Configure BGP VRF Auto RD Auto RT for EVPN on Catalyst 9000 Series Switches

Contents

Introduction

This document describes EVPN simplification CLI for BGP VRF Auto RD and Auto RT in EVPN on Catalyst 9000 series switches.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- Basic BGP configuration
- Basic VRF configuration
- Basic EVPN configuration

Components Used

The information in this document is based on these software and hardware versions:

- Catalyst 9300

- Catalyst 9400
- Catalyst 9500
- Catalyst 9600
- Cisco IOS® XE 17.12.1 and later

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

Layer 3 EVPN deployments involve VRF configurations with many configuration options, including but not limited to, route distinguisher (RD) and route-targets (RT).

- Before the introduction of the BGP VRF Auto RD Auto RT feature at least 5 configuration lines (1 for RD, 4 for RT) would be required to set up a particular VRF for BGP EVPN usage.
- **With BGP VRF Auto RD Auto RT this can be achieved with only 2 lines (potentially one line per VRF if global VRF rd-auto is enabled)**.
- There is no functional difference between Auto RD and static RD. Each RD must be unique within one given router or switch.
- The functional difference between Auto RT and static RT is that Auto RT is only one and same for import and export, regular and stitching, versus static RT can be configured zero to many.
- Also the Auto RT can co-exist with static RT within any particular VRF (you can configure Auto RT in additional to existing static RT prior to this feature).

The auto RD would consist of the BGP router-id plus an internal generated unique number, for example, if the BGP router-id is 192.168.1.1, the auto RD would be like "192.168.1.1:1".

- The auto RT would consist of the BGP AS number plus the vnid being configured. For example: if the BGP AS number is 65000, and the vnid is configured as 123, then the auto RT would be "65000:123".

- This is for both import and export, regular and stitching route-targets.
- If the BGP AS is of 4 byte, then the AS_TRANS is used instead, which is 23456.

The ability to simplify the configuration is highly desirable (if not necessary) for the deployment to be feasible, and has already been widely adopted for the BGP EVPN fabric. This feature is desirable for EVPN, as it helps avoid the writing and maintenance of extensive and complex configurations in Spine-Leaf topologies where many VRFs are configured in a particular leaf.

Note: This feature introduces new CLIs.

Terminology

VRF	Virtual Routing Forwarding	Defines a layer 3 routing domain that be separated from other VRF and global IPv4/IPv6 routing domain
AF	Address Family	Defines which type prefixes and routing info BGP handles
AS	Autonomous System	A set of Internet routable IP prefixes that belong to a network or a collection of networks that are all managed, controlled and supervised by a single entity or organization
RD	Route Distinguisher	Allow BGP to differentiate one prefix from another in different VRFs

RT	Route Target	Route targets are used to constrain routing updates. Determines what prefixes are allowed to be imported by the device
EVPN	Ethernet Virtual Private Network	Extension that allows BGP to transport Layer 2 MAC and Layer 3 IP information is EVPN and uses Multi-Protocol Border Gateway Protocol (MP-BGP) as the protocol to distribute reachability information that pertains to the VXLAN overlay network.
VXLAN	Virtual Extensible LAN (Local Area Network)	VXLAN is designed to overcome the inherent limitations of VLANs and STP. It is a proposed IETF standard [RFC 7348] to provide the same Ethernet Layer 2 network services as VLANs do, but with greater flexibility. Functionally, it is a MAC-in- UDP encapsulation protocol that runs as a virtual overlay on a Layer 3 underlay network.

