How to Troubleshoot the MPLS VPN

Document ID: 13734

Contents

Introduction

This document shows you how to troubleshoot the Configuring a Basic MPLS VPN document. We recommend you read this sample configuration and view the network diagram before you use this document.

Configuring a Basic MPLS VPN shows a fully functional MPLS backbone network which means provider edge (PE) routers are able to reach each other through the backbone. Refer to the MPLS Verification and Troubleshooting Support Page for information on troubleshooting an MPLS network.

Before establishing an MPLS VPN, you must be able to ping PE router A (10.10.10.4) from PE router B (10.10.10.6) and vice−versa.

Remember that VPN routing/forwarding instance (VRF) names are case sensitive, for example, `Customer_A` is not the same as `customer_a`.

Prerequisites

Requirements

Readers of this document should be familiar with:

- Configuring a Basic MPLS VPN

Components Used

This document is not restricted to specific software and hardware versions.

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, make sure

that you understand the potential impact of any command.

Conventions

For more information on document conventions, refer to the Cisco Technical Tips Conventions.

Troubleshooting VRF Configurations

show ip vrf [vrf−name]

The **show ip vrf [***vrf−name***]** command shows a summary of all VRFs present on the current router and their associated route−distinguishers and interface(s).

```
Pesaro# show ip vrf
  Name                              Default RD          Interfaces
  Customer_A                        100:101             Loopback101
                                                        Loopback111
  Customer_B                        100:102             Loopback102
```

This command allows you to verify:

- The configuration of VRFs (and their names).
- That each route−distinguisher (RD) is the same at each concerned PE.

show ip vrf [{detail | interfaces}] vrf−name

The **show ip vrf [{detail | interfaces}]** *vrf−name* command shows detailed configurations about the VRF.

```
Pesaro# show ip vrf detail Customer_A
VRF Customer_A; default RD 100:101
  Interfaces:
    Loopback101              Loopback111
  Connected addresses are not in global routing table
  Export VPN route−target communities
    RT:100:1001
  Import VPN route−target communities
    RT:100:1001
  No import route−map
  No export route−map

Pesaro# show ip vrf interfaces
Interface              IP−Address      VRF              Protocol
Loopback101            200.0.6.1       Customer_A       up
Loopback111            200.1.6.1       Customer_A       up
Loopback102            200.0.6.1       Customer_B       up
```

These commands allow you to verify:

- That connected addresses are not in the global routing table.
- The routing attributes of each VRF. What is exported on one side should be imported somewhere else.
- The interface status (and IP addresses) of interfaces.

Routing Information

Use the same commands you use to verify the global routing table with the extensions shown in this section to verify routing tables or routing protocol databases.

Routing Table

To check the routing table, Add the `vrf [vrf−name]` extension to the **show ip route** command to verify the routing table, as shown here:

```
Pescara# show ip route vrf Customer_A
Codes: C − connected, S − static, I − IGRP, R − RIP, M − mobile, B − BGP
       D − EIGRP, EX − EIGRP external, O − OSPF, IA − OSPF inter area
       N1 − OSPF NSSA external type 1, N2 − OSPF NSSA external type 2
       E1 − OSPF external type 1, E2 − OSPF external type 2, E − EGP
       i − IS−IS, L1 − ISIS level−1, L2 − ISIS level−2, ia − ISIS inter area
       * − candidate default, U − per−user static route, o − ODR
       P − periodic downloaded static route

Gateway of last resort is not set

B    200.0.6.0/24 [200/0] via 10.10.10.6, 00:42:14
B    200.1.6.0/24 [200/0] via 10.10.10.6, 00:42:14
C    200.0.4.0/24 is directly connected, Loopback101
```

You can also use the **show ip route vrf Customer_A 1.2.3.4** command to verify the destination for a particular address.

BGP

Border Gateway Protocol (BGP) is used between the PE routers and is necessary for inter−site connectivity. In this example, we use internal BGP (iBGP). You can also use external BGP (eBGP) as an external routing protocol for PE−CE route propagation.

You can use these commands to troubleshoot BGP:

- **show ip bgp neighbors**
- **show ip bgp vpnv4 all** (or **show ip bgp vpnv4 vrf [***VRF name***]**)
- **show ip bgp vpnv4 vrf** *VRF name* **tags** (this command is VPN/MPLS specific)
- **show ip bgp vpnv4 vrf** *VRF name A.B.C.D*

For example:

```
Pescara# show ip bgp vpnv4 vrf Customer_A
BGP table version is 40, local router ID is 10.10.10.4
Status codes: s suppressed, d damped, h history, * valid, > best, i − internal
Origin codes: i − IGP, e − EGP, ? − incomplete

   Network          Next Hop            Metric LocPrf Weight Path
Route Distinguisher: 100:101 (default for vrf Customer_A)
*>i200.0.6.0        10.10.10.6               0    100      0 ?
*> 200.0.4.0        0.0.0.0                  0         32768 ?
*>i200.1.6.0        10.10.10.6               0    100      0 ?
```

Refer to the BGP Support Pages for more information on troubleshooting BGP issues.

PE−CE Routing Protocol

If the routing protocol used on the customer side isn't BGP, you can use traditional show commands, and apply them to the correct VRF.

