IP Application Services Configuration Guide, Cisco IOS Release 15SY

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

CONTENTS

CHAPTER 10

CHAPTER 11

CHAPTER 1

Configuring Enhanced Object Tracking

Before the introduction of the Enhanced Object Tracking feature, the Hot Standby Router Protocol (HSRP) had a simple tracking mechanism that allowed you to track the interface line-protocol state only. If the line-protocol state of the interface went down, the HSRP priority of the router was reduced, allowing another HSRP router with a higher priority to become active.

The Enhanced Object Tracking feature separates the tracking mechanism from HSRP and creates a separate standalone tracking process that can be used by other processes and HSRP. This feature allows tracking of other objects in addition to the interface line-protocol state.

A client process such as HSRP, Virtual Router Redundancy Protocol (VRRP), or Gateway Load Balancing Protocol (GLBP), can register its interest in tracking objects and then be notified when the tracked object changes state.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Enhanced Object Tracking

Enhanced Object Tracking is not stateful switchover (SSO)-aware and cannot be used with Hot Standby Routing Protocol (HSRP), Virtual Router Redundancy Protocol (VRRP), or Gateway Load Balancing Protocol (GLBP) in SSO mode.

Information About Enhanced Object Tracking

Feature Design of Enhanced Object Tracking

The Enhanced Object Tracking feature provides complete separation between the objects to be tracked and the action to be taken by a client when a tracked object changes. Thus, several clients such as HSRP, VRRP, or GLPB can register their interest with the tracking process, track the same object, and each take different action when the object changes.

Each tracked object is identified by a unique number that is specified on the tracking CLI. Client processes use this number to track a specific object.

The tracking process periodically polls the tracked objects and notes any change of value. The changes in the tracked object are communicated to interested client processes, either immediately or after a specified delay. The object values are reported as either up or down.

You can configure a combination of tracked objects in a list and a flexible method for combining objects using Boolean logic. This functionality includes the following capabilities:

- Threshold—The tracked list can be configured to use a weight or percentage threshold to measure the state of the list. Each object in a tracked list can be assigned a threshold weight. The state of the tracked list is determined by whether the threshold has been met.
- Boolean "and" function—When a tracked list has been assigned a Boolean "and" function, each object defined within a subset must be in an up state so that the tracked object can become up.
- Boolean "or" function—When the tracked list has been assigned a Boolean "or" function, at least one object defined within a subset must be in an up state so that the tracked object can become up.

With CSCtg75700, a maximum of 1000 objects can be tracked. Although 1000 tracked objects can be configured, each tracked object uses CPU resources. The amount of available CPU resources on a router depends on variables such as traffic load and how other protocols are configured and run. The ability to use 1000 tracked objects depends on the available CPU. Testing should be conducted on site to ensure that the service works under the specific site traffic conditions.

Interface State Tracking

An IP-routing object is considered up when the following criteria exist:

- IP routing is enabled and active on the interface.
- The interface line-protocol state is up.

- The interface IP address is known. The IP address is configured or received through Dynamic Host Configuration Protocol (DHCP) or IP Control Protocol (IPCP) negotiation.

Interface IP routing will go down when one of the following criteria exists:

- IP routing is disabled globally.
- The interface line-protocol state is down.
- The interface IP address is unknown. The IP address is not configured or received through DHCP or IPCP negotiation.

Tracking the IP-routing state of an interface using the **track interface ip routing** command can be more useful in some situations than just tracking the line-protocol state using the **track interface line-protocol** command, especially on interfaces for which IP addresses are negotiated. For example, on a serial interface that uses the PPP, the line protocol could be up (link control protocol [LCP] negotiated successfully), but IP could be down (IPCP negotiation failed).

The **track interface ip routing** command supports the tracking of an interface with an IP address acquired through any of the following methods:

- Conventional IP address configuration
- PPP/IPCP
- DHCP
- Unnumbered interface

You can configure Enhanced Object Tracking to consider the carrier-delay timer when tracking the IP-routing state of an interface by using the **carrier-delay** command in tracking configuration mode.

Scaled Route Metrics

The **track ip route** command enables tracking of a route in the routing table. If a route exists in the table, the metric value is converted into a number. To provide a common interface to tracking clients, normalize route metric values to the range from 0 to 255, where 0 is connected and 255 is inaccessible. Scaled metrics can be tracked by setting thresholds. Up and down state notification occurs when the thresholds are crossed. The resulting value is compared against threshold values to determine the tracking state as follows:

- State is up if the scaled metric for that route is less than or equal to the up threshold.
- State is down if the scaled metric for that route is greater than or equal to the down threshold.

Tracking uses a per-protocol configurable resolution value to convert the real metric to the scaled metric. The table below shows the default values used for the conversion. You can use the **track resolution** command to change the metric resolution default values.

Table 1: Metric Conversion

Route Type[1]	Metric Resolution
Static	10
Enhanced Interior Gateway Routing Protocol (EIGRP)	2560

Route Type[1]	Metric Resolution
Open Shortest Path First (OSPF)	1
Intermediate System-to-Intermediate System (IS-IS)	10

[1] RIP is scaled directly to the range from 0 to 255 because its maximum metric is less than 255.

For example, a change in 10 in an IS-IS metric results in a change of 1 in the scaled metric. The default resolutions are designed so that approximately one 2-Mbps link in the path will give a scaled metric of 255.

Scaling the very large metric ranges of EIGRP and IS-IS to a 0 to 255 range is a compromise. The default resolutions will cause the scaled metric to exceed the maximum limit with a 2-Mb/s link. However, this scaling allows a distinction between a route consisting of three Fast-Ethernet links and a route consisting of four Fast-Ethernet links.

IP SLA Operation Tracking

Object tracking of IP Service Level Agreements (SLAs) operations allows tracking clients to track the output from IP SLAs objects and use the provided information to trigger an action.

Cisco IOS IP SLAs is a network performance measurement and diagnostics tool that uses active monitoring. Active monitoring is the generation of traffic in a reliable and predictable manner to measure network performance. software uses IP SLAs to collect real-time metrics such as response time, network resource availability, application performance, jitter (interpacket delay variance), connect time, throughput, and packet loss.

These metrics can be used for troubleshooting, for proactive analysis before problems occur, and for designing network topologies.

Every IP SLAs operation maintains an operation return-code value. This return code is interpreted by the tracking process. The return code can return OK, OverThreshold, and several other return codes. Different operations can have different return-code values, so only values common to all operation types are used.

Two aspects of an IP SLAs operation can be tracked: state and reachability. The difference between these aspects is the acceptance of the OverThreshold return code. The table below shows the state and reachability aspects of IP SLAs operations that can be tracked.

Table 2: Comparison of State and Reachability Operations

Tracking	Return Code	Track State
State	OK (all other return codes)	Up Down
Reachability	OK or OverThreshold (all other return codes)	Up Down

Enhanced Object Tracking and Embedded Event Manager

Enhanced Object Tracking (EOT) is now integrated with Embedded Event Manager (EEM) to allow EEM to report on status change of a tracked object and to allow EOT to track EEM objects. A new type of tracking object--a stub object--is created. The stub object can be modified by an external process through a defined Application Programming Interface (API). See the Embedded Event Manager Overview document in the *Network Management Configuration Guide* for more information on how EOT works with EEM.

Benefits of Enhanced Object Tracking

- Increases the availability and speed of recovery of a network.
- Decreases the number of network outages and their duration.
- Enables client processes such as VRRP and GLBP to track objects individually or as a list of objects. Prior to the introduction of this functionality, the tracking process was embedded within HSRP.

How to Configure Enhanced Object Tracking

Tracking the Line-Protocol State of an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track timer interface** {*seconds* | **msec** *milliseconds*}
4. **track** *object-number* **interface** *type number* **line-protocol**
5. **carrier-delay**
6. **delay** {**up** *seconds* [**down** [*seconds*] | [**up** *seconds*] **down** *seconds*]}
7. **end**
8. **show track** *object-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**track timer interface** {*seconds* | **msec** *milliseconds*}
	Example:
	`Device(config)# track timer interface 5`
		(Optional) Specifies the interval in which the tracking process polls the tracked object.
		• The default interval that the tracking process polls interface objects is 1 second.
		Note All polling frequencies can be configured down to 500 milliseconds, overriding the minimum 1-second interval configured using the **msec** keyword and *milliseconds* argument.

Step 4	**track** *object-number* **interface** *type number* **line-protocol** **Example:** `Device(config)# track 3 interface ethernet 0/1 line-protocol`	Tracks the line-protocol state of an interface and enters tracking configuration mode.
Step 5	**carrier-delay** **Example:** `Device(config-track)# carrier-delay`	(Optional) Enables EOT to consider the carrier-delay timer when tracking the status of an interface.

Step 6	**delay** {**up** *seconds* [**down** [*seconds*] | [**up** *seconds*] **down** *seconds*]}
	Example:
	`Device(config-track)# delay up 30`
		(Optional) Specifies a period of time (in seconds) to delay communicating state changes of a tracked object.

Step 7	**end**
	Example:
	`Device(config-track)# end`
		Exits to privileged EXEC mode.

Step 8	**show track** *object-number* **Example:** `Device# show track 3`	(Optional) Displays tracking information. • Use this command to verify the configuration.

Example

The following example shows the state of the line protocol on an interface when it is tracked:

```
Device# show track 3

Track 3
   Interface Ethernet0/1 line-protocol
   Line protocol is Up
     1 change, last change 00:00:05
   Tracked by:
     HSRP Ethernet0/3 1
```

Tracking the IP-Routing State of an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track timer interface** {*seconds* | **msec** *milliseconds*}
4. **track** *object-number* **interface** *type number* **ip routing**
5. **carrier-delay**
6. **delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
7. **end**
8. **show track** *object-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**track timer interface** {*seconds* | **msec** *milliseconds*}
	Example:
	`Device(config)# track timer interface 5`
		(Optional) Specifies the interval in which the tracking process polls the tracked object.
		• The default interval that the tracking process polls interface objects is 1 second.
		Note All polling frequencies can be configured down to 500 milliseconds, overriding the minimum 1-second interval configured using the **msec** keyword and *milliseconds* argument.

	Command or Action	Purpose
Step 4	**track** *object-number* **interface** *type number* **ip routing** **Example:** `Device(config)# track 1 interface ethernet 0/1 ip routing`	Tracks the IP-routing state of an interface and enters tracking configuration mode. • IP-route tracking tracks an IP route in the routing table and the ability of an interface to route IP packets.
Step 5	**carrier-delay** **Example:** `Device(config-track)# carrier-delay`	(Optional) Enables EOT to consider the carrier-delay timer when tracking the status of an interface.

Step 6	**delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
	Example:
	`Device(config-track)# delay up 30`
		(Optional) Specifies a period of time (in seconds) to delay communicating state changes of a tracked object.

Step 7	**end**
	Example:
	`Device(config-track)# end`
		Returns to privileged EXEC mode.

Step 8	**show track** *object-number* **Example:** `Device# show track 1`	Displays tracking information. • Use this command to verify the configuration.

