002][b101][s0005.0005.0005.00]][L[i10.100.4.4][n10.100.4.5]]
0005.0005.0005.00
  [E][L2][I101][N[c65002][b101][s0005.0005.0005.00]][R[c65002][b101][s0002.0002.0002.00]][L[i10.100.5.5][n10.100.5.2]]
0002.0002.0002.00

Policy statistics:
Last config update: 2025-12-30 19:29:31,602
Last recalculation: 2025-12-30 19:41:37.358
Policy calculation took 0 milliseconds

## YELLOW PATH

TD#show traffic-eng policy PE1_PE2_YELLOW detail
Detailed traffic-eng policy information:

Traffic engineering policy "PE1_PE2_YELLOW"

Valid config, Active, Installed
Headend 1.1.1.1, topology-id 0
Endpoint 2.2.2.2, service-loopback 10.200.2.2
  Endpoint type: Node, Topology-id: 101, Protocol: isis, Router-id: 0002.0002.0002.00

Setup priority: 7, Hold priority: 7

Reserved bandwidth bps: 0
Install direct, protocol labeled-unicast, peer 192.168.0.101
Policy index: 1, SR-TE distinguisher: 16777217
Route-key: [16777217][10.200.2.2/32]

Candidate paths:
Candidate-path preference 200
  Path config valid
  Metric: igp
  Disjoint-group: 100
    Link disjoint: True
    Node disjoint: False
    SRLG disjoint: False
  Path-option: dynamic
  This path is currently active
Candidate-path preference 100
  Path config valid
  Metric: igp
  Path-option: dynamic
  Affinity-set: YELLOW_ONLY
    Constraint: include-all
    List: ['YELLOW']
    Value: 0x2
  This path has not been checked

```



Calculation results:

Aggregate metric: 20

Topologies: ['101']

Segment lists:

[800002]

BGP-LU next-hop: 10.100.1.3

Policy statistics:

Last config update: 2025-12-30 19:34:10,103

Last recalculation: 2025-12-30 19:41:37.357

Policy calculation took 0 milliseconds

!

```
show traffic-eng policy PE1_PE2_YELLOW extensive
```

Extensive traffic-eng policy information:

Traffic engineering policy "PE1_PE2_YELLOW"

Valid config, Active, Installed

Headend 1.1.1.1, topology-id 0

Endpoint 2.2.2.2, service-loopback 10.200.2.2

Endpoint type: Node, Topology-id: 101, Protocol: isis, Router-id: 0002.0002.0002.00

Setup priority: 7, Hold priority: 7

Reserved bandwidth bps: 0

Install direct, protocol labeled-unicast, peer 192.168.0.101

Policy index: 1, SR-TE distinguisher: 16777217

Route-key: [16777217][10.200.2.2/32]

IOS-XR commands:

```
show bgp ipv4 labeled-unicast 10.200.2.2/32
```

```
show cef 10.200.2.2/32
```

JUNOS commands:

*JUNOS doesn't support BGP-LU policies due to nexthop resolution limitations

EOS commands:

```
show bgp ipv4 labeled-unicast 10.200.2.2/32
```

```
show bgp labeled-unicast tunnel | grep 10.200.2.2/32
```

```
show tunnel rib 10.200.2.2/32 brief
```

OcNOS commands:

```
show ip bgp labeled-unicast 10.200.2.2/32
```

FRR commands:

```
show ip bgp 10.200.2.2/32
```

```
show mpls table
```

Candidate paths:

Candidate-path preference 200

Path config valid

Metric: igp

Disjoint-group: 100

Link disjoint: True

Node disjoint: False



```

    SRLG disjoint: False
    Path-option: dynamic
    This path is currently active
Candidate-path preference 100
    Path config valid
    Metric: igp
    Path-option: dynamic
    Affinity-set: YELLOW_ONLY
        Constraint: include-all
        List: ['YELLOW']
        Value: 0x2
    This path has not been checked

Calculation results:
    Aggregate metric: 20
    Topologies: ['101']
    Segment lists:
    -----
    SID list 1:

        Segment type:  PREFIX
        Address family: ipv4
        Prefix:         2.2.2.2
        SID value:      800002
    BGP-LU next-hop: 10.100.1.3

