# Process Placement on Cisco IOS XR Software

This module describes conceptual information and configuration tasks for process placement on a Cisco CRS-1 router.

The Cisco IOS XR Process Placement feature balances application processes between the available Route Processors (RPs) and distributed Route Processors (DRPs) on a Cisco CRS-1 system based on memory usage and other criteria. Use the procedures described in this document to reoptimize the placement of processes, or override the default placement policies.

**Note** For complete descriptions of the commands listed in this module, see the "Related Documents" section on page 99. To locate documentation for other commands that might appear in the course of performing a configuration task, search online in the Cisco IOS XR software master command index for Release 3.5.0.

**Feature History for Configuring Cisco IOS XR Process Placement**

| Release | Modification |
|---|---|
| Release 3.3.0 | The feature was introduced on the Cisco CRS-1. |
| Release 3.4.0 | No modification. |
| Release 3.5.0 | No modification. |

# Contents

# Prerequisites for Configuring Cisco IOS XR Process Placement

Before configuring process placement on your Cisco CRS-1 system, you must be in a user group associated with a task group that includes the proper task IDs for process placement commands. For detailed information about user groups and task IDs, see the *Configuring AAA Services on Cisco IOS XR Software* module of the *Cisco IOS XR System Security Configuration Guide*.

# Restrictions for Cisco IOS XR Process Placement

Only processes that are identified in the Cisco IOS XR software as placeable can be controlled through process placement configuration. Nonplaceable processes are not affected by placement policy. To learn the processes that are placeable, issue the **show placement program all** command.

# Information About Cisco IOS XR Process Placement

To configure process placement policies, you need to understand the following concepts:



## What Is a Process?

To achieve high availability and performance, the Cisco IOS XR software is built on a modular system of processes. Each process provides specific functionality for the system and runs in a protected memory space to ensure that problems with one process cannot impact the entire system. Multiple instances of a process can run on a single node, and multiple threads of execution can run on each process instance.

Under normal operating conditions, processes are managed automatically by the Cisco IOS XR software. Processes are started, stopped, or restarted as required by the running configuration of the router. In addition, processes are checkpointed to optimize performance during process restart and automatic switchover.

# What Is Process Placement?

Process placement is the assignment of placeable processes to specific locations, such as a DRP installed in a Cisco CRS-1 router. The process placement is configured and managed for each Secure Domain Router (SDR) in the system.

Placeable processes for Release 3.3.0 are identified as all routing processes, such as Open Shortest Path First Protocol (OSPF), Border Gateway Protocol (BGP), and multicast routing.

# Default Placement Policy

In a new system, processes are distributed evenly among the available RP and DRP nodes and node pairs in a SDR.

**Note** The default process policy that is shipped on the system upon startup is suitable for general purposes. While customizing is possible, there is no requirement to change the process placement. If you believe the a change is required, you should work closely with Cisco personnel to ensure that the impact to your system is contained to just an instance of a process to avoid any undesirable results.

The default placement policy is specified in this way:

- All processes are assumed to use one megabyte of memory.
- The location type for process placement is a node pair (active and standby nodes).
- The process does not move automatically (**current** = 100) unless an individual node fails and the process needs to be started on a different node.
- When a new node pair is added, the following rules apply:
  - The currently running processes are not automatically moved to the new cards.
  - The general preference is for new processes (such as a new ISIS instance) to start on the new node pair, which contains the most available CPU and memory resources in the system.
  - Other affinity settings may override the general preference. For example, if the IS-IS process has a strong affinity to run on the same node where ipv4_io is running, then IS-IS would be started on that node, and not the new node-pair.

# Reasons to Change the Default Process Placement

Although the default process policy that is shipped on the system upon startup is suitable for general purposes, changes to the router configuration can result in the need for processes to be rebalanced among the available CPU and memory resources.

When a system is initially booted, the system assumes that all processes use the same amount of memory, thereby treating each process as equivalent. As the configuration grows, however, the CPU load and memory requirements of some application processes increase. Centralized applications may need a larger portion of the RP and DRP resources, or distributed applications may require additional instances of processes to be started on new DRPs.