Example

The following example shows the state of IP routing on an interface when it is tracked:

```
Device# show track 1

Track 1
   Interface Ethernet0/1 ip routing
   IP routing is Up
     1 change, last change 00:01:08
   Tracked by:
     HSRP Ethernet0/3 1
```

Tracking IP-Route Reachability

Perform this task to track the reachability of an IP route. A tracked object is considered up when a routing table entry exists for the route and the route is accessible.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track timer ip route** {*seconds* | **msec** *milliseconds*}
4. **track** *object-number* **ip route** *ip-address/prefix-length* **reachability**
5. **delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
6. **ip vrf** *vrf-name*
7. **end**
8. **show track** *object-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track timer ip route** {*seconds* | **msec** *milliseconds*} **Example:** `Device(config)# track timer ip route 20`	(Optional) Specifies the interval in which the tracking process polls the tracked object. • The default interval that the tracking process polls IP-route objects is 15 seconds. **Note** All polling frequencies can be configured down to 500 milliseconds, overriding the minimum 1-second interval configured using the **msec** keyword and *milliseconds* argument.
Step 4	**track** *object-number* **ip route** *ip-address/prefix-length* **reachability** **Example:** `Device(config)# track 4 ip route 10.16.0.0/16 reachability`	Tracks the reachability of an IP route and enters tracking configuration mode.

	Command or Action	Purpose

Step 5	**delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
	Example:
	`Device(config-track)# delay up 30`
		(Optional) Specifies a period of time (in seconds) to delay communicating state changes of a tracked object.

Step 6	**ip vrf** *vrf-name* **Example:** `Device(config-track)# ip vrf VRF2`	(Optional) Configures a VPN routing and forwarding (VRF) table.

Step 7	**end**
	Example:
	`Device(config-track)# end`
		Returns to privileged EXEC mode.

Step 8	**show track** *object-number* **Example:** `Device# show track 4`	(Optional) Displays tracking information. • Use this command to verify the configuration.

Example

The following example shows the state of the reachability of an IP route when it is tracked:

```
Device# show track 4

Track 4
   IP route 10.16.0.0 255.255.0.0 reachability
   Reachability is Up (RIP)
     1 change, last change 00:02:04
   First-hop interface is Ethernet0/1
   Tracked by:
     HSRP Ethernet0/3 1
```

Tracking the Threshold of IP-Route Metrics

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track timer ip route** {*seconds* | **msec** *milliseconds*}
4. **track resolution ip route** {**eigrp** | **isis** | **ospf** | **static**} *resolution-value*
5. **track** *object-number* **ip route** *ip-address/prefix-length* **metric threshold**
6. **delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
7. **ip vrf** *vrf-name*
8. **threshold metric** {**up** *number* [**down** *number*] | **down** *number* [**up** *number*]}
9. **end**
10. **show track** *object-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track timer ip route** {*seconds* | **msec** *milliseconds*} **Example:** `Device(config)# track timer ip route 20`	(Optional) Specifies the interval in which the tracking process polls the tracked object. • The default interval that the tracking process polls IP-route objects is 15 seconds. **Note** All polling frequencies can be configured down to 500 milliseconds, overriding the minimum 1-second interval configured using the **msec** keyword and *milliseconds* argument.
Step 4	**track resolution ip route** {**eigrp** | **isis** | **ospf** | **static**} *resolution-value* **Example:** `Device(config)# track resolution ip route eigrp 300`	(Optional) Specifies resolution parameters for a tracked object. • Use this command to change the default metric resolution values.

	Command or Action	Purpose
Step 5	**track** *object-number* **ip route** *ip-address/prefix-length* **metric threshold** **Example:** `Device(config)# track 6 ip route 10.16.0.0/16 metric threshold`	Tracks the scaled metric value of an IP route to determine if it is above or below a threshold and enters tracking configuration mode. • The default down value is 255, which equates to an inaccessible route. • The default up value is 254.

Step 6	**delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
	Example:
	`Device(config-track)# delay up 30`
		(Optional) Specifies a period of time (in seconds) to delay communicating state changes of a tracked object.

Step 7	**ip vrf** *vrf-name* **Example:** `Device(config-track)# ip vrf VRF1`	(Optional) Configures a VRF table.

Step 8	**threshold metric** {**up** *number* [**down** *number*] | **down** *number* [**up** *number*]}
	Example:
	`Device(config-track)# threshold metric up 254 down 255`
		(Optional) Sets a metric threshold other than the default value.

Step 9	**end**
	Example:
	`Device(config-track)# end`
		Exits to privileged EXEC mode.

Step 10	**show track** *object-number* **Example:** `Device# show track 6`	(Optional) Displays tracking information. • Use this command to verify the configuration.

Example

The following example shows the metric threshold of an IP route when it is tracked:

```
Device# show track 6

Track 6
   IP route 10.16.0.0 255.255.0.0 metric threshold
   Metric threshold is Up (RIP/6/102)
```

```
  1 change, last change 00:00:08
Metric threshold down 255 up 254
First-hop interface is Ethernet0/1
Tracked by:
  HSRP Ethernet0/3 1
```

Tracking the State of an IP SLAs Operation

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track** *object-number* **ip sla** *operation-number* **state**
4. **delay** {**up** *seconds* [**down** *seconds* | [**up** *seconds*] **down** *seconds*}
5. **end**
6. **show track** *object-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *object-number* **ip sla** *operation-number* **state** **Example:** `Device(config)# track 2 ip sla 4 state`	Tracks the state of an IP SLAs object and enters tracking configuration mode. With CScsf08092, the **track rtr** command was replaced by the **track ip sla** command.

Step 4	**delay** {**up** *seconds* [**down** *seconds* | [**up** *seconds*] **down** *seconds*}
	Example:
	`Device(config-track)# delay up 60 down 30`
		(Optional) Specifies a period of time (in seconds) to delay communicating state changes of a tracked object.

Step 5	**end**
	Example:
	`Device(config-track)# end`
		Exits to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show track** *object-number* **Example:** `Device# show track 2`	(Optional) Displays tracking information. • Use this command to verify the configuration.

Example

The following example shows the state of the IP SLAs tracking:

```
Device# show track 2

Track 2
   IP SLA 1 state
   State is Down
     1 change, last change 00:00:47
   Latest operation return code: over threshold
   Latest RTT (millisecs) 4
   Tracked by:
     HSRP Ethernet0/1 3
```

Tracking the Reachability of an IP SLAs IP Host

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track** *object-number* **ip sla** *operation-number* **reachability**
4. **delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down***seconds*}
5. **end**
6. **show track** *object-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *object-number* **ip sla** *operation-number* **reachability** **Example:** `Device(config)#` **`track 2 ip sla 4 reachability`**	Tracks the reachability of an IP SLAs IP host and enters tracking configuration mode. **Note** With CScsf08092, the **track rtr** command was replaced by the **track ip sla** command.

Step 4	**delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down***seconds*}
	Example:
	`Device(config-track)# delay up 30 down 10`
		(Optional) Specifies a period of time (in seconds) to delay communicating state changes of a tracked object.

Step 5	**end**
	Example:
	`Device(config-track)# end`
		Exits to privileged EXEC mode.

Step 6	**show track** *object-number* **Example:** `Device# show track 3`	(Optional) Displays tracking information. - Use this command to verify the configuration.

Example

The following example shows whether the route is reachable:

```
Device# show track 3

Track 3
   IP SLA 1 reachability
   Reachability is Up
     1 change, last change 00:00:47
   Latest operation return code: over threshold
   Latest RTT (millisecs) 4
   Tracked by:
     HSRP Ethernet0/1 3
```

Configuring a Tracked List and Boolean Expression

Perform this task to configure a tracked list of objects and a Boolean expression to determine the state of the list. A tracked list contains one or more objects. The Boolean expression enables two types of calculations by using either “and” or “or” operators. For example, when you configure tracking for two interfaces using the “and” operator up means that *both* interfaces are up, and down means that either interface is down.

You may configure a tracked list state to be measured using a weight or percentage threshold. See the Configuring a Tracked List and Threshold Weight section and the Configuring a Tracked List and Threshold Percentage section.

Before You Begin

An object must exist before it can be added to a tracked list.

Note The “not” operator is specified for one or more objects and negates the state of the object.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track** *track-number* **list boolean** {**and** | **or**}
4. **object** *object-number* [**not**]
5. **delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**track** *track-number* **list boolean** {**and** | **or**}
	Example:
	`Device(config)# track 100 list boolean and`
		Configures a tracked list object and enters tracking configuration mode.

	Command or Action	Purpose
Step 4	**object** *object-number* [**not**] **Example:** `Device(config-track)# object 3 not`	Specifies the object to be tracked. • The*object-number* argument has a valid range from 1 to 500. There is no default. The optional **not** keyword negates the state of the object. **Note** The example means that when object 3 is up, the tracked list detects object 3 as down.

Step 5	**delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
	Example:
	`Device(config-track)# delay up 3`
		(Optional) Specifies a tracking delay in seconds between up and down states.

Step 6	**end**
	Example:
	`Device(config-track)# end`
		Returns to privileged EXEC mode.

Configuring a Tracked List and Threshold Weight

Perform this task to configure a list of tracked objects, to specify that weight be used as the threshold, and to configure a weight for each of the objects in the list of tracked objects. A tracked list contains one or more objects. Enhanced object tracking uses a threshold weight to determine the state of each object by comparing the total weight of all objects that are up against a threshold weight for each object.

You can also configure a tracked list state to be measured using a Boolean calculation or threshold percentage. See the Configuring a Tracked List and Boolean Expression section and the Configuring a Tracked List and Threshold Percentage section.

Before You Begin

An object must exist before it can be added to a tracked list.

Note You cannot use the Boolean “not” operator in a weight or percentage threshold list.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track** *track-number* **list threshold weight**
4. **object** *object-number* [**weight** *weight-number*]
5. **threshold weight** {**up** *number* **down** *number* | **up** *number* | **down** *number*}
6. **delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *track-number* **list threshold weight** **Example:** `Device(config)# track 100 list threshold weight`	Configures a tracked list object and enters tracking configuration mode. The keywords are as follows: • **threshold** —Specifies that the state of the tracked list is based on a threshold. • **weight** —Specifies that the threshold is based on a specified weight.
Step 4	**object** *object-number* [**weight** *weight-number*] **Example:** `Device(config-track)# object 3 weight 30`	Specifies the object to be tracked. The *object-number* argument has a valid range from 1 to 500. There is no default. The optional **weight** keyword specifies a threshold weight for each object.

Step 5	**threshold weight** {**up** *number* **down** *number* | **up** *number* | **down** *number*}
	Example:
	`Device(config-track)# threshold weight up 30`
		Specifies the threshold weight.
		• **up** *number* —Valid range is from 1 to 255.
		• **down** *number*—Range depends upon what you select for the **up** keyword. For example, if you configure 25 for up, you will see a range from 0 to 24 for down.

	Command or Action	Purpose

Step 6	**delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
	Example:
	`Device(config-track)# delay up 3`
		(Optional) Specifies a tracking delay in seconds between up and down states.

Step 7	**end**
	Example:
	`Device(config-track)# end`
		Returns to privileged EXEC mode.

Configuring a Tracked List and Threshold Percentage

Perform this task to configure a tracked list of objects, to specify that a percentage will be used as the threshold, and to specify a percentage for each object in the list. A tracked list contains one or more objects. Enhanced object tracking uses the threshold percentage to determine the state of the list by comparing the assigned percentage of each object to the list.

You may also configure a tracked list state to be measured using a Boolean calculation or threshold weight. See the Configuring a Tracked List and Boolean Expression section and theConfiguring a Tracked List and Threshold Weight section.

Note You cannot use the Boolean "not" operator in a weight or percentage threshold list.

