Accessing the Networking Stack

This section is applicable for Cisco IOS XR Release 7.9.1 and earlier for the following variants:

- N540-ACC-SYS
- N540X-ACC-SYS
- N540-24Z8Q2C-SYS

Accessing the Networking Stack

The Cisco IOS XR Software serves as a networking stack for communication. This section explains how applications on IOS XR can communicate with internal processes, and with servers or outside devices.

Communication Outside Cisco IOS XR

To communicate outside Cisco IOS XR, applications use the `fwdintf` interface address that maps to the `loopback0` interface or a configured Gigabit Ethernet interface address.

To have an application on IOS XR communicate with its respective server outside IOS XR, you must configure an interface address as the source address on XR. The remote servers must configure this route address to reach the respective clients on IOS XR.

This section provides an example of configuring a Gigabit Ethernet interface address as the source address for external communication.

Configure the Source Interface for External Communication

To configure a GigE interface on IOS XR for external communication, use these steps:

1. Configure a GigE interface.

```
RP/0/RP0/CPU0:ios(config)# interface GigabitEthernet 0/0/0/1
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 192.57.43.10 255.255.255.0
RP/0/RP0/CPU0:ios(config-if)# no shut
RP/0/RP0/CPU0:ios(config-if)# commit
Fri Oct 30 07:51:14.785 UTC
RP/0/RP0/CPU0:ios(config-if)# exit
RP/0/RP0/CPU0:ios(config)# exit
```

2. Verify whether the configured interface is up and operational on IOS XR.

```
RP/0/RP0/CPU0:ios# show ipv4 interface brief
Fri Oct 30 07:51:48.996 UTC

Interface                      IP-Address      Status                Protocol
Loopback0                      1.1.1.1         Up                    Up
Loopback1                      8.8.8.8         Up                    Up
GigabitEthernet0/0/0/0         192.164.168.10  Up                    Up
GigabitEthernet0/0/0/1         192.57.43.10    Up                    Up
GigabitEthernet0/0/0/2         unassigned      Shutdown              Down
MgmtEth0/RP0/CPU0/0            192.168.122.197 Up                    Up
RP/0/RP0/CPU0:ios#
```

3. Configure the GigE interface as the source address for external communication.

```
[xr-vm_node0_RP0_CPU0:~]$ exit

RP/0/RP0/CPU0:ios# config
Fri Oct 30 08:55:17.992 UTC
RP/0/RP0/CPU0:ios(config)# tpa address-family ipv4 update-source gigabitEthernet 0/0/0/1
RP/0/RP0/CPU0:ios(config)# commit
Fri Oct 30 08:55:38.795 UTC
```

Note By default, the `fwdintf` interface maps to the `loopback0` interface for external communication. This is similar to binding a routing process or router ID to the `loopback0` interface. When you use the `tpa address-family ipv4 update-source` command to bind the `fwdintf` interface to a Gigabit Ethernet interface, network connectivity can be affected if the interface goes down.

External communication is successfully enabled on IOS XR.

East-West Communication for Third-Party Applications

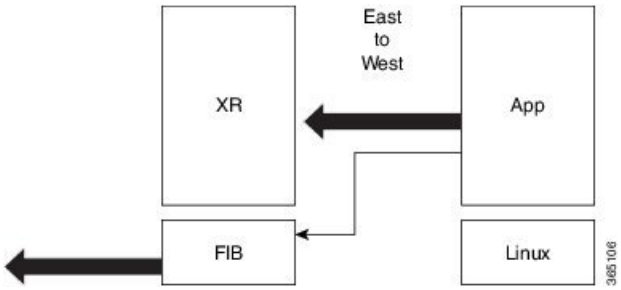
East-West communication on IOS XR is a mechanism by which applications hosted in containers interact with native XR applications (hosted in the XR control plane).

The following figure illustrates how a third-party application hosted on IOS XR interacts with the XR Control Plane.

The application sends data to the Forwarding Information Base (FIB) of IOS XR. The application is hosted in the east portion of IOS XR, while the XR control plane is located in the west region. Therefore, this form of communication between a third-party application and the XR control plane is termed as East-West (E-W) communication.

