

# VRF Support for CUPS


- Revision History, on page 1
- Feature Description, on page 1
- Configuring VRF, on page 3
- Monitoring and Troubleshooting, on page 5


## Revision History

**Note** Revision history details are not provided for features introduced before release 21.24.

| Revision Details | Release |
|---|---|
| First introduced | Pre 21.24 |

## Feature Description

The VRF Support for CUPS feature enables association of IP pools with virtual routing and forwarding (VRF). These IP pools are chunked like any pools. The chunks from this pool are allocated to the User Planes (UPs) that are configured to use these pools. As in the existing deployment, VRF-associated pools in CUPS can only be of type—STATIC or PRIVATE.

The chunks from the PRIVATE VRF pool are allocated when the UP comes for registration similar to the normal private pools. The chunks from the STATIC VRF pool are allocated only when calls come up for that chunk, similar to normal static pools.

**Note** VRF limit per UP is 205.

### Overlapping Pools in Same UP

Overlapping pools share and use an IP range. Overlapping pools can either be of type STATIC or PRIVATE. No public pools can be configured as overlapping pools. Each overlapping pool is part of different VRF (routing domain) and pool-group. Since an APN can use only one pool-group, overlapping pools are part of different APN as well.

Without this functionality, overlapping pools can be configured at CP but chunks from two overlapping pools can't be sent to same UP. That is, the UP can't handle chunks from two different overlapping pools. So, same number of UPs and overlapping pools are required for sharing same IP range.

With this functionality, UP can handle chunks from two different overlapping pools. So, a single UP can handle any number of overlapping pools sharing the same IP range.

**Note** Only VRF-based overlapping pools are supported in CUPS. Other flavors of overlapping pools, like NH-based, VLAN-based, and so on, aren't supported in CUPS.

The functionality of overlapping pools in same UP includes:

- When a chunk from particular pool is installed on an UP, its corresponding vrf-name is sent along with the chunk.
- The UPs are made VRF-aware of chunks and therefore, UPs install chunks on the corresponding VRFs and the chunk database is populated under the VRFs.
- During call allocation, release, recovery, or any communication towards VPNMgr, the corresponding SessMgr at UP includes vrf-id. This enables VPNMgr to pick the correct chunk for that IP under the provided vrf-id for processing.

# VPNMgr Crash Outage Improvement for IP Pool under VRF

In case of Demux card migration or if VPNMgr goes down, new calls are rejected until VPNMgr rebuilds its database. For enterprise solutions where there are lots of VRFs, the new call impact may be higher than expected.

The Delayed VRF Programming, a CLI-controlled feature, is introduced to reduce the new call impact by delaying the programming of IP pool VRFs during VPNMgr recovery (restart and switchover) scenarios.

### Configuring Delayed VRF Programming

Use the following CLI commands to enable faster recovery of VPNmgr with VRF with IP pool configured on it in CP and UP.

```
configure
  context context_name
    ip vrf vrf_name
      ip delay-vrf-programming-during-recovery
      end
```

**NOTES:**

- By default, the keyword/feature is disabled.

- The CLI keyword is applicable to both CP and UP VRF configurations.
- Enabling the feature on non-IP pool VRFs isn't recommended.
- It's assumed that the IP pool VRF won't have any other control protocols (such as SRP) enabled, which requires TCP connections/kernel interactions.
- During the delayed interval:
  - Any functionality which requires kernel interaction for recovering VRF will not work. No subscriber data outage is expected.
  - Any configuration change related to Route/BGP/BFD/Interface/VRF fails and configuration must be reapplied.

### Change in CLI Syntax

As part of this feature, the syntax of **show ip vrf** *vrf_name_string* CLI command is changed for all platforms, including non-CUPS.

Following is the new syntax: **show ip vrf name** *vrf_name_string*

Also, all existing optional keyword after **show ip vrf** *vrf_name_string* is changed to **show ip vrf name** *vrf_name_string*. However, there's no change in output of the CLI commands.

# Configuring VRF

Follow these steps to implement VRF support for CUPS.

**At Control Plane:**

1. Associate the IP pool with VRF.
2. Create an APN to use this pool.
3. Associate UP with UP Group to ensure that the UP uses only the specific APN.

   If there are overlapping pools, ensure that you create separate APNs for each one of the pools. Also, ensure that different UPs use each of these APNs.