In addition, when a new RP or DRP is added to a system, only new processes or process or process instances are added to the node. This could result in some processes with too few resources, while the newer RP and DRP cards are under-utilized.

Therefore, as the software configuration changes, or hardware is added, it may become necessary to rebalance processes among the available RPs and DRPs in an SDR.

# Reoptimizing Process Placements

The easiest and most reliable method for users to redistribute processes among the available RPs and DRPs in an SDR is with the command **placement reoptimize**.

During router operation, the actual resource usage of each process is collected and compared to the router configuration and network topology. An ideal configuration for process placement is created and updated in real time.

To implement this ideal process placement configuration, enter the command **placement reoptimize** in EXEC mode. Before the changes are made, the system displays a summary of the predicted changes. You can either accept the changes or cancel the operation.

See the "Reomptimizing Process Placement" section on page 88 for detailed instructions.

# Reconfiguring Process Placements

You can also change the process placement *affinities*, or preferences, to override the default policies. For example, you may learn that some placeable processes may perform better on the DSDRSC of a node pair (active RP and standby) to fulfill high availability obligations or that some processes may do better on the standby RP. Other processes might benefit from co-location or by being assigned to nodes far apart from each other.

**Note** Consult with your technical support representative before changing the default process placement configuration. Incorrect configurations can cause system error, poor performance or downtime.



## Recommended Guidelines for Process Placement

The following are a few recommended guidelines for changes to the process placement configuration:

- Generally, the process placement feature functions well upon system startup so fine tuning is seldom required.
- Use the EXEC mode command **placement reoptimize**, as described in the "Reoptimizing Process Placements" section on page 86 to automatically redistribute the processes among the available RPs and DRPs.
- Keep process placement updates to a minimum and always consult technical support personnel before implementation.
- Ensure that affinity values are below 100 points.
- Location set affinities are not recommended because the command hardcodes placement to a node or node pair and disallows automatic process placement.

## Process Placement Based on Memory Consumption

You can change process placements based on memory use of processes in an SDR. Memory use is expressed in terms of the memory "footprint" of the placeable process. The system attempts to spread the load among the nodes without exceeding their memory capacity. In addition, the system computes the affinity values to determine the best placement.

For Release 3.3.0, the software assumes that every placeable process uses one megabyte of memory.

For detailed instructions, see the "Setting Memory Consumption Thresholds" section on page 90.

## Changing Process Affinities

Process placement can also be controlled by changing the *affinities*, or preferences, of a process or process group. The following types of process affinities are operator configurable:

- affinity location set
- affinity location type
- affinity program
- affinity self

### affinity location set

This affinity assigns a process to run on a specific node pair. A node pair is either an active and standby pair of nodes [hosted on route processors (RPs) or distributed RPs], or a single active node on an RP or DRP that does not have a standby.

This affinity overrides the placement process logic for determining optimal placement for processes. It forces a process to remain in or away from a location on the router regardless of what might occur in the system. This command also makes the configuration more specific to a router and less general.

### affinity location type

This affinity assigns a process to a particular location. The default policy is that the location type be a node pair (active and standby nodes), and that the process does not move automatically (**current** = 100) unless a solo node fails and the process needs to be started on a different node.

You can configure the placement policy to allow certain processes to stay where they are (**current**) or move just by indicating so through the various affinity choices. The higher the positive value of an affinity, the stronger the requirement that the process run at a location, and so on. A low or zero point value indicates a weaker requirement (or no preference) that a process run at a location.

### affinity program

This affinity collocates processes or keeps them apart. You would want to use this affinity because you have learned that certain processes perform better when they are running together on the same node (**attract**); or on different nodes, apart from each other (**repulse**).

### affinity self

This affinity adjusts placement decisions when multiple instances of a process are started. An attract (positive) affinity indicates a preference to have all instances of a process run on the same node, while a repulse (negative) affinity indicates a preference to have each instance of a process run on different nodes.

## Hierarchical Placement Policy

When you configure placement policies, you must remember that affinities are applied to the software in a hierarchical way.