Third-party applications use this mode of communication to configure and manage containers, packages, and applications on IOS XR. In the future, this support could be extended to IOS XR, configured and managed by such third-party applications.

Figure 1: East-West Communication on IOS XR



For a third-party application to communicate with IOS XR, the Loopback1 interface must be configured. This is explained in the following procedure.

1. Configure the Loopback1 interface on IOS XR.

```
RP/0/RP0/CPU0:ios(config)# interface Loopback1
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 8.8.8.8/32
RP/0/RP0/CPU0:ios(config-if)# no shut
RP/0/RP0/CPU0:ios(config-if)# commit
RP/0/RP0/CPU0:ios(config-if)# exit
```

2. Configure another Loopback interface that will be the East interface. You must configure this loopback interface to act as the TPA-facing interface, and Cisco IOS XR interacts with the TPA using this interface.

```
RP/0/RP0/CPU0:ios(config)# interface Loopback100
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 15.1.1.1/32
RP/0/RP0/CPU0:ios(config-if)# no shut
RP/0/RP0/CPU0:ios(config-if)# commit
RP/0/RP0/CPU0:ios(config-if)# exit
```

3. Verify the creation of the Loopback1 (West) and Loopback100 (East) interfaces.

```
RP/0/RP0/CPU0:ios# show ipv4 interface brief
Thu Nov 12 10:01:00.874 UTC

Interface                  IP-Address      Status                Protocol
Loopback0                  1.1.1.1         Up                    Up
Loopback1                  8.8.8.8         Up                    Up
Loopback100                15.1.1.1        Up                    Up
GigabitEthernet0/0/0/0     192.164.168.10  Up                    Up
GigabitEthernet0/0/0/1     192.57.43.10    Up                    Up
GigabitEthernet0/0/0/2     unassigned      Shutdown              Down
MgmtEth0/RP0/CPU0/0        192.168.122.197 Up                    Up
```

4. Configure the TPAs.

```
RP/0/RP0/CPU0:ios(config)#tpa vrf default east-west loopback 1
RP/0/RP0/CPU0:ios(config)#tpa vrf default address-family ipv4 default-route mgmt
RP/0/RP0/CPU0:ios(config)#tpa vrf default address-family ipv4 update-source dataports
loopback 100
RP/0/RP0/CPU0:ios(config)#commit
```

5. Verify that a TPA interface that sets up Loopback100 as the East interface is configured.

```
RP/0/RP0/CPU0:ios#sh run tpa
Mon Jun 7 07:22:08.324 UTC
tpa
vrf default
east-west Loopback1
address-family ipv4
default-route mgmt
```

```
update-source dataports Loopback100
!
!
!
```

6. Verify the E-W communication configuration by logging into the container and checking the routes. You can also ping the router-side East interface.

Note You can use the bash command to connect to the router and execute commands only in the testing environment.

```
RP/0/RP0/CPU0:ios#bash
Mon Jun 7 07:22:57.650 UTC
[ios:~]$ docker exec -it a0 bash
root@host:0_RP0:/# ip route
default dev fwd_ew scope link src 15.1.1.1
8.8.8.8 dev fwd_ew scope link src 15.1.1.1
192.168.104.0/24 dev Mg0_RP0_CPU0_0 scope link src 192.168.104.10
root@host:0_RP0:/# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=255 time=0.397 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=255 time=0.535 ms
^C
--- 8.8.8.8 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 2ms
rtt min/avg/max/mdev = 0.397/0.466/0.535/0.069 ms
root@host:0_RP0:/#
```

For more information on how to launch your own containers, see Using Docker for Hosting Applications on Cisco IOS XR topic. .

Configuring Multiple VRFs for Application Hosting

Cisco NCS 540 routers support the configuration of multiple VRFs. The applications hosted in third-party containers can communicate with VRFs configured on XR, after east-west communication has been enabled on the VRFs.