Path details:

    Topology-id 100
    -----
    Nexthop list 1:

    0001.0001.0001.00
        [E][L2][I101][N[c65002][b101][s0001.0001.0001.00]][R[c65002][b101][s0003.0003.0003.00]][L[i10.100.1.1][n10.100.1.3]]
    0003.0003.0003.00
        [E][L2][I101][N[c65002][b101][s0003.0003.0003.00]][R[c65002][b101][s0002.0002.0002.00]][L[i10.100.3.3][n10.100.3.2]]
    0002.0002.0002.00

Policy statistics:
    Last config update: 2025-12-30 19:34:10,103
    Last recalculation: 2025-12-30 19:41:37.357
    Policy calculation took 0 milliseconds

```

In our VRFs services which see these new paths:

```

### VRFs VPN101

vyos@PE1:~$ show ip route vrf VPN101
Codes: K - kernel route, C - connected, L - local, S - static,
       R - RIP, O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
       T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
       f - OpenFabric, t - Table-Direct,
       > - selected route, * - FIB route, q - queued, r - rejected, b - backup
       t - trapped, o - offload failure

VRF VPN101:
C* 10.100.6.0/24 is directly connected, eth3, weight 1, 01w5d22h
K * 10.100.6.0/24 [0/0] is directly connected, eth3, weight 1, 01w5d22h
L * 10.100.6.1/32 is directly connected, eth3, weight 1, 01w5d22h
L>* 10.100.6.1/32 is directly connected, eth3, weight 1, 01w5d22h
B> 10.100.8.0/24 [200/0] via 2.2.2.2 (vrf default) (recursive), label 81, weight 1, 01w0d00h

```



```

*                               via 10.100.1.3, eth1 (vrf default), label 800002/81, weight 1, 01w0d00h
B>* 192.168.1.1/32 [20/0] via 10.100.6.6, eth3, weight 1, 6d23h41m
B> 192.168.3.3/32 [200/0] via 10.200.2.1 (vrf default) (recursive), label 81, weight 1, 00:02:15
*                               via 10.100.2.4, eth2 (vrf default), label 800005/800002/81, weight 1, 00:02:15

### VRFs VPN102

vyos@PE1:~$ show ip route vrf VPN102
Codes: K - kernel route, C - connected, L - local, S - static,
        R - RIP, O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP,
        T - Table, v - VNC, V - VNC-Direct, A - Babel, F - PBR,
        f - OpenFabric, t - Table-Direct,
        > - selected route, * - FIB route, q - queued, r - rejected, b - backup
        t - trapped, o - offload failure

VRF VPN102:
C>* 10.100.7.0/24 is directly connected, eth4, weight 1, 01w5d22h
K * 10.100.7.0/24 [0/0] is directly connected, eth4, weight 1, 01w5d22h
L * 10.100.7.1/32 is directly connected, eth4, weight 1, 01w5d22h
L>* 10.100.7.1/32 is directly connected, eth4, weight 1, 01w5d22h
B> 10.100.9.0/24 [200/0] via 2.2.2.2 (vrf default) (recursive), label 80, weight 1, 01w0d00h
*                               via 10.100.1.3, eth1 (vrf default), label 800002/80, weight 1, 01w0d00h
B>* 192.168.2.2/32 [20/0] via 10.100.7.7, eth4, weight 1, 6d23h31m
B> 192.168.4.4/32 [200/0] via 10.200.2.2 (vrf default) (recursive), label 80, weight 1, 00:02:27
*                               via 10.100.1.3, eth1 (vrf default), label 800002/80, weight 1, 00:02:27

```

Conclusion

Traffic Dictator is a powerful SR-TE controller that lowers the entry barrier into SR-TE. It allows you to create different path calculations and enriches the VyOS network capabilities. It integrates powerfully with enterprise, cloud provider, and ISP capabilities to create more complex and robust networks.

The integration between VyOS and Traffic Dictator enables centralized, policy-based traffic control directly on the network edge. VyOS provides sr-mpls routing, firewalling, and packet-processing foundation, while Traffic Dictator adds dynamic traffic classification and decision logic. Together, they allow precise traffic steering, prioritization, and enforcement based on real-time conditions, improving network efficiency, scalability, and operational control.

