MPLS VPN

March 25, 2020

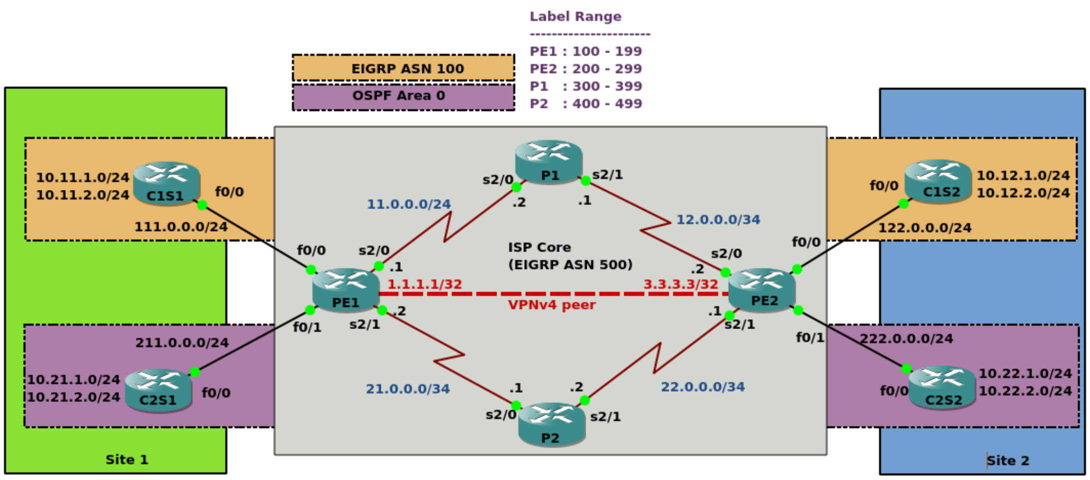
1 MPLS VPN how it works ?

There are two major VPN Models 1. Overlay model (GRE, IPSec) 2. Peer to peer models (MPLS)

1.1 MPLS VPN

- Traffic entern into ISP as normal IP packets
- PE router generates and *imposes* a label
  - MPLS works on layer 2.5
- P routers forward the traffic by swapping the label to terminal PE where it gets popped
  - P router maintains customer routes (but not the IP)
- Combines benifits of Overlay and P2P model
- PE maintains routes of eace customer in separate VRF routing table
- Between PE - PE a VPNv4 peering tunnel is established (for large network DMVPN)
- Packtes are forwarded without seeing the IP address (no routing table lookup)
- **BGP-free Core**

2 Steps to configure MPLS L3 VPN



1. **Configure IGP inside SP core**

2. **Configure MPLS LDP inside the SP core :** LDP binding is created based on IGP information
3. **Create VRF, RD and RT :** Configured on PE routers, assign customer facing interfaces to respective VRFs. RD and RT values to be assigned to resolve any prefix conflict.
4. **Configure routing between PE and CE (IGP or BGP) :** Advertise customer routes into PE. VRFs are populated by customer routes
5. **Configure VPNv4 peering between PEs :** Eshtablish MPLS tunnel between end-to-end (PE-PE). Its one of the implementation of MP-BGP.
6. **Configure Redistribution on PE routers :** Inject the cutomer routes from VRF to MP-BGP tunnel on and vice-versa.

3 VRF , RD and RT

3.1 Virtual Routing and Forwarding

- When a same PE is connedted to multiple CE. It maintains custer specific routes into virtual routing tables called VRF
- CEF creates virtual RIB and FIB for each VRF and manages them parallelly.
- Without VRF, all customer routes would have stored into the global routing table, which may lead them to be advertised to other customer. Since cusomer isolation is required **ACL based route filtering** was used. The problem is, this is not a scalable solution.
- All non VRF routes are placed into global routing table
- Route-sharing among VRFs are not possible by default untill addition confing are done
- VRFs are defined on customer facing interfaces.