Configure

Global VRF RD-auto

<#root>

```
Leaf-01#

sh run | include vrf rd-auto

vrf rd-auto                                          <-- Enable Auto RD for all the VRFs

Leaf-01#

sh run | section vrf definition blue


vrf definition blue
vnid 123 evpn-instance                               <-- Enable Auto RT
!
address-family ipv4                                  <-- address-family needs to be specified
 route-target 100:123                                <-- Optionally can have static route-target as requ
exit-address-family
!
```

Per VRF rd-auto Configuration

<#root>

```
Leaf-01#

sh run | section vrf definition green


vrf definition green
rd-auto                                              <-- Enable Auto RD for this VRF green
vnid 35 evpn-instance                                <-- Enable Auto RT
```

```
!
address-family ipv4                                  <-- address-family needs to be specified
exit-address-family
!
address-family ipv6
exit-address-family
```

Note: It is possible to have static and auto RD for different VRFs, but static RD must NOT have the same actual RD as the Auto RD if the Auto RD is assigned first.

Tip: Currently delete the static RD would delete the configuration of the route-targets being configured in the VRFs, as well as the BGP IPv4 and/or IPv6 VRF address-families (and the associated configuration underneath). Therefore deleting an auto RD would have similar behavior. It is recommended not to trigger the deletion of the RD unless absolutely necessary. A change of RD (that is, a delete of the existing RD, either static or auto, and then an add of a new RD, either static or auto, is costly and requires time of delay for the command to go through)

Mixed Static RD and Auto RD

<#root>

```
vrf rd-auto
vrf definition green                                 <-- This VRF green uses auto RD
 vnid 35 evpn-instance
!
 address-family ipv6
 exit-address-family
vrf definition red                                         <-- This VRF red uses static RD
 rd-auto disable
 rd 100:1
 !
 address-family ipv4
  route-target export 100:1
  route-target import 100:1
  route-target export 100:1 stitching
  route-target import 100:1 stitching
 exit-address-family
```

BGP Address-family IPv4 Vrf and Ipv6 Vrf

(This config example is a recap of the existing feature)

<#root>

```
Leaf-01#

show run | sec r bgp

router bgp 65000



<-- Required for Auto RT



bgp router-id 192.168.1.1



<-- Required for Auto RD

 bgp log-neighbor-changes
 no bgp default ipv4-unicast
 neighbor 192.168.1.2 remote-as 65000
 neighbor 192.168.1.2 update-source Loopback0
 neighbor 192.168.1.3 remote-as 65001
 neighbor 192.168.1.3 update-source Loopback0
 !
 address-family ipv4 vrf green
  advertise l2vpn evpn
  redistributed connected
 exit-address-family
 !
 address-family ipv6 vrf green
  advertise l2vpn evpn
  redistribute connected
 exit-address-family
```

Note: The configuration for the other Spine Route Reflector is the same, so are not repeated in this section

Note: Other EVPN leafs can use static RD or RT configuration. As long as the RT matches, the EVPN prefixes are able to import/export to each other.

Verify

Leaf

Verify the Leaf, to have auto RD

<#root>

```
VTEP1#

show vrf blue

  Name                             Default RD            Protocols   Interfaces
  blue                             192.168.1.1:1(auto)   ipv4        Vl34
                                                                     Lo101
                                                                     Et1/1
                                                                     Vl4
                                                                     Vl15
```

<#root>

```
VTEP1#
```

show vrf green

```
  Name                             Default RD            Protocols   Interfaces
  green                            192.168.1.1:2(auto)   ipv6        Lo102
                                                                     Et1/2
                                                                     Vl5
                                                                     Vl13
```

<#root>

```
VTEP1#
```

show vrf detail blue

```
VRF blue (VRF Id = 2); default RD 192.168.1.1:1(auto); default VPNID
  New CLI format, supports multiple address-families
  vnid: 123 evpn-instance vni 35000 core-vlan 34
  Flags: 0x180C
  Interfaces:
    Vl34                     Lo101                    Et1/1
    Vl4                      Vl15
Address family ipv4 unicast (Table ID = 0x2):
  Flags: 0x0
  Export VPN route-target communities
    RT:100:123               RT:65000:123 (auto)
  Import VPN route-target communities
    RT:100:123               RT:65000:123 (auto)
  Export VPN route-target stitching communities
    RT:65000:123 (auto)
  Import VPN route-target stitching communities
    RT:65000:123 (auto)
  No import route-map
  No global export route-map
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix
Address family ipv6 unicast not active
Address family ipv4 multicast not active
Address family ipv6 multicast not active
```