Use the **show ip rip database vrf [***VRF name***]** command if you use Routing Information Protocol (RIP). For example:

```
Alcazaba# show ip rip database vrf vrf101
       0.0.0.0/0 auto-summary
       0.0.0.0/0
       [2] via 150.150.0.2, 00:00:12, Ethernet1/1
       6.0.0.0/8 auto-summary
       6.6.6.6/32 redistributed
       [1] via 223.0.0.21,
       7.0.0.0/8 auto-summary
       7.7.7.0/24
       [1] via 150.150.0.2, 00:00:12, Ethernet1/1
       10.0.0.0/8 auto-summary
       10.0.0.0/8 redistributed
       [1] via 125.2.2.2,
       10.0.0.0/16
       [1] via 150.150.0.2, 00:00:12, Ethernet1/1
       10.200.8.0/22
```

Use the **show ip ospf [***process-id area-id***] database** command and specify the correct process number if you use OSPF. For example:

```
Alcazaba# show ip ospf 2 database

                  OSPF Router with ID (222.0.0.10) (Process ID 2)

                      Router Link States (Area 1)

      Link ID         ADV Router      Age         Seq#       Checksum Link count
      222.0.0.1       222.0.0.1       1364        0x80000013 0x7369   3
      222.0.0.10      222.0.0.10      1363        0x80000002 0xFEFE   2

                      Net Link States (Area 1)

      Link ID         ADV Router      Age         Seq#       Checksum
      150.150.0.1     222.0.0.10      1363        0x80000001 0xEC6D

                      Summary Net Link States (Area 1)

      Link ID         ADV Router      Age         Seq#       Checksum
      6.6.6.6         222.0.0.10      1328        0x80000001 0x4967
      69.69.0.0       222.0.0.10      1268        0x80000001 0x2427
      222.0.0.3       222.0.0.10      1328        0x80000001 0xEEF7
      222.0.0.30      222.0.0.10      1268        0x80000001 0x7B5A
```

This command allows you to verify:

- If the routing table is correct (from a customer point of view), or what is missing from the routing table.
- That BGP is up and working (or you can see which neighbor is missing).

Labels

MPLS VPN uses a two-level label stack. One of the labels is used to identify the VRF and is set up between the two PEs. The other label (on the top of the stack) is the "backbone" label, set up by the standard MPLS network.

You can use the **traceroute VRF [***vrf-name***] A.B.C.B** command to verify transport labels.

Note: This command only works with a MPLS−aware traceroute, if the backbone routers are configured to propagate and generate IP Time to Live (TTL) information. Refer to the documentation on the **mpls ip propagate−ttl** command for more information.

```
Pesaro# traceroute vrf Customer_B 200.0.4.1

Type escape sequence to abort.
Tracing the route to 200.0.4.1

  1 10.1.1.21 [MPLS: Labels 25/28 Exp 0] 464 msec 280 msec 308 msec
  2 10.1.1.5 [MPLS: Labels 22/28 Exp 0] 236 msec 572 msec 228 msec
  3 200.0.4.1 108 msec *  100 msec
```

The absence of 10.1.1.14 in this traceroute is normal due to the MPLS/VPN architecture.

You can use the **show ip bgp vpnv4 all tags** command to obtain more precise output, like the labels table for a particular VRF, for example:

```
Pescara# show ip bgp vpnv4 all tags
   Network          Next Hop      In tag/Out tag
Route Distinguisher: 100:101 (Customer_A)
   200.0.6.0        10.10.10.6      notag/28
   200.0.4.0        0.0.0.0         16/aggregate(Customer_A)
   200.1.6.0        10.10.10.6      notag/29
Route Distinguisher: 100:102 (Customer_B)
   200.0.6.0        10.10.10.6      notag/30
   200.0.4.0        0.0.0.0         28/aggregate(Customer_B)
```

You can also use the traditional **show ip cef** command:

```
Pescara# show ip cef vrf Customer_B detail
IP CEF with switching (Table Version 10), flags=0x0
  8 routes, 0 reresolve, 0 unresolved (0 old, 0 new)
  46 leaves, 51 nodes, 54640 bytes, 361 inserts, 315 invalidations
  0 load sharing elements, 0 bytes, 0 references
  universal per−destination load sharing algorithm, id F968AD29
  5 CEF resets, 38 revisions of existing leaves
  refcounts:  1400 leaf, 1392 node

Adjacency Table has 2 adjacencies
0.0.0.0/32, version 0, receive
200.0.6.0/24, version 9, cached adjacency to Serial0/1.1
0 packets, 0 bytes
  tag information set
    local tag: VPN−route−head
    fast tag rewrite with Se0/1.1, point2point, tags imposed: {20 30}
  via 10.10.10.6, 0 dependencies, recursive
    next hop 10.1.1.13, Serial0/1.1 via 10.10.10.6/32
    valid cached adjacency
    tag rewrite with Se0/1.1, point2point, tags imposed: {20 30}
200.0.4.0/24, version 6, attached, connected
0 packets, 0 bytes
  tag information set
    local tag: 28
  via Loopback102, 0 dependencies
    valid discard adjacency
    tag rewrite with , , tags imposed: {}
200.0.4.0/32, version 4, receive
200.0.4.1/32, version 3, receive
200.0.4.255/32, version 5, receive
224.0.0.0/24, version 2, receive
255.255.255.255/32, version 1, receive
```

This command allows you to verify:

- That labels are effectively used.
- That a stack of (at least) two labels is used for VPN destinations.

Test

You can use the **ping** command to verify that the VRF works, but if you are on a PE router, you must indicate the specific VRF name.

```
Pescara# ping vrf Customer_A 200.0.6.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 200.0.6.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 176/264/576 ms
```

Related Information

- **MPLS Support Page**
- **IP Routing Support Page**
- **Technical Support − Cisco Systems**

Updated: Nov 16, 2007 Document ID: 13734