3.2 Route Distinguisher value

- Mandatory for each VRF. 32 bit IPv4 + 96 bits Prepend prefic (RD values)= unique 96 bits address called (VPNv4 address)
  - eg. 500:1 + 10.0.0.0
  - naming convension : 500 represents ASN and 1 is index number
  - each RD is 1:1 with a VRF
- uniquely identify customer IPv4 address inside ISP core as Multiple customer may use overlapping routes.

3.3 Route Target value

- Identifies VPN membership
- A route can be assigned with multiple RT values
- We RD is not enough ?
  - Let an ISP has multiple customer with multiple sites and some shared servers which must be access by all customer
  - site-to-site reachibility for customers is ensured by RD but it prevents shared access.
  - Between PEs we have a MP-BGP tunnel that caries all VPN routes.With that another Extended-Community feature called *Route-target* values

  - VRFs are given RT values, all routes from that VRF caries respective RT values (Export RT). The other end of PE matches a set of RT (Import RT) values specific to VRFs on that router. If there's a match it accepts the route.

  - prefix specifig RT import/export is also possible using export map (Advanced concept)
- Looks similar to RD but serves different purpose
- To give all customer access to a specific shared server.
  1. create a VRF in router connected to shared server. and export a RT = 500:10
  2. create VRFs per customer in other CE routers and import RT = 500:10
  3. In similar way routes between VRFs can be also shared
- Export RT (Egress)
  - identifies VPN membership (which VPN tunnel a route is assigned to)
  - attached to client route when it is converted to a VPNv4 route
- Import RT (Ingress)
  - identifies which VPNv4 routes to inject in a VRF
  - A VPNv4 route is accepted by a VRF is at least one RT values among its tags matches to the those specified on the VRF.

4 Step 1 & 2 : Base Config

4.0.1 Interface Config

```
!c1s1
conf t
    int f0/0
        ip add  111.0.0.11 255.255.255.0
        no sh
    int l0
        ip add 10.11.1.1 255.255.255.0
    int l1
        ip add 10.11.2.1 255.255.255.0

end

!c1s2
conf t
    int f0/0
        ip add  122.0.0.12 255.255.255.0
        no sh
    int l0
        ip add 10.12.1.1 255.255.255.0
    int l1
        ip add 10.12.2.1 255.255.255.0

end
```

```
!c2s1
conf t
    int f0/0
        ip add  211.0.0.21 255.255.255.0
        no sh
    int l0
        ip add 10.21.1.1 255.255.255.0
    int l1
        ip add 10.21.2.1 255.255.255.0

end

!c2s2
conf t
    int f0/0
        ip add  222.0.0.22 255.255.255.0
        no sh
    int l0
        ip add 10.22.1.1 255.255.255.0
    int l1
        ip add 10.22.2.1 255.255.255.0

end

!pe1
conf t
    int f0/0
        ip add 111.0.0.1 255.255.255.0
        no sh
    int f0/1
        ip add 211.0.0.1 255.255.255.0
        no sh
    int s2/0
        ip add 11.0.0.1 255.255.255.0
        no sh
    int s2/1
        ip add 21.0.0.2 255.255.255.0
        no sh
    int l0
        ip add 1.1.1.1 255.255.255.255
end

!pe2
conf t
    int f0/0
        ip add 122.0.0.2 255.255.255.0
        no sh
    int f0/1
```

```
        ip add 222.0.0.2 255.255.255.0
        no sh
    int s2/0
        ip add 12.0.0.2 255.255.255.0
        no sh
    int s2/1
        ip add 22.0.0.1 255.255.255.0
        no sh
    int l0
        ip add 3.3.3.3 255.255.255.255
end

!p1
conf t
    int s2/0
        ip add 11.0.0.2 255.255.255.0
        no sh
    int s2/1
        ip add 12.0.0.1 255.255.255.0
        no sh
    int l0
        ip add 2.2.2.2 255.255.255.255
end

!p2
conf t
    int s2/0
        ip add 21.0.0.1 255.255.255.0
        no sh
    int s2/1
        ip add 22.0.0.2 255.255.255.0
        no sh
    int l0
        ip add 4.4.4.4 255.255.255.255
end
```