Before You Begin

An object must exist before it can be added to a tracked list.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track** *track-number* **list threshold percentage**
4. **object** *object-number*
5. **threshold percentage** {**up** *number* [**down** *number*] | **down** *number* [**up** *number*]}
6. **delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *track-number* **list threshold percentage** **Example:** `Device(config)# track 100 list threshold percentage`	Configures a tracked list object and enters tracking configuration mode. The keywords are as follows: - **threshold** —Specifies that the state of the tracked list is based on a threshold. - **percentage** —Specifies that the threshold is based on a percentage.
Step 4	**object** *object-number* **Example:** `Device(config-track)# object 3`	Specifies the object to be tracked. - The *object-number* argument has a valid range from 1 to 500. There is no default.

Step 5	**threshold percentage** {**up** *number* [**down** *number*] | **down** *number* [**up** *number*]}
	Example:
	`Device(config-track)# threshold percentage up 30`
		Specifies the threshold percentage.
		- **up** *number*—Valid range is from 1 to 100.
		- **down** *number* —Range depends upon what you have selected for the **up** keyword. For example, if you specify 25 as up, a range from 26 to 100 is displayed for the **down** keyword.

Step 6	**delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
	Example:
	`Device(config-track)# delay up 3`
		(Optional) Specifies a tracking delay in seconds between up and down states.

Step 7	**end**
	Example:
	`Device(config-track)# end`
		Returns to privileged EXEC mode.

Configuring Track List Defaults

Perform this task to configure a default delay value for a tracked list, a default object, and default threshold parameters for a tracked list.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track** *track-number*
4. **default** {**delay** | **object** *object-number* | **threshold percentage**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *track-number* **Example:** `Device(config)# track 3`	Enters tracking configuration mode.

Step 4	**default** {**delay** | **object** *object-number* | **threshold percentage**}
	Example:
	`Device(config-track)# default delay`
		Specifies a default delay value for a tracked list, a default object, and default threshold parameters for a tracked list.
		• **delay** —Reverts to the default delay.
		• **object** *object-number*—Specifies a default object for the track list. The valid range is from 1 to 1000.
		• **threshold percentage**—Specifies a default threshold percentage.

	Command or Action	Purpose

Step 5	**end**
	Example:
	`Device(config-track)# end`
		Returns to privileged EXEC mode.

Configuring Tracking for Mobile IP Applications

Perform this task to configure a tracked list of Mobile IP application objects.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **track** *track-number* **application home-agent**
4. **exit**
5. **track** *track-number* **application pdsn**
6. **exit**
7. **track** *track-number* **application ggsn**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**track** *track-number* **application home-agent** **Example:** `Device(config)# track 100 application home-agent`	(Optional) Tracks the presence of Home Agent traffic on a router and enters tracking configuration mode.

	Command or Action	Purpose
Step 4	**exit** **Example:** `Device(config-track)# exit`	Returns to global configuration mode.
Step 5	**track** *track-number* **application pdsn** **Example:** `Device(config)# track 100 application pdsn`	(Optional) Tracks the presence of Packet Data Serving Node (PDSN) traffic on a router tracking configuration mode.
Step 6	**exit** **Example:** `Device(config-track)# exit`	Returns to global configuration mode.
Step 7	**track** *track-number* **application ggsn** **Example:** `Device(config)# track 100 application ggsn`	(Optional) Tracks the presence of Gateway GPRS Support Node (GGSN) traffic on a router tracking configuration mode.

Step 8	**end**
	Example:
	`Device(config)# end`
		Returns to privileged EXEC mode.

Configuration Examples for Enhanced Object Tracking

Example: Interface Line Protocol

In the following example, the tracking process is configured to track the line-protocol state of GigabitEthernet interface 1/0/0. HSRP on GigabitEthernet interface 0/0/0 then registers with the tracking process to be informed of any changes to the line-protocol state of GigabitEthernet interface 1/0/0. If the line protocol on GigabitEthernet interface 1/0/0 goes down, the priority of the HSRP group is reduced by 10.

Router A Configuration

```
Device(config)# track 100 interface GigabitEthernet1/0/0 line-protocol
!
Device(config)# interface GigabitEthernet0/0/0
Device(config-if)# ip address 10.1.0.21 255.255.0.0
Device(config-if)# standby 1 preempt
Device(config-if)# standby 1 ip 10.1.0.1
Device(config-if)# standby 1 priority 110
Device(config-if)# standby 1 track 100 decrement 10
```

Router B Configuration

```
Device(config)# track 100 interface GigabitEthernet1/0/0 line-protocol
!
Device(config)# interface GigabitEthernet0/0/0
Device(config-if)# ip address 10.1.0.22 255.255.0.0
Device(config-if)# standby 1 preempt
Device(config-if)# standby 1 ip 10.1.0.1
Device(config-if)# standby 1 priority 105
Device(config-if)# standby 1 track 100 decrement 10
```

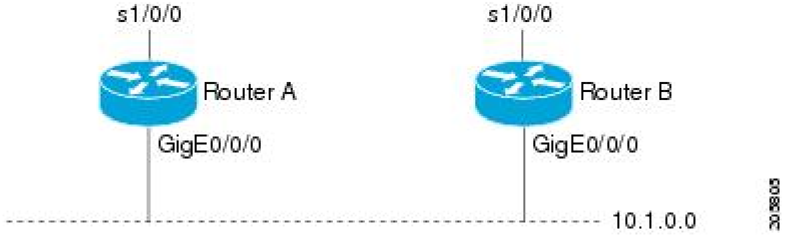
Example: Interface IP Routing

In the following example, the tracking process is configured to track the IP-routing capability of GigabitEthernet interface 1/0/0. HSRP on GigabitEthernet interface 0/0/0 then registers with the tracking process to be informed of any changes to the IP-routing state of GigabitEthernet interface 1/0/0. If the IP-routing state on GigabitEthernet interface 1/0/0 goes down, the priority of the HSRP group is reduced by 10.

If both serial interfaces are operational, Router A will be the HSRP active router because it has the higher priority. However, if IP on GigabitEthernet interface 1/0/0 in Router A fails, the HSRP group priority will be reduced and Router B will take over as the active router, thus maintaining a default virtual gateway service to hosts on the 10.1.0.0 subnet.

See the figure below for a sample topology.

Figure 1: Topology for IP-Routing Support



Router A Configuration

```
Device(config)# track 100 interface GigabitEthernet1/0/0 ip routing
!
Device(config)# interface GigabitEthernet0/0/0
Device(config-if)# ip address 10.1.0.21 255.255.0.0
Device(config-if)# standby 1 preempt
Device(config-if)# standby 1 ip 10.1.0.1
Device(config-if)# standby 1 priority 110
Device(config-if)# standby 1 track 100 decrement 10
```

Router B Configuration

```
Device(config)# track 100 interface GigabitEthernet1/0/0 ip routing
!
Device(config)# interface GigabitEthernet0/0/0
Device(config-if)# ip address 10.1.0.22 255.255.0.0
Device(config-if)# standby 1 preempt
Device(config-if)# standby 1 ip 10.1.0.1
Device(config-if)# standby 1 priority 105
Device(config-if)# standby 1 track 100 decrement 10
```

Example: IP-Route Reachability

In the following example, the tracking process is configured to track the reachability of IP route 10.2.2.0/24:

Router A Configuration

```
Device(config)# track 100 ip route 10.2.2.0/24 reachability
!
Device(config)# interface GigabitEthernet0/0/0
Device(config-if)# ip address 10.1.1.21 255.255.255.0
Device(config-if)# standby 1 preempt
Device(config-if)# standby 1 ip 10.1.1.1
Device(config-if)# standby 1 priority 110
Device(config-if)# standby 1 track 100 decrement 10
```

Router B Configuration

```
Device(config)# track 100 ip route 10.2.2.0/24 reachability
!
Device(config)# interface GigabitEthernet0/0/0
Device(config-if)# ip address 10.1.1.22 255.255.255.0
Device(config-if)# standby 1 preempt
Device(config-if)# standby 1 ip 10.1.1.1
Device(config-if)# standby 1 priority 105
Device(config-if)# standby 1 track 100 decrement 10
```

Example: IP-Route Threshold Metric

In the following example, the tracking process is configured to track the threshold metric of IP route 10.2.2.0/24:

Router A Configuration

```
Device(config)# track 100 ip route 10.2.2.0/24 metric threshold
!
Device(config)# interface GigabitEthernet0/0/0
Device(config-if)# ip address 10.1.1.21 255.255.255.0
Device(config-if)# standby 1 preempt
Device(config-if)# standby 1 ip 10.1.1.1
Device(config-if)# standby 1 priority 110
Device(config-if)# standby 1 track 100 decrement 10
```

Router B Configuration

```
Device(config)# track 100 ip route 10.2.2.0/24 metric threshold
!
Device(config)# interface GigabitEthernet0/0/0
Device(config-if)# ip address 10.1.1.22 255.255.255.0
Device(config-if)# standby 1 preempt
Device(config-if)# standby 1 ip 10.1.1.1
Device(config-if)# standby 1 priority 105
Device(config-if)# standby 1 track 100 decrement 10
```

Example: IP SLAs IP Host Tracking

The following example shows how to configure IP host tracking for IP SLAs operation 1 prior to CSCsf08092:

```
Device(config)# ip sla 1
Device(config-ip-sla)# icmp-echo 10.51.12.4
Device(config-ip-sla-echo)# timeout 1000
Device(config-ip-sla-echo)# threshold 2
Device(config-ip-sla-echo)# frequency 3
Device(config-ip-sla-echo)# request-data-size 1400
Device(config-ip-sla-echo)# exit
Device(config)# ip sla schedule 1 start-time now life forever
Device(config-ip-sla)# track 2 rtr 1 state
Device(config-ip-sla)# exit
Device(config)# track 3 rtr 1 reachability
Device(config-track)# exit
Device(config)# interface ethernet0/1
Device(config-if)# ip address 10.21.0.4 255.255.0.0
Device(config-if)# no shutdown
Device(config-if)# standby 3 ip 10.21.0.10
Device(config-if)# standby 3 priority 120
Device(config-if)# standby 3 preempt
Device(config-if)# standby 3 track 2 decrement 10
Device(config-if)# standby 3 track 3 decrement 10
```

The following example shows how to configure IP host tracking for IP SLAs operation 1 prior to CSCsf08092:

```
Device(config)# ip sla 1
Device(config-ip-sla)# icmp-echo 10.51.12.4
Device(config-ip-sla-echo)# threshold 2
Device(config-ip-sla-echo)# timeout 1000
Device(config-ip-sla-echo)# frequency 3
Device(config-ip-sla-echo)# request-data-size 1400
Device(config-ip-sla-echo)# exit
Device(config)# ip sla schedule 1 start-time now life forever
Device(config)# track 2 ip sla 1 state
Device(config-track)# exit
Device(config)# track 3 ip sla 1 reachability
Device(config-track)# exit
Device(config)# interface ethernet0/1
Device(config-if)# ip address 10.21.0.4 255.255.0.0
Device(config-if)# no shutdown
Device(config-if)# standby 3 ip 10.21.0.10
Device(config-if)# standby 3 priority 120
Device(config-if)# standby 3 preempt
Device(config-if)# standby 3 track 2 decrement 10
Device(config-if)# standby 3 track 3 decrement 10
```