Affinities applied to process instances take precedence over affinities applied to a process class. In the following example, all OSPF instances are placed on the primary RP, but OSPF instance 10 are placed on a node pair:

```
RP/0/RP0/CPU0:router(config)# placement program ospf
RP/0/RP0/CPU0:router(config-place)# affinity location-type primary attract 200

RP/0/RP0/CPU0:router(config)# placement program ospf instance 10
RP/0/RP0/CPU0:router(config-place)# affinity location-type paired attract 200
```

Class affinities take precedence over default process affinities. In the following example, all OSPF instances are placed on a single node. This placement overrides the default policy that places processes on node pairs.

```
RP/0/RP0/CPU0:router(config)# placement program ospf
RP/0/RP0/CPU0:router(config-place)# affinity location-type paired repulse 200
```

# How to Configure Cisco IOS XR Process Placement

This section contains instructions for the following tasks:

- Reomptimizing Process Placement, page 88 (required)
- Setting Memory Consumption Thresholds, page 90 (required)
- Creating a Location Set Affinity, page 91 (required)
- Creating a Location Type Affinity, page 93 (required)
- Creating a Program Affinity, page 94 (required)
- Creating a Self Affinity, page 96 (required

# Reomptimizing Process Placement

This task reoptimizes the placeable processes among the available RP and DRP nodes according to memory and CPU usage.

### SUMMARY STEPS

1. **placement reoptimize**
2. **yes**
   or
   **no**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **placement reoptimize**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# placement reoptimize` | Displays the predicted changes of the optimization. |
| **Step 2** | **yes**<br>or<br>**no**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# yes` | Accepts or rejects the changes. |

# Setting Memory Consumption Thresholds

## SUMMARY STEPS

1. **show placement policy global**
2. **configure**
3. **placement memory** {{**maximum** | **threshold**} *value*}
4. **end**
   or
   **commit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **show placement policy global**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show placement policy global` | Displays the current memory settings. |
| **Step 2** | **configure**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 3** | **placement memory** {{**maximum** \| **threshold**} *value*}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# placement memory maximum 80` | Use the **placement memory** command with the **maximum** *value* keyword and argument to set the maximum percentage of memory that can be used on a node (based on the estimated memory usage of the processes).<br><br>Use the **placement memory** command with the **threshold** *value* keyword and argument to define the memory load level to trigger migration. The system attempts to balance all nodes at or below the threshold memory percentage. In other words, the system does not place a process on a node that has exceeded the threshold value, unless all other nodes have also reached their thresholds (or unless some other large affinity overrides this consideration). |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-place)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-place)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

# Creating a Location Set Affinity

This task sets the affinity of a placement program (process) to or from node pairs.

## SUMMARY STEPS

1. **configure**
2. **placement program** *program* {**instance** *instance* | **default**}
3. **affinity location-set** *node-id1* [*node-id2*] {**attract** *strength* | **repulse** *strength* | **default** | **none**}
4. **end**
   or
   **commit**
5. **show placement location** {*node-id* | **all**}
6. **show placement program** {*program* | **all**}

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **placement program** *program* {**instance** *instance* \| **default**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# placement program ospf` | Enters placement program configuration mode. |
| **Step 3** | **affinity location-set** *node-id1* [*node-id2*] {**attract** *strength* \| **repulse** *strength* \| **default** \| **none**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-place)# affinity location-set 0/1/cpu0 0/1/cpu1 attract 200` | Sets the affinity of a placement program (process) to or from node pairs.<br>To specify multiple nodes, enter the value of the *node-id* argument for each node. You can specify up to 5 nodes. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-place)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-place)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **show placement location** {*node-id* \| **all**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show placement location all` | Displays the location of a placement process. |
| **Step 6** | **show placement program** {*program* \| **all**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show placement program ospf` | Displays the operational state for each placement program. |

# Creating a Location Type Affinity

This task sets affinity of a placement program (process) to or from a location type.