4.1 Step 1: ISP Core routing

```
!c1
conf t
    router eigrp 100
        no auto
        passive def
        net 0.0.0.0
        no passive f0/0
end
```

```
!c2
conf t
    router ospf 1
        passive def
        net 0.0.0.0 255.255.255.255 area 0
        no passive f0/0
end

!ISP
conf t
    router eigrp 500
        no auto
        passive def
        net 0.0.0.0
        no passive s2/0
        no passive s2/1
end
```

4.2 Step 2: MPLS Configuration

- if your core routing protocol is OSPF, make sure of setting Loopbacks as point to point.

```
!pe1
conf t
    mpls ip
    mpls label range 100 199
    int s2/0
        mpls ip
    int s2/1
        mpls ip
end

!pe2
conf t
    mpls ip
    mpls label range 200 299
    int s2/0
        mpls ip
    int s2/1
        mpls ip
end

!p1
conf t
    mpls ip
    mpls label range 300 399
    int s2/0
```

```
        mpls ip
    int s2/1
        mpls ip
end

!p2
conf t
    mpls ip
    mpls label range 400 499
    int s2/0
        mpls ip
    int s2/1
        mpls ip
end
```

5 Step 3 & 4 : VRF Configuration and PE-CE routing

5.1 Configuration

```
!PE1
conf t
    ip vrf cust_1
        rd 500:1
        route-target import 500:1
        route-target export 500:1
    ip vrf cust_2
        rd 500:2
        route-target import 500:2
        route-target export 500:2

    int f0/0
        ip vrf forwarding cust_1
        ip add 111.0.0.1 255.255.255.0
        no sh
    int f0/1
        ip vrf forwarding cust_2
        ip address 211.0.0.1 255.255.255.0
        no sh

    router eigrp 100
        address-family ipv4 unicast vrf cust_1
            autonomous-system 100
            network 0.0.0.0
        exit

    router ospf 1 vrf cust_2
```

```
        network 0.0.0.0 255.255.255.255 area 0
end

!PE2
conf t
    ip vrf cust_1                  ! create vrf for customer 1
        rd 500:1                   ! route distinguisher <asn>:<nn>
        route-target import 500:1  ! route target <asn>:<nn>
        route-target export 500:1
    ip vrf cust_2                  ! create vrf for customer 2
        rd 500:2
        route-target import 500:2
        route-target export 500:2

    int f0/0
        ip vrf forwarding cust_1       ! assign interface into VRF
        ip add 122.0.0.2 255.255.255.0 ! VRF assignment removes IP addr
        no sh
    int f0/1
        ip vrf forwarding cust_2
        ip address 222.0.0.2 255.255.255.0
        no sh

    router eigrp 100
        address-family ipv4 unicast vrf cust_1 ! assign EIGRP to VRF
            autonomous-system 100              ! ASN for VRF
            network 0.0.0.0                    ! advert
        exit

    router ospf 1 vrf cust_2                   ! assign OSPF to VRF
        network 0.0.0.0 255.255.255.255 area 0 ! advert
end
```

5.2 Verification

5.2.1 VRF interface

```
PE1#sh ip vrf interfaces

Interface              IP-Address       VRF                                 Protocol
Fa0/0                  111.0.0.1        cust_1                              up
Fa0/1                  211.0.0.1        cust_2                              up
```

5.2.2 RD value

```
PE1#sh ip vrf brief

  Name                             Default RD          Interfaces
  cust_1                           500:1               Fa0/0
  cust_2                           500:2               Fa0/1
```