Example: Boolean Expression for a Tracked List

In the following example, a track list object is configured to track two GigabitEthernet interfaces when both interfaces are up and when either interface is down:

```
Device(config)# track 1 interface GigabitEthernet2/0/0 line-protocol
Device(config)# track 2 interface GigabitEthernet2/1/0 line-protocol
Device(config-track)# exit
Device(config)# track 100 list boolean and
Device(config-track)# object 1
Device(config-track)# object 2
```

In the following example, a track list object is configured to track two GigabitEthernet interfaces when either interface is up and when both interfaces are down:

```
Device(config)# track 1 interface GigabitEthernet2/0/0 line-protocol
Device(config)# track 2 interface GigabitEthernet2/1/0 line-protocol
Device(config-track)# exit
Device(config)# track 101 list boolean or
Device(config-track)# object 1
Device(config-track)# object 2
```

The following configuration example shows that tracked list 4 has two objects and one object state is negated (if the list is up, the list detects that object 2 is down):

```
Device(config)# track 4 list boolean and
Device(config-track)# object 1
Device(config-track)# object 2 not
```

Example: Threshold Weight for a Tracked List

In the following example, three GigabitEtherent interfaces in tracked list 100 are configured with a threshold weight of 20 each. The down threshold is configured to 0 and the up threshold is configured to 40:

```
Device(config)# track 1 interface GigabitEthernet2/0/0 line-protocol
Device(config)# track 2 interface GigabitEthernet2/1/0 line-protocol
Device(config)# track 3 interface GigabitEthernet2/2/0 line-protocol
Device(config-track)# exit
Device(config)# track 100 list threshold weight
Device(config-track)# object 1 weight 20
Device(config-track)# object 2 weight 20
Device(config-track)# object 3 weight 20
Device(config-track)# threshold weight  up 40 down 0
```

In the example above the track-list object goes down only when all three serial interfaces go down, and comes up again only when at least two interfaces are up (because 20 + 20 >= 40). The advantage of this configuration is that it prevents the track-list object from coming up if two interfaces are down and the third interface is flapping.

The following configuration example shows that if object 1 and object 2 are down, then track list 4 is up, because object 3 satisfies the up threshold value of up 30. But, if object 3 is down, both objects 1 and 2 need to be up in order to satisfy the threshold weight.

```
Device(config)# track 4 list threshold weight
Device(config-track)# object 1 weight 15
Device(config-track)# object 2 weight 20
Device(config-track)# object 3 weight 30
Device(config-track)# threshold weight up 30 down 10
```

This configuration may be useful to you if you have two small bandwidth connections (represented by object 1 and 2) and one large bandwidth connection (represented by object 3). Also the down 10 value means that once the tracked object is up, it will not go down until the threshold value is lower or equal to 10, which in this example means that all connections are down.

Example: Threshold Percentage for a Tracked List

In the following example, four GigabitEthernet interfaces in track list 100 are configured for an up threshold percentage of 75. The track list is up when 75 percent of the interfaces are up and down when fewer than 75 percent of the interfaces are up.

```
Device(config)# track 1 interface GigabitEthernet2/0/0 line-protocol
Device(config)# track 2 interface GigabitEthernet2/1/0 line-protocol
Device(config)# track 3 interface GigabitEthernet2/2/0 line-protocol
Device(config)# track 4 interface GigabitEthernet2/3/0 line-protocol
Device(config-track)# exit
Device(config)# track 100 list threshold percentage
Device(config-track)# object 1
Device(config-track)# object 2
Device(config-track)# object 3
Device(config-track)# object 4
Device(config-track)# threshold percentage up 75
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Embedded Event Manager	*Embedded Event Manager Overview*
HSRP concepts and configuration tasks	*Configuring HSRP*
GLBP concepts and configuration tasks	*Configuring GLBP*
IP SLAs commands	*Cisco IOS IP SLAs Command Reference*
VRRP concepts and configuration tasks	*Configuring VRRP*
GLBP, HSRP, and VRRP commands	*Cisco IOS IP Application Services Command Reference*

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
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

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 Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Enhanced Object Tracking

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 3: Feature Information for Enhanced Object Tracking

Feature Name	Releases	Feature Information
Enhanced Tracking Support	15.0(1)SY	The Enhanced Tracking Support feature separates the tracking mechanism from HSRP and creates a separate standalone tracking process that can be used by other Cisco IOS processes and HSRP. This feature allows tracking of other objects in addition to the interface line-protocol state. The following commands were introduced or modified: **show track**, **standby track**, **threshold metric**, **track interface**, **track ip route**, **track timer**.
FHRP—Enhanced Object Tracking Integration with Embedded Event Manager	15.0(1)SY	EOT is integrated with Embedded Event Manager (EEM) to allow EEM to report on a status change of a tracked object and to allow EOT to track EEM objects. The following commands were introduced or modified by this feature: **default-state**, **event resource**, **event rf**, **event track**, **show track**, **track stub**.
FHRP—Enhanced Object Tracking of IP SLAs Operations	15.0(1)SY	This feature enables First Hop Redundancy Protocols (FHRPs) and other Enhanced Object Tracking (EOT) clients to track the output from IP SLAs objects and use the provided information to trigger an action. The following command was introduced by this feature: **track rtr**.
FHRP—EOT Deprecation of rtr Keyword	15.0(1)SY	This feature replaces the **track rtr** command with the **track ip sla** command.

Feature Name	Releases	Feature Information
FHRP—Object Tracking List	15.0(1)SY	This feature enhances the tracking capabilities to enable the configuration of a combination of tracked objects in a list, and a flexible method of combining objects using Boolean logic. The following commands were introduced or modified by this feature: **show track**, **threshold percentage**, **threshold weight**, **track list**, **track resolution**.

Glossary

DHCP—Dynamic Host Configuration Protocol. DHCP is a protocol that delivers IP addresses and configuration information to network clients.

GGSN—Gateway GPRS Support Node. A wireless gateway that allows mobile cell phone users to access the public data network (PDN) or specified private IP networks. The GGSN function is implemented on the Cisco routers.

GLBP—Gateway Load Balancing Protocol. Provides automatic router backup for IP hosts that are configured with a single default gateway on an IEEE 802.3 LAN. Multiple first-hop routers on the LAN combine to offer a single virtual first-hop IP router while sharing the IP packet forwarding load. Other routers on the LAN may act as redundant (GLBP) routers that will become active if any of the existing forwarding routers fail.

GPRS—General Packet Radio Service. A 2.5G mobile communications technology that enables mobile wireless service providers to offer their mobile subscribers with packet-based data services over GSM networks.

GSM network—Global System for Mobile Communications network. A digital cellular technology that is used worldwide, predominantly in Europe and Asia. GSM is the world's leading standard in digital wireless communications.

Home Agent—A Home Agent is a router on the home network of the Mobile Node (MN) that maintains an association between the home IP address of the MN and its care-of address, which is the current location of the MN on a foreign or visited network. The HA redirects packets by tunneling them to the MN while it is away from the home network.

HSRP—Hot Standby Router Protocol. Provides high network availability and transparent network topology changes. HSRP creates a Hot Standby router group with a lead router that services all packets sent to the Hot Standby address. The lead router is monitored by other routers in the group, and if it fails, one of these standby routers inherits the lead position and the Hot Standby group address.

IPCP—IP Control Protocol. The protocol used to establish and configure IP over PPP.

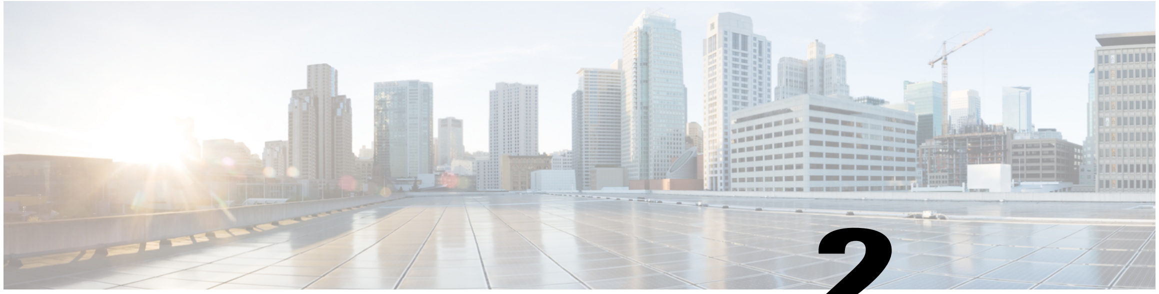
LCP—Link Control Protocol. The protocol used to establish, configure, and test data-link connections for use by PPP.

PDSN—Packet Data Serving Node. The Cisco PDSN is a standards-compliant, wireless gateway that enables packet data services in a Code Division Multiplex Access (CDMA) environment. Acting as an access gateway, the Cisco PDSN provides simple IP and Mobile IP access, foreign-agent support, and packet transport for Virtual Private Networks (VPN).

PPP—Point-to-Point Protocol. Provides router-to-router and host-to-network connections over synchronous and asynchronous circuits. PPP is most commonly used for dial-up Internet access. Its features include address notification, authentication via CHAP or PAP, support for multiple protocols, and link monitoring.

VRF—VPN routing and forwarding instance. A VRF consists of an IP routing table, a derived forwarding table, a set of interfaces that use the forwarding table, and a set of rules and routing protocols that determine what goes into the forwarding table. In general, a VRF includes the routing information that defines a customer VPN site that is attached to a provider edge router.

VRRP—Virtual Router Redundancy Protocol. Eliminates the single point of failure inherent in the static default routed environment. VRRP specifies an election protocol that dynamically assigns responsibility for a virtual router to one of the VRRP routers on a LAN. The VRRP router that controls the IP addresses associated with a virtual router is called the master, and forwards packets sent to these IP addresses. The election process provides dynamic failover in the forwarding responsibility should the master become unavailable. Any of the virtual router IP addresses on a LAN can then be used as the default first-hop router by end hosts.



CHAPTER 2

Configuring IP Services

This module describes how to configure optional IP services. For a complete description of the IP services commands in this chapter, refer to the *Cisco IOS IP Application Services Command Reference*. To locate documentation of other commands that appear in this module, use the master command list, or search online.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About IP Services

Cisco IP Accounting

Cisco IP accounting support provides basic IP accounting functions. By enabling IP accounting, users can see the number of bytes and packets switched through the software on a source and destination IP address basis. Only transit IP traffic is measured and only on an outbound basis; traffic generated by the software or terminating in the software is not included in the accounting statistics. To maintain accurate accounting totals, the software maintains two accounting databases: an active and a checkpointed database.

Cisco IP accounting support also provides information identifying IP traffic that fails IP access lists. Identifying IP source addresses that violate IP access lists alerts you to possible attempts to breach security. The data also indicates that you should verify IP access list configurations. To make this functionality available to users, you must enable IP accounting of access list violations using the **ip accounting access-violations** interface configuration command. Users can then display the number of bytes and packets from a single source that attempted to breach security against the access list for the source destination pair. By default, IP accounting displays the number of packets that have passed access lists and were routed.

How to Configure IP Services

Configuring IP Accounting

To configure IP accounting, perform this task for each interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip accounting-threshold** *threshold*
4. **ip accounting-list** *ip-address wildcard*
5. **ip accounting-transits** *count*
6. **interface** *type number*
7. **ip accounting** [**access-violations**] [**output-packets**]
8. **ip accounting mac-address** {**input** | **output**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip accounting-threshold** *threshold* **Example:** `Router(config)# ip accounting-threshold 500`	(Optional) Sets the maximum number of accounting entries to be created.
Step 4	**ip accounting-list** *ip-address wildcard* **Example:** `Router(config)# ip accounting-list 192.31.0.0 0.0.255.255`	(Optional) Filters accounting information for hosts.
Step 5	**ip accounting-transits** *count* **Example:** `Router(config)# ip accounting-transits 100`	(Optional) Controls the number of transit records that will be stored in the IP accounting database.
Step 6	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 1/0/0`	Specifies the interface and enters interface configuration mode.
Step 7	**ip accounting** [**access-violations**] [**output-packets**] **Example:** `Router(config-if)# ip accounting access-violations`	Configures basic IP accounting. • Use the optional **access-violations** keyword to enable IP accounting with the ability to identify IP traffic that fails IP access lists. • Use the optional **output-packets** keyword to enable IP accounting based on the IP packets output on the interface.

Step 8	**ip accounting mac-address** {**input** | **output**}
	Example:
	`Router(config-if)# ip accounting mac-address output`
		(Optional) Configures IP accounting based on the MAC address of received (input) or transmitted (output) packets.

Monitoring and Maintaining the IP Network

You can display specific statistics such as the contents of IP routing tables, caches, databases and socket processes. The resulting information can be used to determine resource utilization and to solve network problems.

SUMMARY STEPS

1. **clear ip traffic**
2. **clear ip accounting** [**checkpoint**]
3. **clear sockets** *process-id*
4. **show ip accounting** [**checkpoint**] [**output-packets** | **access-violations**]
5. **show interface** *type number* **mac**
6. **show interface** [*type number*] **precedence**
7. **show ip redirects**
8. **show sockets** *process-id* [**detail**] [**events**]
9. **show udp** [**detail**]
10. **show ip traffic**

DETAILED STEPS

Step 1 **clear ip traffic**

To clear all IP traffic statistical counters on all interfaces, use the following command:

Example:

```
Router# clear ip traffic
```

Step 2 **clear ip accounting** [**checkpoint**]

You can remove all contents of a particular cache, table, or database. Clearing a cache, table, or database can become necessary when the contents of the particular structure have become or are suspected to be invalid. To clear the active IP accounting database when IP accounting is enabled, use the following command:

Example:

```
Router# clear ip accounting
```

To clear the checkpointed IP accounting database when IP accounting is enabled, use the following command:

Example:

```
Router# clear ip accounting checkpoint
```

Step 3 **clear sockets** *process-id*

To close all IP sockets and clear the underlying transport connections and data structures for the specified process, use the following command:

Example:

```
Router# clear sockets 35

All sockets (TCP, UDP and SCTP) for this process will be cleared.
Do you want to proceed? [yes/no]: y
Cleared sockets for PID 35
```

Step 4 **show ip accounting** [**checkpoint**] [**output-packets** | **access-violations**]

To display access list violations, use the **show ip accounting** command. To use this command, you must first enable IP accounting on a per-interface basis.

Use the **checkpoint** keyword to display the checkpointed database. Use the **output-packets** keyword to indicate that information pertaining to packets that passed access control and were routed should be displayed. Use the **access-violations** keyword to display the number of the access list failed by the last packet for the source and destination pair. The number of packets reveals how aggressive the attack is upon a specific destination. If you do not specify the **access-violations** keyword, the command defaults to displaying the number of packets that have passed access lists and were routed.

If neither the **output-packets** nor **access-violations** keyword is specified, **output-packets** is the default.

The following is sample output from the **show ip accounting** command:

Example:

```
Router# show ip accounting

   Source           Destination              Packets               Bytes
 172.16.19.40    192.168.67.20                  7                 306
 172.16.13.55    192.168.67.20                  67                2749
 172.16.2.50     192.168.33.51                  17                1111
 172.16.2.50     172.31.2.1                     5                 319
 172.16.2.50     172.31.1.2                     463               30991
 172.16.19.40    172.16.2.1                      4                 262
 172.16.19.40    172.16.1.2                      28                2552
 172.16.20.2     172.16.6.100                    39                2184
 172.16.13.55    172.16.1.2                      35                3020
 172.16.19.40    192.168.33.51                  1986               95091
 172.16.2.50     192.168.67.20                   233               14908
 172.16.13.28    192.168.67.53                   390               24817
 172.16.13.55    192.168.33.51               214669              9806659
 172.16.13.111   172.16.6.23                  27739             1126607
 172.16.13.44    192.168.33.51                 35412             1523980
 192.168.7.21    172.163.1.2                      11                 824
 172.16.13.28    192.168.33.2                     21                1762
 172.16.2.166    192.168.7.130                   797              141054
 172.16.3.11     192.168.67.53                     4                 246
 192.168.7.21    192.168.33.51                 15696              695635
 192.168.7.24    192.168.67.20                    21                  916
 172.16.13.111   172.16.10.1                     16                1137
 accounting threshold exceeded for 7 packets and 433 bytes
```

The following is sample output from the **show ip accounting access-violations** command. The output pertains to packets that failed access lists and were not routed:

Example:

```
Router#  show ip accounting access-violations

   Source           Destination      Packets        Bytes        ACL
172.16.19.40    192.168.67.20              7          306         77
172.16.13.55    192.168.67.20             67         2749        185
172.16.2.50     192.168.33.51            17          1111        140
172.16.2.50     172.16.2.1                5          319        140
172.16.19.40    172.16.2.1                4          262         77
Accounting data age is 41
```

Step 5 **show interface** *type number* **mac**

To display information for interfaces configured for MAC accounting, use the **show interface mac** command. The following is sample output from the **show interface mac** command:

Example:

```
Router# show interface ethernet 0/1 mac

Ethernet0/1
Input  (511 free)
0007.f618.4449(228):  4 packets, 456 bytes, last: 2684ms ago
```

```
Total:  4 packets, 456 bytes
Output  (511 free)
0007.f618.4449(228):  4 packets, 456 bytes, last: 2692ms ago
Total:  4 packets, 456 bytes
```

Step 6 **show interface** [*type number*] **precedence**

To display information for interfaces configured for precedence accounting, use the **show interface precedence** command.

The following is sample output from the **show interface precedence** command. In this example, the total packet and byte counts are calculated for the interface that receives (input) or sends (output) IP packets and sorts the results based on IP precedence.

Example:

```
Router# show interface ethernet 0/1 precedence

Ethernet0/1
Input
Precedence 0:  4 packets, 456 bytes
Output
Precedence 0:  4 packets, 456 bytes
```

Step 7 **show ip redirects**

To display the address of the default router and the address of hosts for which an ICMP redirect message has been received, use the **show ip redirects**command.

Example:

```
Router#  show ip redirects

Default gateway is 172.16.80.29

Host               Gateway           Last Use    Total Uses  Interface
172.16.1.111      172.16.80.240         0:00             9  Ethernet0
172.16.1.4        172.16.80.240         0:00             4  Ethernet0
```

Step 8 **show sockets** *process-id* [**detail**] [**events**]

To display the number of sockets currently open and their distribution with respect to the transport protocol process specified by the *process-id* argument, use the **show sockets** command. The following sample output from the **show sockets** command displays the total number of open sockets for the specified process:

Example:

```
Router# show sockets 35

Total open sockets - TCP:7, UDP:0, SCTP:0
```

The following sample output shows information about the same open processes with the **detail** keyword specified:

Example:

```
Router# show sockets 35 detail

   FD LPort FPort Proto Type    TransID

   0  5000  0     TCP   STREAM  0x6654DEBC
State: SS_ISBOUND
Options: SO_ACCEPTCONN

   1  5001  0     TCP   STREAM  0x6654E494
State: SS_ISBOUND
Options: SO_ACCEPTCONN

   2  5002  0     TCP   STREAM  0x656710B0
```

```
State: SS_ISBOUND
Options: SO_ACCEPTCONN

   3  5003  0     TCP   STREAM  0x65671688
State: SS_ISBOUND
Options: SO_ACCEPTCONN

   4  5004  0     TCP   STREAM  0x65671C60
State: SS_ISBOUND
Options: SO_ACCEPTCONN

   5  5005  0     TCP   STREAM  0x65672238
State: SS_ISBOUND
Options: SO_ACCEPTCONN

   6  5006  0     TCP   STREAM  0x64C7840C
State: SS_ISBOUND
Options: SO_ACCEPTCONN

Total open sockets - TCP:7, UDP:0, SCTP:0
```

The following example displays IP socket event information:

Example:

```
Router# show sockets 35 events

Events watched for this process: READ
FD Watched Present Select Present

0 --- --- R-- R--
```

Step 9 **show udp** [**detail**]

To display IP socket information about UDP processes, use the **show udp** command. The following example shows how to display detailed information about UDP sockets:

Example:

```
Router# show udp detail

 Proto    Remote      Port      Local       Port   In Out Stat TTY OutputIF
 17       10.0.0.0    0         10.0.21.70  67     0  0   2211 0
 Queues: output 0
         input  0 (drops 0, max 50, highwater 0)
 Proto    Remote      Port      Local       Port   In Out Stat TTY OutputIF
 17       10.0.0.0    0         10.0.21.70  2517   0   0  11   0
 Queues: output 0
         input  0 (drops 0, max 50, highwater 0)
 Proto    Remote      Port      Local       Port   In Out Stat TTY OutputIF
 17       10.0.0.0    0         10.0.21.70  5000   0  0   211  0
 Queues: output 0
         input  0 (drops 0, max 50, highwater 0)
 Proto    Remote      Port      Local       Port   In Out Stat TTY OutputIF
 17       10.0.0.0    0         10.0.21.70  5001   0  0   211  0
 Queues: output 0
         input  0 (drops 0, max 50, highwater 0)
 Proto    Remote      Port      Local       Port   In Out Stat TTY OutputIF
 17       10.0.0.0    0         10.0.21.70  5002   0  0   211  0
 Queues: output 0
         input  0 (drops 0, max 50, highwater 0)
 Proto    Remote      Port      Local       Port   In Out Stat TTY OutputIF
 17       10.0.0.0    0         10.0.21.70  5003   0  0   211  0
 Queues: output 0
         input  0 (drops 0, max 50, highwater 0)
 Proto    Remote      Port      Local       Port   In Out Stat TTY OutputIF
 17       10.0.0.0    0         10.0.21.70  5004   0  0   211  0
 Queues: output 0
         input  0 (drops 0, max 50, highwater 0)
```

Step 10 **show ip traffic**

To display IP protocol statistics, use the **show ip traffic** command. The following example shows that the IP traffic statistics have been cleared by the **clear ip traffic** command:

Example:

```
Router# clear ip traffic

Router# show ip traffic

IP statistics:
 Rcvd:  0 total, 0 local destination
        0 format errors, 0 checksum errors, 0 bad hop count
        0 unknown protocol, 0 not a gateway
        0 security failures, 0 bad options, 0 with options
 Opts:  0 end, 0 nop, 0 basic security, 0 loose source route
        0 timestamp, 0 extended security, 0 record route
        0 stream ID, 0 strict source route, 0 alert, 0 cipso
        0 other
 Frags: 0 reassembled, 0 timeouts, 0 couldn't reassemble
        0 fragmented, 0 couldn't fragment
 Bcast: 0 received, 0 sent
 Mcast: 0 received, 0 sent
 Sent: 0 generated, 0 forwarded
 Drop: 0 encapsulation failed, 0 unresolved, 0 no adjacency
       0 no route, 0 unicast RPF, 0 forced drop