## SUMMARY STEPS

1. **configure**
2. **placement program** *program* {**instance** *instance* | **default**}
3. **affinity location-type** {**current** | **paired** | **primary**} {**attract** *strength* | **repulse** *strength* | **default** | **none**}
4. **end**
   or
   **commit**
5. **show placement location** {*node-id* | **all**}
6. **show placement program** {*program* | **all**}

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **placement program** *program* {**instance** *instance* \| **default**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# placement program bgp` | Enters placement program configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **affinity location-type** {**current** \| **paired** \| **primary**} {**attract** *strength* \| **repulse** *strength* \| **default** \| **none**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-place)# affinity location-type current attract 10` | Sets the affinity of a placement program (process) to or from a location type.<br>• This example shows how to place Border Gateway Protocol (BGP) in the most optimal location at run time when load balancing is required. BGP will not be tied to a node pair but move when necessary. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-place)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-place)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 5** | **show placement location** {*node-id* \| **all**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show placement location all` | Displays the location of a placement process. |
| **Step 6** | **show placement program** {*program* \| **all**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show placement program bgp` | Displays the operational state for each placement program. |

# Creating a Program Affinity

This task sets the affinity of a placement program (process) to or from another program.

## SUMMARY STEPS

1. **configure**
2. **placement program** *program* {**instance** *instance* | **default**}
3. **affinity program** *program* {**attract** *strength* | **repulse** *strength* | **default** | **none**}

4. **end**
   or
   **commit**
5. **show placement location** {*node-id* | **all**}
6. **show placement program** {*program* | **all**}

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **placement program** *program* {**instance** *instance* \| **default**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# placement program ipv4_rib` | Enters placement program configuration mode. |
| **Step 3** | **affinity program** *program* {**attract** *strength* \| **repulse** *strength* \| **default** \| **none**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-place)# affinity program ipv6_rib repulse 200` | Sets the affinity of a placement program (process) to or from another program.<br>• This example shows how to keep IPv4 and IPv6 Routing Information Bases (RIBs) apart. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-place)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-place)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **show placement location** {*node-id* \| **all**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show placement location all` | Displays the location of a placement process. |
| **Step 6** | **show placement program** {*program* \| **all**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show placement program all` | Displays the operational state for each placement program. |

# Creating a Self Affinity

This task sets the affinity of a placement program (process) to or from one of its own instances.

## SUMMARY STEPS

1. **configure**
2. **placement program** *program* {**instance** *instance* | **default**}
3. **affinity self** {**attract** *strength* | **repulse** *strength* | **default** | **none**}
4. **end**
   or
   **commit**
5. **show placement location** {*node-id* | **all**}
6. **show placement program** {*program* | **all**}

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters global configuration mode. |
| **Step 2** | **placement program** *program* {**instance** *instance* \| **default**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# placement program bgp` | Enters placement program configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **affinity self** {**attract** *strength*\| **repulse** *strength* \| **default** \| **none**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-place)# affinity self repulse 200` | Sets the affinity of a placement program (process) to or from one of its own instances. |
| **Step 4** | **end**<br>or<br>**commit**<br><br>**Example:**<br>`RP/0/RP0/CPU0:router(config-place)# end`<br>or<br>`RP/0/RP0/CPU0:router(config-place)# commit` | Saves configuration changes.<br>• When you issue the **end** command, the system prompts you to commit changes:<br>`Uncommitted changes found, commit them before exiting (yes/no/cancel)? [cancel]:`<br>– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.<br>– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.<br>– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.<br>• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session. |
| **Step 5** | **show placement location** {*node-id* \| **all**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show placement location all` | Displays the location of a placement process. |
| **Step 6** | **show placement program** {*program* \| **all**}<br><br>**Example:**<br>`RP/0/RP0/CPU0:router# show placement program bgp` | Displays the operational state for each placement program. |

# Configuration Examples

This section contains examples to view the processes that are placeable in an SDR in a Cisco CRS-1 router.

If you believe that a custom reconfiguration of the processes on your system is required, you should work closely with Cisco personnel to ensure that the impact to your system is contained to just an instance of a process to avoid any undesirable results.