This section describes the configuration for creating mulitple VRFs, and enabling east-west communication between the applications and the VRFs.

Configuration Procedure

Use the following steps to configure multiple VRFs for use on Cisco IOS XR.

1. Configure VRFs on XR.

```
RP/0/RP0/CPU0:ios(config)# vrf purple
RP/0/RP0/CPU0:ios(config-vrf)# address-family ipv4
RP/0/RP0/CPU0:ios(config-vrf)# address-family ipv6
RP/0/RP0/CPU0:ios(config-vrf)# exit

RP/0/RP0/CPU0:ios(config)# vrf green
RP/0/RP0/CPU0:ios(config-vrf)# address-family ipv4
RP/0/RP0/CPU0:ios(config-vrf)# address-family ipv6
RP/0/RP0/CPU0:ios(config-vrf)# exit
```

```
RP/0/RP0/CPU0:ios(config)# telnet vrf purple ipv4 server max-servers 2
RP/0/RP0/CPU0:ios(config)# telnet vrf purple ipv6 server max-servers 2
RP/0/RP0/CPU0:ios(config)# telnet vrf green ipv4 server max-servers 2
RP/0/RP0/CPU0:ios(config)# telnet vrf green ipv6 server max-servers 2
RP/0/RP0/CPU0:ios(config)# telnet ipv4 server max-servers 2
RP/0/RP0/CPU0:ios(config)# telnet ipv6 server max-servers 2
```

2. Configure the interfaces to be used with the VRFs.

```
RP/0/RP0/CPU0:ios(config)# interface loopback1
RP/0/RP0/CPU0:ios(config-if)# vrf purple
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 1.1.1.1 255.255.255.0
RP/0/RP0/CPU0:ios(config-if)# ipv6 address 10::1/64
RP/0/RP0/CPU0:ios(config-if)# exit

RP/0/RP0/CPU0:ios(config)# interface loopback2
RP/0/RP0/CPU0:ios(config-if)# vrf purple
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 2.2.2.2 255.255.255.0
RP/0/RP0/CPU0:ios(config-if)# ipv6 address 20::1/64
RP/0/RP0/CPU0:ios(config-if)# exit

RP/0/RP0/CPU0:ios(config)# interface loopback3
RP/0/RP0/CPU0:ios(config-if)#vrf green
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 3.3.3.3 255.255.255.0
RP/0/RP0/CPU0:ios(config-if)# ipv6 address 30::1/64
RP/0/RP0/CPU0:ios(config-if)# exit

RP/0/RP0/CPU0:ios(config)# interface loopback4
RP/0/RP0/CPU0:ios(config-if)# vrf green
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 4.4.4.4 255.255.255.0
RP/0/RP0/CPU0:ios(config-if)# ipv6 address 40::1/64
RP/0/RP0/CPU0:ios(config-if)# exit

RP/0/RP0/CPU0:ios(config)# interface mgmtEth 0/RP0/CPU0/0
RP/0/RP0/CPU0:ios(config-if)# vrf purple
RP/0/RP0/CPU0:ios(config-if)# ipv4 address dhcp
RP/0/RP0/CPU0:ios(config-if)# exit

RP/0/RP0/CPU0:ios(config)# interface GigabitEthernet 0/0/0/0
RP/0/RP0/CPU0:ios(config-if)# vrf purple
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 10.20.30.40 255.255.255.0
RP/0/RP0/CPU0:ios(config-if)# ipv6 address 24::1/64
RP/0/RP0/CPU0:ios(config-if)# exit

RP/0/RP0/CPU0:ios(config)# interface gigabitEthernet 0/0/0/1
RP/0/RP0/CPU0:ios(config-if)# vrf green
RP/0/RP0/CPU0:ios(config-if)# ipv4 address 40.30.20.10 255.255.255.0
RP/0/RP0/CPU0:ios(config-if)# ipv6 address 22::1/64
RP/0/RP0/CPU0:ios(config-if)# exit
RP/0/RP0/CPU0:ios(config)# commit
Fri Sep  1 12:04:37.796 UTC
```