ICMP statistics:
 Rcvd: 0 format errors, 0 checksum errors, 0 redirects, 0 unreachable
       0 echo, 0 echo reply, 0 mask requests, 0 mask replies, 0 quench
       0 parameter, 0 timestamp, 0 info request, 0 other
       0 irdp solicitations, 0 irdp advertisements
 Sent: 0 redirects, 0 unreachable, 0 echo, 0 echo reply
       0 mask requests, 0 mask replies, 0 quench, 0 timestamp
       0 info reply, 0 time exceeded, 0 parameter problem
       0 irdp solicitations, 0 irdp advertisements

UDP statistics:
 Rcvd: 0 total, 0 checksum errors, 0 no port
 Sent: 0 total, 0 forwarded broadcasts

TCP statistics:
 Rcvd: 0 total, 0 checksum errors, 0 no port
 Sent: 0 total

Probe statistics:
 Rcvd: 0 address requests, 0 address replies
       0 proxy name requests, 0 where-is requests, 0 other
 Sent: 0 address requests, 0 address replies (0 proxy)
       0 proxy name replies, 0 where-is replies

EGP statistics:
 Rcvd: 0 total, 0 format errors, 0 checksum errors, 0 no listener
 Sent: 0 total

IGRP statistics:
 Rcvd: 0 total, 0 checksum errors
 Sent: 0 total

OSPF statistics:
 Rcvd: 0 total, 0 checksum errors
       0 hello, 0 database desc, 0 link state req
       0 link state updates, 0 link state acks

 Sent: 0 total
```

```
IP-IGRP2 statistics:
 Rcvd: 0 total
 Sent: 0 total

PIMv2 statistics: Sent/Received
 Total: 0/0, 0 checksum errors, 0 format errors
 Registers: 0/0, Register Stops: 0/0, Hellos: 0/0
 Join/Prunes: 0/0, Asserts: 0/0, grafts: 0/0
 Bootstraps: 0/0, Candidate_RP_Advertisements: 0/0

IGMP statistics: Sent/Received
 Total: 0/0, Format errors: 0/0, Checksum errors: 0/0
 Host Queries: 0/0, Host Reports: 0/0, Host Leaves: 0/0
 DVMRP: 0/0, PIM: 0/0
```

Configuration Examples for IP Services

Example: Configuring IP Accounting

The following example shows how to enable IP accounting based on the source and destination MAC address and based on IP precedence for received and transmitted packets:

```
Router# configure terminal
Router(config)# interface ethernet 0/5
Router(config-if)# ip accounting mac-address input
Router(config-if)# ip accounting mac-address output
Router(config-if)# ip accounting precedence input
Router(config-if)# ip accounting precedence output
```

The following example shows how to enable IP accounting with the ability to identify IP traffic that fails IP access lists and with the number of transit records that will be stored in the IP accounting database limited to 100:

```
Router# configure terminal
Router(config)# ip accounting-transits 100
Router(config)# interface ethernet 0/5
Router(config-if)# ip accounting output-packets
Router(config-if)# ip accounting access-violations
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
IP application services commands	Cisco IOS IP Application Services Command Reference

Standards and RFCs

Standard	Title
RFC 1256	ICMP Router Discovery Messages

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for IP Services

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 4: Feature Information for IP Services

Feature Name	Releases	Feature Information
IP Precedence Accounting	15.0(1)SY	The IP Precedence Accounting feature provides accounting information for IP traffic based on the precedence of any interface. This feature calculates the total packet and byte counts for an interface that receives or sends IP packets and sorts the results based on the IP precedence. This feature is supported on all interfaces and subinterfaces and supports Cisco Express Forwarding, distributed Cisco Express Forwarding, flow, and optimum switching. The following commands were introduced by this feature: **ip accounting precedence**, **show interface precedence**.

CHAPTER 3

Configuring IPv4 Broadcast Packet Handling

This module explains what IPv4 broadcast packets are, when they are used, and how to customize your router's configuration for situations when the default behavior for handling IPv4 broadcast packets isn't appropriate.

This module also explains some common scenarios that require customizing IPv4 broadcast packet handling by routers. For example, UDP forwarding of Dynamic Host Configuration Protocol (DHCP) traffic to ensure broadcast packets sent by DHCP clients can reach DHCP servers that are not on the same network segment as the client. Configuration tasks and examples are also provided in this module.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About IPv4 Broadcast Packet Handling

IP Unicast Address

An IP unicast address is not a broadcast addresses. A packet with an unicast destination IP address is intended for a specific IP host. For example, 172.16.1.1/32. Only the intended host of a unicast packets receives and processes the packet. This term is often used in conjunction with references to types of IP broadcast traffic. For example, a network administrator considering upgrading a router in a network must consider the amount of unicast, multicast, and broadcast traffic because each type of traffic can have a different effect on the performance of the router.

IP Broadcast Address

IP broadcast packets are sent to the destination IP broadcast address 255.255.255.255 (or the older but still occasionally used IP broadcast address of 000.000.000.000). The broadcast destination IP addresses 255.255.255.255 and 000.000.000.000 are used when a packet is intended for every IP-enabled device on a network.

Note Packets that use the broadcast IP address as the destination IP address are known as broadcast packets.

If routers forwarded IP broadcast packets by default, the packets would have to be forwarded out every interface that is enabled for IP because the 255.255.255.255 IP destination address is assumed to be reachable via every IP enabled interface in the router. Forwarding IP broadcast packets out every interface that is enabled for IP would result in what is known as a broadcast storm (network overload due to high levels of broadcast traffic). In order to avoid the IP packet broadcast storm that would be created if a router forwarded packets with a broadcast IP destination address out every IP-enabled interface, the default behavior for a router is to *not* forward broadcast packets. This is a key difference between routing IP traffic at Layer 3 versus bridging it at Layer 2. Layer 2 bridges by default forward IP broadcast traffic out every interface that is in a forwarding state, which can lead to scalability problems.

Some TCP/IP protocols use the IP broadcast address to either communicate with all of the hosts on a network segment or to identify the IP address of a specific host on a network segment. For example:

- Routing Information Protocol (RIP) version 1 sends routing table information using the IP broadcast address so that any other host on the network segment running RIP version 1 can receive and process the updates.
- The Address Resolution Protocol (ARP) is used to determine the Layer 2 MAC address of the host that owns a specific Layer 3 IP address. ARP sends an IP broadcast packet (that is also a Layer 2 broadcast frame) on the local network. All of the hosts on the local network receive the ARP broadcast packet because it is sent to as a Layer 2 broadcast frame. All of the hosts on the local network process the ARP packet because it is sent to the IP broadcast address. Only the host that owns the IP address indicated in the data area of the ARP packet responds to the ARP broadcast packet.

IP Directed Broadcast Address

An IP directed broadcast is intended to reach all hosts on a remote network. A router that needs to send data to a remote IP host when only the IP network address is known uses an IP directed broadcast to reach the remote host. For example, a directed broadcast sent by a host with an IP address of 192.168.100.1 with a destination IP address of 172.16.255.255 is intended only for hosts that are in the 172.16.0.0 address space (hosts that have an IP address that begins with 172.16.0.0).

An IP directed broadcast packet is routed through the network as a unicast packet until it arrives at the target subnet, where it is converted into a Layer 2 broadcast frame (MAC address of FFFF.FFFF.FFFF). Because of the nature of the IP addressing architecture, only the last router in the chain, the one that is connected directly to the target subnet, can conclusively identify a directed broadcast. For example, only a router with an interface connected to a network using an IP address in the 172.16.0.0/16 address space such as 172.16.1.1/16 can determine that a packet sent to 172.16.255.255 is a directed broadcast and convert it to a Layer 2 broadcast that is received by all hosts on the local network. The other routers in the network that are not connected to the 172.16.0.0/16 network forward packets addressed to 172.16.255.255 as if they were for a specific IP host.

All of the hosts on the remote network receive IP directed broadcasts after they are converted to Layer 2 broadcast frames. Ideally only the intended destination host will fully process the IP directed broadcast and respond to it. However, IP directed broadcasts can be used for malicious purposes. For example, IP directed broadcasts are used in "smurf" Denial of Service (DoS) attack and derivatives thereof. In a "smurf" attack, the attacker sends Internet Control Message Protocol (ICMP) echo requests (pings) to a directed broadcast address using the source IP address of the device that is the target of the attack. The target is usually a host inside a company’s network such as a web server. The ICMP echo requests are sent to an IP directed broadcast address in the company’s network that causes all the hosts on the target subnet to send ICMP echo replies to the device under attack. By sending a continuous stream of such requests, the attacker can create a much larger stream of replies, which can completely inundate the host that is under attack. For information on how IP directed broadcasts are used in DoS attacks, search the Internet for "IP directed broadcasts," "denial of service," and "smurf attacks."

Due to the security implications of allowing a router to forward directed broadcasts and the reduction in applications that require directed broadcasts, IP directed broadcasts are disabled by default in Cisco IOS Release 12.0 and later releases. If your network requires support for IP directed broadcasts, you can enable it on the interfaces that you want to translate the IP directed broadcasts to Layer 2 broadcasts using the **ip directed-broadcast** command. For example, if your router is receiving IP directed broadcasts on Fast Ethernet interface 0/0 for the network address assigned to Fast Ethernet interface 0/1, and you want the IP directed broadcasts to be translated to Layer 2 broadcasts out interface Fast Ethernet interface 0/1, configure the **ip directed-broadcast** command on Fast Ethernet interface 0/1. You can specify an access list to control which IP directed broadcasts are translated to Layer 2 broadcasts. When an access list is specified, only those IP packets permitted by the access list are eligible to be translated from directed broadcasts to Layer 2 broadcasts. For example, if you know that the only legitimate source IP address of any IP directed broadcasts in your network is 192.168.10.2, create an extended IP access list allowing traffic from 192.168.10.2 and assign the access list with the **ip directed-broadcast***access-list* command.

IP Directed Broadcasts

IP directed broadcasts are dropped by default. Dropping IP directed broadcasts reduces the risk of DoS attacks.

You can enable forwarding of IP directed broadcasts on an interface where the broadcast becomes a physical broadcast. You enable the translation of directed IP broadcast packets to Layer 2 broadcast frames on the interface that is connected to the IP network that the IP directed broadcast is addressed to. For example, if you need to translate IP directed broadcasts with the IP destination address of 172.16.10.255 to Layer 2 broadcast frames, you enable the translation on the interface that is connected to IP network 172.16.10.0/24.

You can specify an access list to control which directed broadcasts are forwarded. When an access list is specified, only those IP packets permitted by the access list are eligible to be translated from directed broadcasts to physical broadcasts.

IP directed broadcasts are disabled by default in Cisco IOS Release 12.0 and newer releases.

IP Multicast Addresses

IP multicast addresses are intended to reach an arbitrary subset of the hosts on a local network. IP broadcast addresses create a problem because every host must receive and process the data in each packet to determine if it contains information that the host must process further. IP multicast addresses resolve this problem by using well-known IP addresses that a host must be configured to recognize before it will process packets addressed to it. When a host receives an IP multicast packet, the host compares the IP multicast address with the list of multicast addresses it is configured to recognize. If the host is not configured to recognize the IP multicast address, the host ignores the packet instead of processing it further to analyze the data in the packet. Because the host can ignore the packet it spends less time and fewer resources than it would have had to spend if the packet had been an IP broadcast that had to be processed all the way to the data layer before it was discarded.