To learn the processes that are placeable, enter the **show placement program all** command in EXEC mode.

```
RP/0/RP0/CPU0:router# show placement program all

If a program is shown as having 'rejected locations' (i.e., locations on which
it cannot be placed), the locations in question can been seen using the "show
placement policy program" command.

If a program has been placed but not yet started, the amount of time elapsed
since the program was placed is shown in the 'waiting to start' field.

Parentheses around the node indicate that the node has not yet fully booted.
  This will be true of standby nodes.

Program                      Placed at location            # rejected  Waiting
                                                           locations   to start
-------------------------------------------------------------------------------
ipv4_static                  0/RP0/CPU0 (0/RP1/CPU0)
mpls_vpn_mib                 0/RP0/CPU0 (0/RP1/CPU0)
rt_check_mgr                 0/RP0/CPU0 (0/RP1/CPU0)
mpls_rid_helper              0/RP0/CPU0 (0/RP1/CPU0)
ital_test2                   0/RP0/CPU0 (0/RP1/CPU0)
ital_test1                   0/RP0/CPU0 (0/RP1/CPU0)
fm_server                    0/RP0/CPU0 (0/RP1/CPU0)
fm_script_dir                0/RP0/CPU0 (0/RP1/CPU0)
fm_metric_dir                0/RP0/CPU0 (0/RP1/CPU0)
fm_fd_stats                  0/RP0/CPU0 (0/RP1/CPU0)
fm_fd_hardware               0/RP0/CPU0 (0/RP1/CPU0)
fm_fd_drvinfra               0/RP0/CPU0 (0/RP1/CPU0)
fm_fd_counter                0/RP0/CPU0 (0/RP1/CPU0)
ipv6_rump                    0/RP0/CPU0 (0/RP1/CPU0)
ipv6_local                   0/RP0/CPU0 (0/RP1/CPU0)
ipv6_connected               0/RP0/CPU0 (0/RP1/CPU0)
ipv4_rump                    0/RP0/CPU0 (0/RP1/CPU0)
ipv4_local                   0/RP0/CPU0 (0/RP1/CPU0)
ipv4_connected               0/RP0/CPU0 (0/RP1/CPU0)
tftp_fs                      0/RP0/CPU0 (0/RP1/CPU0)
rcp_fs                       0/RP0/CPU0 (0/RP1/CPU0)
ftp_fs                       0/RP0/CPU0 (0/RP1/CPU0)
domain_services              0/RP0/CPU0 (0/RP1/CPU0)
bfd                          0/RP0/CPU0 (0/RP1/CPU0)
ipv6_mpa                     0/RP0/CPU0 (0/RP1/CPU0)
ipv4_mpa                     0/RP0/CPU0 (0/RP1/CPU0)
ipv6_arm                     0/RP0/CPU0 (0/RP1/CPU0)
ipv4_arm                     0/RP0/CPU0 (0/RP1/CPU0)
policy_repository            0/RP0/CPU0 (0/RP1/CPU0)
ipv6_rib                     0/RP0/CPU0 (0/RP1/CPU0)
ipv4_rib                     0/RP0/CPU0 (0/RP1/CPU0)
statsd_manager               0/RP0/CPU0 (0/RP1/CPU0)
rsi_master                   0/RP0/CPU0 (0/RP1/CPU0)
```

# Additional References

The following sections provide references related to Cisco IOS XR Process Placement.

## Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS XR process placement commands | *Process and Memory Management Commands on Cisco IOS XR Software* module of *Cisco IOS XR System Management Command Reference*, Release 3.5 |
| Cisco IOS XR master command index | *Cisco IOS XR Commands Master List*, Release 3.5 |
| Cisco IOS XR getting started material | *Cisco IOS XR Getting Started Guide*, Release 3.5 |
| Information about user groups and task IDs | *Configuring AAA Services on Cisco IOS XR Software* module of *Cisco IOS XR System Security Configuration Guide*, Release 3.5 |

## Standards

| Standards | Title |
|---|---|
| No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature. | — |

## MIBs

| MIBs | MIBs Link |
|---|---|
| — | To locate and download MIBs using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL and choose a platform under the Cisco Access Products menu: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml |

## RFCs

| RFCs | Title |
|---|---|
| No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature. | — |

# Technical Assistance

| Description | Link |
|---|---|
| The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content. | http://www.cisco.com/techsupport |