<#root>

```
VTEP1#
```

show vrf detail green

```
VRF green (VRF Id = 4); default RD 192.168.1.1:2(auto); default VPNID
  New CLI format, supports multiple address-families
  vnid: 35 evpn-instance
  Flags: 0x380C
  Interfaces:
    Lo102                    Et1/2                    Vl5
    Vl13
Address family ipv4 unicast not active
Address family ipv6 unicast (Table ID = 0x1E000002):
  Flags: 0x0
  Export VPN route-target communities
    RT:65000:35 (auto)
  Import VPN route-target communities
    RT:65000:35 (auto)
  Export VPN route-target stitching communities
    RT:65000:35 (auto)
  Import VPN route-target stitching communities
    RT:65000:35 (auto)
  No import route-map
  No global export route-map
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix
Address family ipv4 multicast not active
Address family ipv6 multicast not active
```

Troubleshoot

Debugs

If there is an issue with VRF auto RD auto RT you can use debugs to see more about the problem

Enable relevant debugs

<#root>

```
Leaf-01#

debug ip bgp autordrt

Leaf-01#

debug vrf create

Leaf-01#

debug vrf delete
```

Display debug information

<#root>

```
VTEP1#

show debug

VRF Manager:

VRF creation debugging is on
VRF deletion debugging is on
```

```
Packet Infra debugs:
Ip Address Port
------------------------------------------------------|----------
IP routing:
```

BGP auto rd rt debugging is on

Observe the debugs produced at each configuration step

<#root>

```
Leaf-01(config)#
```

vrf definition test

```
*Jun 26 08:19:44.173: LID: Get id @0x7F4414FE4A18 - current A [1..2705] (checking enabled)
*Jun 26 08:19:44.173: LID: AVAIL (verified) - id A
*Jun 26 08:19:44.173: vrfmgr: VRF test: Created vrf_rec with vrfid 0xA
*Jun 26 08:19:44.173: BGP: VRF config event of
```

rd-auto change for vrf test

```
*Jun 26 08:19:44.173: BGP-VPN: bgp vpn global
```

rd-auto for vrf test assigns rd of 192.168.1.1:6

```
*Jun 26 08:19:44.173: BGP: VRF config event of
```

vnid change for vrf test

```
Leaf-01(config-vrf)#
```

vnid 246 evpn-instance

```
% vnid 246 evpn-instance  auto (vni 0 core-vlan 0) is configured in "vrf test"
*Jun 26 08:20:03.466: BGP: VRF config event of
```

vnid change for vrf test

```
Leaf-01(config-vrf)#
```

address-family ipv4

```
*Jun 26 08:20:12.276: vrfmgr: VRF test ipv4 unicast: Received topology create notification
*Jun 26 08:20:12.276: vrfmgr: VRF test ipv4 multicast: Received topology create notification
*Jun 26 08:20:12.276: vrfmgr: VRF test ipv4 unicast:
```

Created vrf_sub_rec with vrfid 0xA, tableid 0xA

```
*Jun 26 08:20:12.276: BGP: VRF config event of vnid change for vrf test
*Jun 26 08:20:12.276: BGP: afi 0 vrf
```

test vnid 246 RT assign

```
*Jun 26 08:20:12.276: BGP: vrf assign auto import stitching rt for VRF test
*Jun 26 08:20:12.276: BGP: vrf assign auto export stitching rt for VRF test
```

```
Leaf-01(config-vrf-af)#
```

address-family ipv6

```
*Jun 26 08:20:20.949: vrfmgr: VRF test ipv6 unicast: Received topology create notification
*Jun 26 08:20:20.949: vrfmgr: VRF test ipv6 multicast: Received topology create notification
*Jun 26 08:20:20.949: vrfmgr: VRF test ipv6 unicast:
```