5.2.3 VRF routing tables

```
PE1#sh ip route vrf cust_1 eigrp

     10.0.0.0/24 is subnetted, 2 subnets
D       10.11.1.0 [90/156160] via 111.0.0.11, 00:06:57, FastEthernet0/0
D       10.11.2.0 [90/156160] via 111.0.0.11, 00:06:57, FastEthernet0/0

PE1#sh ip route vrf cust_2 ospf

Routing Table: cust_2

     10.0.0.0/32 is subnetted, 2 subnets
O       10.21.2.1 [110/2] via 211.0.0.21, 00:05:59, FastEthernet0/1
O       10.21.1.1 [110/2] via 211.0.0.21, 00:05:59, FastEthernet0/1


PE2#sh ip route vrf cust_1 eigrp

     10.0.0.0/24 is subnetted, 2 subnets
D       10.12.2.0 [90/156160] via 122.0.0.12, 00:03:39, FastEthernet0/0
D       10.12.1.0 [90/156160] via 122.0.0.12, 00:03:39, FastEthernet0/0

PE2#sh ip route vrf cust_2 ospf

Routing Table: cust_2

     10.0.0.0/32 is subnetted, 2 subnets
O       10.22.1.1 [110/2] via 222.0.0.22, 00:03:33, FastEthernet0/1
O       10.22.2.1 [110/2] via 222.0.0.22, 00:03:33, FastEthernet0/1
PE2#
```

5.2.4 Ping

standard ping queries on the global routing table, hence it does't work for VRF routs. You need to tell ping query specific VRF

```
PE1#ping 10.11.1.1             ! lookup at global RT
```

```
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.11.1.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)

PE1#ping vrf cust_1 10.11.1.1  ! lookup at specific VRF

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 10.11.1.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 8/18/24 ms
```

6 Step 5 : VPNv4 Peering (PE-PE Tunnel)

- **MP-BGP Address families**
  - IPv4 : stanard routing on IPv4 (Default)
  - IPv6 : standard routing on IPv6
  - VPNv4 : MPLS VPN over IPv4
  - VPNv6 : MPLS VPN over IPv6
- Create a iBGP peering between loopback interfaces of PE routers

6.1 Configuration

6.1.1 Post check

fist check the L3 reachibility of neighbours with MPLS (LSP)

```
PE1#trace 3.3.3.3

Type escape sequence to abort.
Tracing the route to 3.3.3.3

  1 21.0.0.1 [MPLS: Label 402 Exp 0] 24 msec
    11.0.0.2 [MPLS: Label 301 Exp 0] 28 msec
    21.0.0.1 [MPLS: Label 402 Exp 0] 32 msec
  2 12.0.0.2 16 msec
    22.0.0.1 44 msec
    12.0.0.2 24 msec
```

6.1.2 VPNv4 Config

Now start configuring the routers

```
!pe1
conf t
    router bgp 500
```

```
        no bgp default ipv4-unicast            ! revoke default behaviour

        nei 3.3.3.3 remote-as 500              ! iBGP peering
        nei 3.3.3.3 update-source loop0        ! load balancing

        address-family vpnv4 unicast           ! MPLS VPN over IPv4
            nei 3.3.3.3 activate               ! initiate session
            nei 3.3.3.3 send-community extended ! to carry RT value
            nei 3.3.3.3 next-hop self          ! next hop = bgp peer
        exit
end

!pe1
conf t
    router bgp 500
        no bgp default ipv4-unicast

        nei 1.1.1.1 remote-as 500
        nei 1.1.1.1 update-source loop0

        address-family vpnv4 unicast
            nei 1.1.1.1 activate
            nei 1.1.1.1 send-community extended
            nei 1.1.1.1 next-hop self
        exit
end
```