The range of IP addresses reserved for Class D multicast addresses is 224.0.0.0 to 239.255.255.255/32 (255.255.255.255).

Most of the TCP/IP routing protocols use IP multicast addresses to send routing updates and other information to hosts on the same local network that are running the same routing protocol. Many other applications such as audio/video streaming over the Internet use IP multicast addresses. For a list of the currently assigned IP multicast addresses see Internet Multicast Addresses.

Information on configuring network devices for IP multicast support is available in the following documentation:

- *Cisco IOS IP Multicast Configuration Guide*
- *Cisco IOS IP Multicast Command Reference*

Early IP Implementations

Several early IP implementations do not use the current broadcast address standard of 255.255.255.255. Instead, they use the old standard, which calls for all zeros (000.000.000.000) instead of all ones to indicate broadcast addresses. Many of these implementations do not recognize an all-1s broadcast address and fail to respond to the broadcast correctly. Others forward all-1s broadcasts by default, which causes a serious network overload known as a *broadcast storm*. Implementations that exhibit these problems include systems based on versions of Berkeley Standard Distribution (BSD) UNIX prior to Version 4.3.

DHCP and IPv4 Broadcast Packets

DHCP requires that the client (host requiring information from the DHCP server) send broadcast packets to find a DHCP server to request configuration information from. If the DHCP server is not on the same network segment as the client that is sending the DHCP broadcasts, the router must be configured to forward the DHCP requests to the appropriate network.

For more information on DHCP, see RFC 2131 *Dynamic Host Configuration Protocol,* at http://www.ietf.org/rfc/rfc2131.txt.

UDP Broadcast Packet Forwarding

UDP broadcast packets are used by TCP/IP protocols such as DHCP and applications that need to send the same data to multiple hosts concurrently. Because routers by default do not forward broadcast packets you need to customize your router's configuration if your network has UDP broadcast traffic on it. One option for forwarding UDP broadcast packets is to use the UDP forwarding feature. UDP forwarding rewrites the broadcast IP address of a UDP packet to either a unicast (specific host) IP address or a directed IP broadcast. After the address is rewritten the UDP packet is forwarded by all of the routers in the path to the destination network without requiring additional configuration changes on the other routers.

You can enable forwarding of UDP broadcast packets, such as DHCP requests, to a host, or to multiple hosts on the same target network. When a UDP broadcast packet is forwarded, the destination IP address is rewritten to match the address that you configure. For example, the **ip helper-address 172.16.10.2** command rewrites the IP destination address from 255.255.255.255 to 172.16.10.2.

To enable UDP broadcast packet forwarding to specific host, use a specific host IP address as the helper address when you configure the **ip helper-address** *address* command. To enable UDP broadcast packet forwarding to a range of hosts to allow for load sharing and redundancy, use an IP directed broadcast address as the helper address when you configure the **ip helper-address** *address* command.

UDP Broadcast Packet Flooding

You can allow IP broadcasts to be flooded throughout your network in a controlled fashion using the database created by theLayer 2 bridging Spanning Tree Protocol (STP). Enabling this feature also prevents flooding loops. In order to support this capability, the Cisco IOS software on your router must include support for transparent bridging, and transparent bridging must be configured on each interface that is to participate in the flooding. If bridging is not configured on an interface, the interface is still able to receive broadcasts. However, the interface will never forward broadcasts it receives, and the router will never use that interface to send broadcasts received on a different interface.

Packets that are forwarded to a single network address using the IP helper address mechanism can be flooded. Only one copy of the packet is sent on each network segment.

In order to be considered for flooding, packets must meet the following criteria. (These are the same conditions used to consider packet forwarding using IP helper addresses.)

- The packet must be a MAC-level broadcast (FFFF.FFFF.FFFF).
- The packet must be an IP-level broadcast (255.255.255.255).
- The packet must be a Trivial File Transfer Protocol (TFTP), Domain Name System (DNS), Time, NetBIOS, Neighbor Discovery (ND), or BOOTP packet, or a UDP protocol specified by the **ip forward-protocol udp** global configuration command.
- The time-to-live (TTL) value of the packet must be at least two.

If you want to send the flooded UDP packets to a specific host, you can change the Layer 3 IP broadcast address of the flooded UDP packets with the **ip broadcast-address** command in interface configuration mode. The address of the flooded UDP packets can be set to any desired IP address. The source address of the flooded UDP packet is never changed. The TTL value of the flooded UDP packet is decremented.

After a decision has been made to send the datagram out on an interface (and the destination IP address possibly changed), the datagram is handed to the normal IP output routines and is, therefore, subject to access lists if they are present on the output interface.

If no actual bridging is desired, you can configure a type-code bridging filter that will deny all packet types from being bridged. Refer to the "Configuring Transparent Bridging" module of the *Cisco IOS Bridging and IBM Networking Configuration Guide* for more information about using access lists to filter bridged traffic. The Spanning-Tree database is still available to the IP forwarding code to use for the flooding.

IP Broadcast Flooding Acceleration

You can accelerate flooding of UDP datagrams using the spanning-tree algorithm. Used in conjunction with the **ip forward-protocol spanning-tree** command in global configuration mode, this feature boosts the performance of spanning-tree-based UDP flooding by a factor of about four to five times. The feature, called *turbo flooding*, is supported over Ethernet interfaces configured for Advanced Research Projects Agency (ARPA) encapsulated, FDDI, and high-level data link control (HDLC)-encapsulated serial interfaces. However, it is not supported on Token Ring interfaces. As long as the Token Rings and the non-HDLC serial interfaces are not part of the bridge group being used for UDP flooding, turbo flooding will behave normally.

Default UDP Port Numbers

If a helper address is specified and UDP forwarding is enabled, broadcast packets destined to the following port numbers are forwarded by default:

- Time service (port 37)
- IEN-116 Name Service (port 42)
- TACACS service (port 49)
- Domain Naming System (port 53)
- BOOTP client and server packets (ports 67 and 68)
- TFTP (port 69)
- NetBIOS Name Server (port 137)
- NetBIOS Datagram Server (port 138)

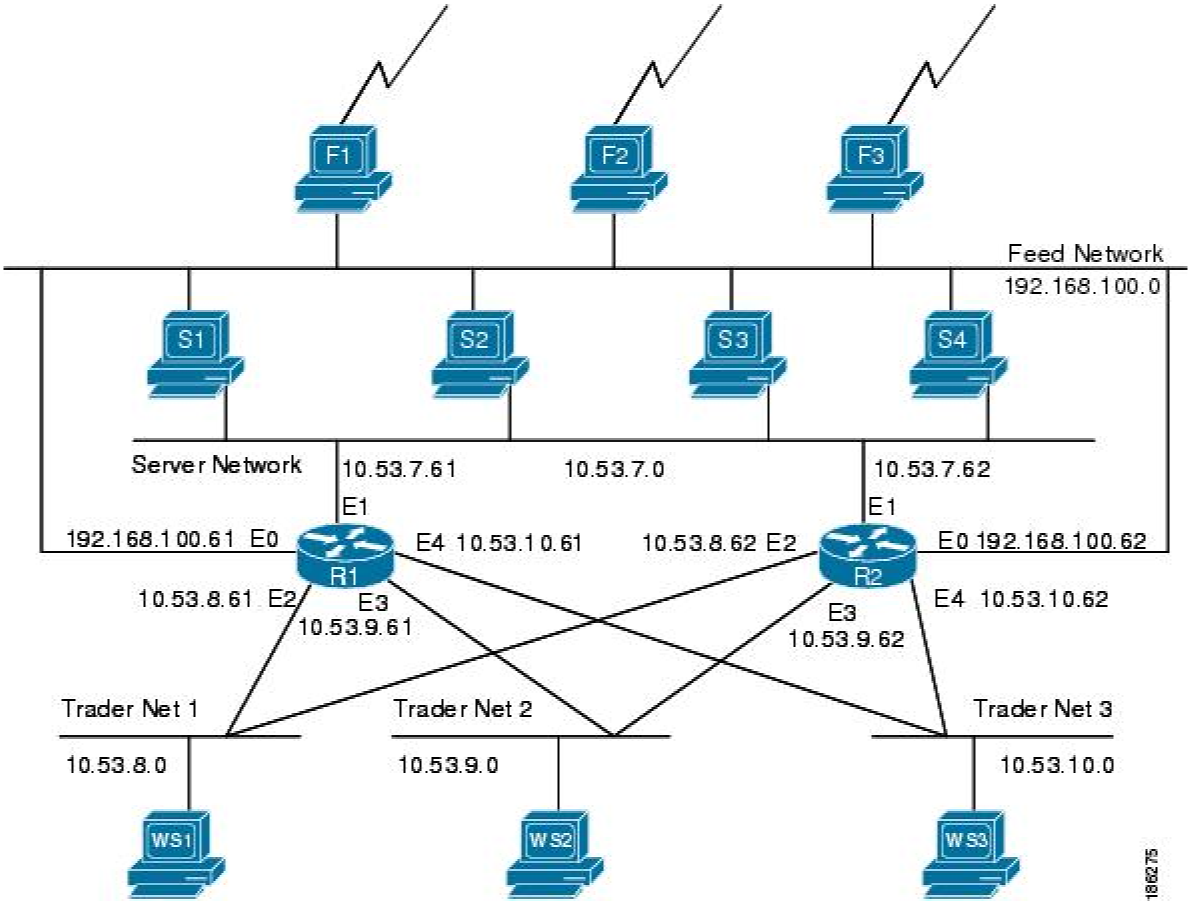
Default IP Broadcast Address

The Cisco IOS software supports sending IP broadcasts on both LANs and WANs. There are several ways to indicate an IP broadcast address. The default is an address consisting of all ones (255.255.255.255), although the software can be configured to generate any form of IP broadcast address such as all zeros (0.0.0.0), and directed broadcasts such as 172.16.255.255. Cisco IOS software can receive and process most IP broadcast addresses.

UDP Broadcast Packet Case Study

This case study is from a trading floor application in a financial company. The workstations (WS1, WS2, and WS3) in the following figure receive financial data from the feed network. The financial data is sent using UDP broadcasts.