The following is a sample of the CP configuration:

```
context EPC2
  apn mpls1.com
    pdp-type ipv4 ipv6
    bearer-control-mode mixed
    selection-mode subscribed sent-by-ms chosen-by-sgsn
    ims-auth-service iasGx
    ip access-group css in
    ip access-group css out
    ip context-name isp
    ip address pool name PRIVATE
    ipv6 address prefix-pool PRIVATEV6
    ipv6 access-group css6 in
    ipv6 access-group css6 out
    cc-profile any prepaid-prohibited
    active-charging rulebase cisco
    user-plane-group mpls1
```

```
    exit
    apn mpls2.com
      pdp-type ipv4 ipv6
      bearer-control-mode mixed
      selection-mode subscribed sent-by-ms chosen-by-sgsn
      ims-auth-service iasGx
      ip access-group css in
      ip access-group css out
      ip context-name isp
      ip address pool name PRIVATE_1
      ipv6 address prefix-pool PRIVATEV6_1
      ipv6 access-group css6 in
      ipv6 access-group css6 out
      cc-profile any prepaid-prohibited
      active-charging rulebase cisco
      user-plane-group mpls2
    exit

config
  context isp
    ip vrf mpls-vrf-1
    ip vrf mpls-vrf-2
    #exit

    #exit
    cups enable
    ip pool PRIVATE 209.165.200.225 255.255.255.224 private 0 chunk-size 64 vrf mpls-vrf-1
   ip pool PRIVATE_1 209.165.200.225 255.255.255.224 private 0 chunk-size 64 vrf mpls-vrf-2

    ip pool STATIC 209.165.200.226 255.255.255.224 static vrf mpls-vrf-1
    ipv6 pool PRIVATEV6 prefix 8001::aaaa/54 private 0 chunk-size 64 vrf mpls-vrf-1
    ipv6 pool PRIVATEV6_1 prefix 8001::aaaa/54 private 0 chunk-size 64 vrf mpls-vrf-2
    ipv6 pool v6pool2 prefix 2a02:2121:2c4::/46 static 0 vrf mpls-vrf-1
exit

  user-plane-group mpls1
    peer-node-id ipv4-address 209.165.200.226
  #exit
  user-plane-group mpls2
    peer-node-id ipv4-address 209.165.200.228
  #exit
```

**At User Plane:**

It's recommended to configure VRF in UP before chunk is pushed from CP. Else, it leads to the failure of complete IP pool transaction (including chunks that don't belong to the VRF), and retry attempt by CP after some time.

The following is a sample of the UP configurations:

**User-Plane 1:**

```
Config
  context EPC2
    sx-service sx
      instance-type userplane
      bind ipv4-address 209.165.200.226 ipv6-address bbbb:aaaa::4
    exit
    user-plane-service up
      associate gtpu-service pgw-gtpu pgw-ingress
      associate gtpu-service sgw-ingress-gtpu sgw-ingress
      associate gtpu-service sgw-engress-gtpu sgw-egress
      associate gtpu-service saegw-sxu cp-tunnel
      associate sx-service sx
      associate fast-path service
```

```
      associate control-plane-group g1
    exit

  context isp
    ip vrf mpls-vrf-1
    #exit
    ip vrf mpls-vrf-2
    #exit
    apn mpls1.com
      pdp-type ipv4 ipv6
      bearer-control-mode mixed
      selection-mode sent-by-ms
      ip context-name isp
    exit
exit
control-plane-group g1
    peer-node-id ipv4-address 209.165.200.227
  #exit
  user-plane-group default
```

**User-Plane 2:**

```
Config
  context EPC2
    sx-service sx
      instance-type userplane
      bind ipv4-address 209.165.200.228 ipv6-address bbbb:aaaa::5
    exit
    user-plane-service up
      associate gtpu-service pgw-gtpu pgw-ingress
      associate gtpu-service sgw-ingress-gtpu sgw-ingress
      associate gtpu-service sgw-engress-gtpu sgw-egress
      associate gtpu-service saegw-sxu cp-tunnel
      associate sx-service sx
      associate fast-path service
      associate control-plane-group g1
    exit
exit

  context isp
    ip vrf mpls-vrf-1
    #exit
    ip vrf mpls-vrf-2
    #exit
    apn mpls2.com
      pdp-type ipv4 ipv6
      bearer-control-mode mixed
      selection-mode sent-by-ms
      ip context-name isp
    exit
exit

control-plane-group g1
    peer-node-id ipv4-address 209.165.200.228
  #exit
  user-plane-group default
```

# Monitoring and Troubleshooting

This section provides information regarding the CLI command available in support of monitoring and troubleshooting the feature.

# Show Command(s) and/or Outputs

This section provides information regarding show commands and/or their outputs in support of this feature.

## show ip chunks

The output of this CLI command displays all the chunks in that context.

With Overlapping Pools in Same UP functionality, VRF option is introduced in the CLI, **show ip chunks vrf** *vrf_name*, that displays only the chunks under that VRF.

- chunk-id
- chunk-size
- vrf-name
- start-addr
- end-addr
- used-addrs
- Peer Address

## show ipv6 chunks

The output of this CLI command displays all the chunks in that context.

With Overlapping Pools in Same UP functionality, VRF option is introduced in the CLI, **show ipv6 chunks vrf** *vrf_name*, that displays only the chunks under that VRF.

- chunk-id
- chunk-size
- vrf-name
- start-prefix
- end-prefix
- used-prefixes
- Peer Address