3. Configure TPA VRFs.

```
RP/0/RP0/CPU0:ios(config)# tpa
RP/0/RP0/CPU0:ios(config-tpa)# vrf purple
RP/0/RP0/CPU0:ios(config-tpa-vrf)# east-west loopback1
RP/0/RP0/CPU0:ios(config-tpa-vrf)# east-west loopback2
RP/0/RP0/CPU0:ios(config-tpa-vrf)# address-family ipv4
RP/0/RP0/CPU0:ios(config-tpa-vrf-afi)# update-source GigabitEthernet 0/0/0/0
RP/0/RP0/CPU0:ios(config-tpa-vrf-afi)# exit
```

```
RP/0/RP0/CPU0:ios(config-tpa-vrf)# address-family ipv6
RP/0/RP0/CPU0:ios(config-tpa-vrf-afi)# update-source Gigabitethernet 0/0/0/0
RP/0/RP0/CPU0:ios(config-tpa-vrf-afi)# exit
RP/0/RP0/CPU0:ios(config-tpa-vrf)# exit


RP/0/RP0/CPU0:ios(config-tpa)# vrf green
RP/0/RP0/CPU0:ios(config-tpa-vrf)# east-west loopback3
RP/0/RP0/CPU0:ios(config-tpa-vrf)# east-west loopback4
RP/0/RP0/CPU0:ios(config-tpa-vrf)# address-family ipv4
RP/0/RP0/CPU0:ios(config-tpa-vrf-afi)# update-source GigabitEthernet 0/0/0/1
RP/0/RP0/CPU0:ios(config-tpa-vrf-afi)# exit
RP/0/RP0/CPU0:ios(config-tpa-vrf)# address-family ipv6
RP/0/RP0/CPU0:ios(config-tpa-vrf-afi)# update-source Gigabitethernet 0/0/0/1
RP/0/RP0/CPU0:ios(config-tpa-vrf-afi)# exit
RP/0/RP0/CPU0:ios(config-tpa-vrf)# exit
RP/0/RP0/CPU0:ios(config-tpa)# exit
```

4. Validate the configuration.

```
RP/0/RP0/CPU0:ios(config)# show run
Fri Sep  1 12:06:35.596 UTC
...
vrf purple
address-family ipv4
address-family ipv6
vrf green
address-family ipv4
address-family ipv6

telnet vrf green ipv4 server max-servers 2
telnet vrf green ipv6 server max-servers 2
telnet vrf purple ipv4 server max-servers 2
telnet vrf purple ipv6 server max-servers 2
telnet vrf default ipv4 server max-servers 2
telnet vrf default ipv6 server max-servers 2
...
!
tpa
 vrf purple
  east-west loopback1
  east-west loopback2
  address-family ipv4
   update-source GigabitEthernet0/0/0/0
 !
  address-family ipv6
   update-source GigabitEthernet0/0/0/0
 !

 vrf green
  east-west loopback3
  east-west loopback4
  address-family ipv4
   update-source GigabitEthernet0/0/0/1
 !
  address-family ipv6
   update-source GigabitEthernet0/0/0/1
 !
!
interface loopback1
 vrf purple
 ipv4 address 1.1.1.1 255.255.255.0
 ipv6 address 10::1/64
!
```

```
interface loopback2
 vrf purple
 ipv4 address 2.2.2.2 255.255.255.0
 ipv6 address 20::1/64
!
interface loopback3
 vrf green
 ipv4 address 3.3.3.3 255.255.255.0
 ipv6 address 30::1/64
!
interface loopback4
 vrf green
 ipv4 address 4.4.4.4 255.255.255.0
 ipv6 address 40::1/64
!
interface MgmtEth0/RP0/CPU0/0
 vrf purple
 ipv4 address dhcp
!
router static
 address-family ipv4 unicast
  0.0.0.0/0 MgmtEth0/RP0/CPU0/0 10.0.2.2
 !
!
```

You have successfully configured multiple VRFs for use on Cisco IOS XR.

Accessing the Networking Stack