6.1.3 Verification

- Similar to `ip bgp summary` but this command won't show you anything as default bgp is not running.
- notice that prefix = 0, the customer routes will come after redistribution (step 6)

```
PE1#sh ip bgp vpnv4 all summary

BGP router identifier 1.1.1.1, local AS number 500
BGP table version is 1, main routing table version 1

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
3.3.3.3         4   500      27      27        1    0    0 00:23:57        0


PE2#sh ip bgp vpnv4 all summary

BGP router identifier 3.3.3.3, local AS number 500
BGP table version is 1, main routing table version 1
```

```
Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
1.1.1.1         4   500      26      26        1    0    0 00:22:47        0
```

7 Step 6 : Redistribute PR routes into VPNv4 BGP

- Redistribution must be done on both side if dynamic routing is used
- for static routing its one way static –> bgp
- all redistribution must take place under VRF

```
!pe
conf t
    router bgp 500
        address-family ipv4 vrf cust_1
            redistribute eigrp 100 metric 156160
        address-family ipv4 vrf cust_2
            redistribute ospf 1 vrf cust_2 metric 2
    router eigrp 100
        address-family ipv4 vrf cust_1
            redistribute bgp 500 metric 1544 100 255 1 1500
        exit
    router ospf 1 vrf 1
        redistribute bgp 500 metric 100 subnets
```

7.1 Verification

```
PE1#sh ip route vrf cust_1

     111.0.0.0/24 is subnetted, 1 subnets
C       111.0.0.0 is directly connected, FastEthernet0/0
     10.0.0.0/24 is subnetted, 4 subnets
D       10.11.1.0 [90/156160] via 111.0.0.11, 01:56:52, FastEthernet0/0
D       10.11.2.0 [90/156160] via 111.0.0.11, 01:56:52, FastEthernet0/0
B       10.12.2.0 [200/156160] via 3.3.3.3, 00:02:23
B       10.12.1.0 [200/156160] via 3.3.3.3, 00:02:23
     122.0.0.0/24 is subnetted, 1 subnets
B       122.0.0.0 [200/0] via 3.3.3.3, 00:02:23


PE1#sh ip route vrf cust_2

B    222.0.0.0/24 [200/0] via 3.3.3.3, 00:02:37
     10.0.0.0/32 is subnetted, 4 subnets
B       10.22.1.1 [200/2] via 3.3.3.3, 00:02:37
O       10.21.2.1 [110/2] via 211.0.0.21, 00:07:43, FastEthernet0/1
B       10.22.2.1 [200/2] via 3.3.3.3, 00:02:37
O       10.21.1.1 [110/2] via 211.0.0.21, 00:07:43, FastEthernet0/1
```

```
C    211.0.0.0/24 is directly connected, FastEthernet0/1

PE1#sh ip route vrf cust_1

     111.0.0.0/24 is subnetted, 1 subnets
B       111.0.0.0 [200/0] via 1.1.1.1, 00:09:50
     10.0.0.0/24 is subnetted, 4 subnets
B       10.11.1.0 [200/156160] via 1.1.1.1, 00:09:50
B       10.11.2.0 [200/156160] via 1.1.1.1, 00:09:50
D       10.12.2.0 [90/156160] via 122.0.0.12, 01:54:53, FastEthernet0/0
D       10.12.1.0 [90/156160] via 122.0.0.12, 01:54:53, FastEthernet0/0
     122.0.0.0/24 is subnetted, 1 subnets
C       122.0.0.0 is directly connected, FastEthernet0/0


PE1#sh ip route vrf cust_2

C    222.0.0.0/24 is directly connected, FastEthernet0/1
     10.0.0.0/32 is subnetted, 4 subnets
O       10.22.1.1 [110/2] via 222.0.0.22, 00:00:36, FastEthernet0/1
B       10.21.2.1 [200/2] via 1.1.1.1, 00:09:11
O       10.22.2.1 [110/2] via 222.0.0.22, 00:00:36, FastEthernet0/1
B       10.21.1.1 [200/2] via 1.1.1.1, 00:09:11
B    211.0.0.0/24 [200/0] via 1.1.1.1, 00:09:11
```

[ ]:

```
```