Created vrf_sub_rec with vrfid 0xA, tableid 0x1E000004

```
*Jun 26 08:20:20.949: BGP: VRF config event of vnid change for vrf test
*Jun 26 08:20:20.949: BGP:
```

afi 0 vrf test vnid 246 RT assign

```
*Jun 26 08:20:20.949: BGP: vrf assign auto import stitching rt for VRF test
*Jun 26 08:20:20.949: BGP: vrf assign auto export stitching rt for VRF test
*Jun 26 08:20:20.949: BGP:
```

afi 1 vrf test vnid 246 RT assign

```
*Jun 26 08:20:20.949: BGP: vrf assign auto import stitching rt for VRF test
*Jun 26 08:20:20.949: BGP: vrf assign auto export stitching rt for VRF test

Leaf-01(config-vrf-af)#
```

do sh vrf detail test

VRF test (VRF Id = 10)

```
; default
```

RD 192.168.1.1:6(auto)

```
; default VPNID
```

<-- VRF ID = 10 (hex 0xA) | auto RD assigned matches debug "assigns rd of 192.168.1.1:6"

```
  New CLI format, supports multiple address-families
```

vnid: 246

```
 evpn-instance
  Flags: 0x180C
  No interfaces
Address family ipv4 unicast (Table ID = 0xA):
  Flags: 0x0
  Export VPN route-target communities
    RT:65000:246 (auto)
  Import VPN route-target communities
    RT:65000:246 (auto)
  Export VPN route-target stitching communities
    RT:65000:246 (auto)
  Import VPN route-target stitching communities
    RT:65000:246 (auto)
  No import route-map
  No global export route-map
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix
Address family ipv6 unicast
```

(Table ID = 0x1E000004)

```
:
```

<-- ID matches debug

```
 "
```

```
Created vrf_sub_rec with vrfid 0xA, tableid 0x1E000004"
```

```
  Flags: 0x0
  Export VPN route-target communities
    RT:65000:246 (auto)
  Import VPN route-target communities
    RT:65000:246 (auto)
  Export VPN route-target stitching communities
    RT:65000:246 (auto)
  Import VPN route-target stitching communities
    RT:65000:246 (auto)
  No import route-map
  No global export route-map
  No export route-map
  VRF label distribution protocol: not configured
  VRF label allocation mode: per-prefix
Address family ipv4 multicast not active
Address family ipv6 multicast not active

Leaf-01(config-vrf-af)#
```

```
do sh run vrf test
```

```
Building configuration...

Current configuration : 145 bytes
vrf definition test
 vnid 246 evpn-instance
 !
 address-family ipv4
 exit-address-family
 !
 address-family ipv6
 exit-address-family
```

Catalyst and Nexus Interoperability

Problem

By default, Nexus assigns vni-based route-targets (ASN:VNI), while Catalyst assigns evi-based route-targets (ASN:EVI).

When route-target are mismatched you can observe symptoms such as these:

- BGP connection for the L2VPN EVPN establishes, and type 3 routes are visible in the BGP table
- NVE peering not established
- Tunnel adjacency remains incomplete

Remediation

There are a couple options to fix this interop issue

1. Configure manual route-targets on one side, so they match
2. Configure the C9500 to assign vni-based route-targets using 'route-target auto vni'

Apply these cli (for option 2) under the l2vpn evpn section

<#root>

```
address-family l2vpn evpn

rewrite-evpn-rt-asn  <---
```

Related Information

- BGP EVPN VXLAN Configuration Guide, Cisco IOS XE Dublin 17.11.x (Catalyst 9500 Switches)
- Technical Support & Documentation - Cisco Systems