Figure 2: Topology that Requires UDP Broadcast Forwarding

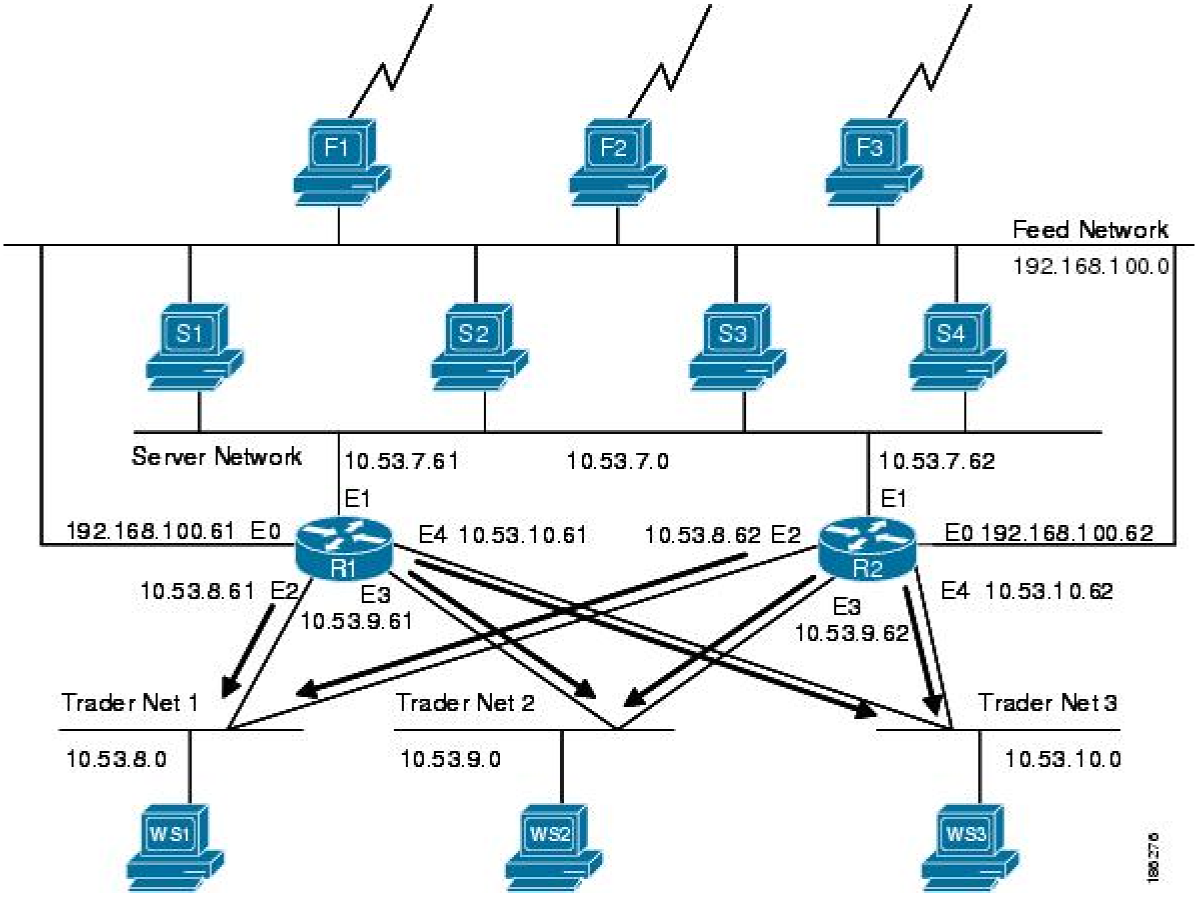


The following sections explain the possible solutions for this application:

UDP Broadcast Packet Forwarding

The first option is UDP broadcast packet using helper addresses. To configure helper addressing, you must specify the **ip helper-address** command on every interface on every router that receives a UDP broadcast that needs to be forwarded. On router 1 and router 2 in the figure below, IP helper addresses can be configured to move data from the server network to the trader networks. However IP helper addressing was determined not to be an optimal solution for this type of topology because each router receives unnecessary broadcasts from the other router, as shown in the figure below.

Figure 3: Flow of UDP Packets



In this case, router 1 receives each broadcast sent by router 2 three times, one for each segment, and router 2 receives each broadcast sent by router 1 three times, one for each segment. When each broadcast is received, the router must analyze it and determine that the broadcast does not need to be forwarded. As more segments are added to the network, the routers become overloaded with unnecessary traffic, which must be analyzed and discarded.

When IP helper addressing is used in this type of topology, no more than one router can be configured to forward UDP broadcasts (unless the receiving applications can handle duplicate broadcasts). This is because duplicate packets arrive on the trader network. This restriction limits redundancy in the design and can be undesirable in some implementations.

To configure routers to send UDP broadcasts bidirectionally in this type of topology, a second **ip helper address** command must be applied to every router interface that receives UDP broadcasts. As more segments and devices are added to the network, more **ip helper address** commands are required to reach them, so the administration of these routers becomes more complex over time.

Note Bidirectional traffic in this topology significantly impacts router performance.

Although IP helper addressing is well-suited to nonredundant, nonparallel topologies that do not require a mechanism for controlling broadcast loops, IP helper addressing does not work well in this topology. To improve performance, the network designers considered four other alternatives:

- Setting the broadcast address on the servers to all ones (255.255.255.255)—This alternative was dismissed because the servers have more than one interface, causing server broadcasts to be sent back onto the feed network. In addition, some workstation implementations do not allow all 1s broadcasts when multiple interfaces are present.
- Setting the broadcast address of the servers to the major network broadcast IP address--This alternative was dismissed because the TCP/IP implementation on the servers does not allow the use of major network IP broadcast addresses when the network is subnetted.
- Eliminating the subnets and letting the workstations use Address Resolution Protocol (ARP) to learn addresses—This alternative was dismissed because the servers cannot quickly learn an alternative route in the event of a primary router failure.
- UDP broadcast packet flooding—This alternative uses the spanning-tree topology created with transparent bridging to forward UDP broadcast packets in a redundant topology while avoiding loops and duplicate broadcast traffic.

UDP Broadcast Packet Flooding

UDP flooding uses the spanning-tree algorithm to forward packets in a controlled manner. Bridging is enabled on each router interface for the sole purpose of building the spanning tree. The spanning tree prevents loops by stopping a broadcast from being forwarded out an interface on which the broadcast was received. The spanning tree also prevents packet duplication by placing certain interfaces in the blocked state (so that no packets are forwarded) and other interfaces in the forwarding state (so that packets that need to be forwarded are forwarded).

Before you can enable UDP flooding, the router must be running software that supports transparent bridging and bridging must be configured on each interface that is to participate in the flooding. If bridging is not configured for an interface, the interface will receive broadcasts, but the router will not forward those broadcasts and will not use that interface as a destination for sending broadcasts received on a different interface.

When configured for UDP flooding, the router uses the destination address specified by the **ip broadcast-address** command on the output interface to assign a destination address to a flooded UDP datagram. Thus, the destination address might change as the datagram propagates through the network. The source address, however, does not change.

With UDP flooding, both routers shown in the figure below use a spanning-tree to control the network topology for the purpose of forwarding broadcasts. The **bridge protocol** command can specify either the **dec** keyword (for the Digital Equipment Corporation (DEC) spanning-tree protocol) or the **ieee** keyword (for the IEEE Ethernet protocol). All routers in the network must enable the same spanning-tree protocol. The **ip forward-protocol spanning-tree** command uses the database created by the **bridge protocol** command. Only one broadcast packet arrives at each segment, and UDP broadcasts can traverse the network in both directions.

Because bridging is enabled only to build the spanning-tree database, use access lists to prevent the spanning-tree from forwarding non-UDP traffic.

The router configuration specifies a path cost for each interface to determine which interface forwards or blocks packets. The default path cost for Ethernet is 100. Setting the path cost for each interface on router 2 to 50 causes the spanning-tree algorithm to place the interfaces in router 2 in forwarding state. Given the higher path cost (100) for the interfaces in router 1, the interfaces in router 1 are in the blocked state and do not forward the broadcasts. With these interface states, broadcast traffic flows through router 2. If router 2 fails, the spanning-tree algorithm will place the interfaces in router 1 in the forwarding state, and router 1 will forward broadcast traffic.

With one router forwarding broadcast traffic from the server network to the trader networks, you should configure the other router to forward unicast traffic. For that reason, each router enables the ICMP Router Discovery Protocol (IRDP), and each workstation on the trader networks runs the IRDP daemon. On router 1, the **preference** keyword of the **ip irdp** command sets a higher IRDP preference than does the configuration for router 2, which causes each IRDP daemon to use router 1 as its preferred default gateway for unicast traffic forwarding. Users of those workstations can use the **netstat -rn** command to see how the routers are being used.

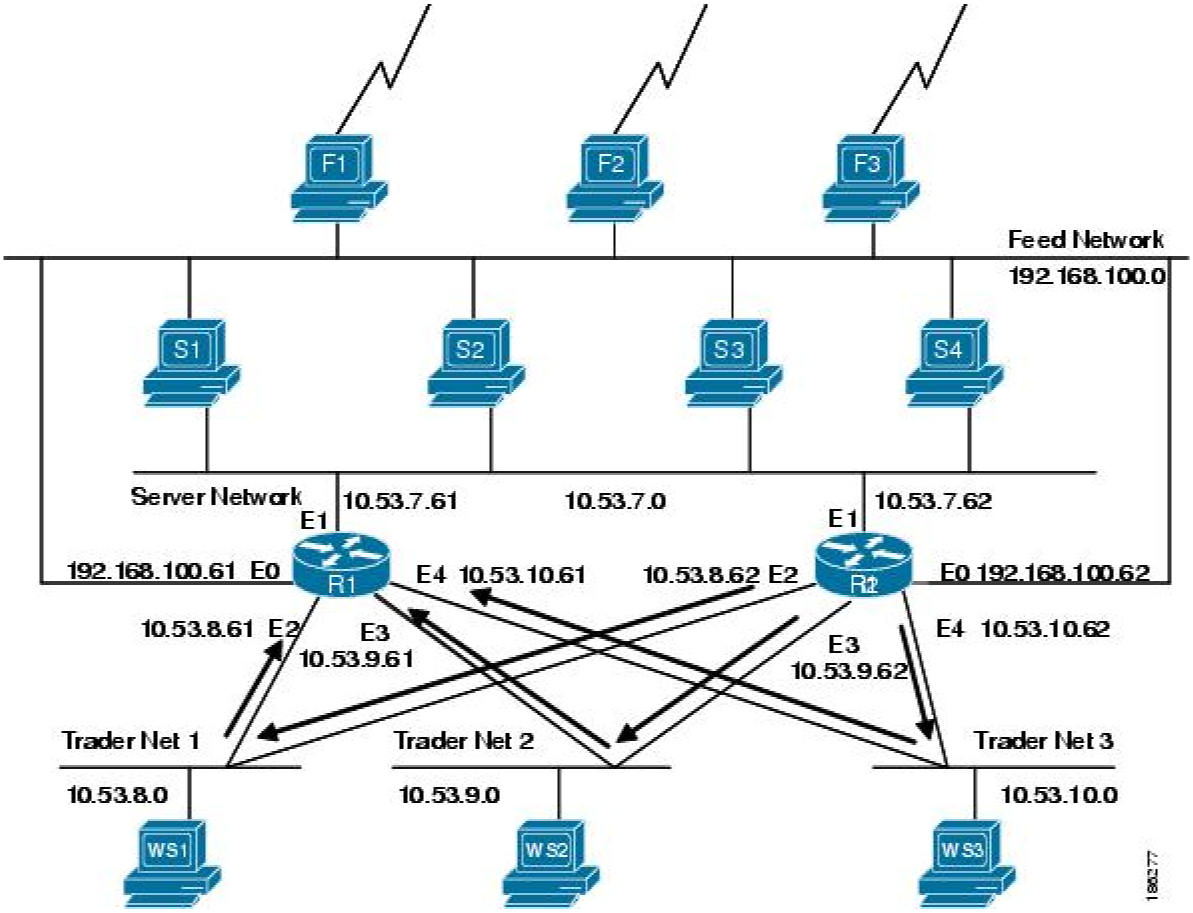
On the routers, the **holdtime**, **maxadvertinterval**, and **minadvertinterval** keywords of the **ip irdp** command reduce the advertising interval from the default so that the IRDP daemons running on the hosts expect to see advertisements more frequently. With the advertising interval reduced, the workstations will adopt router 2 more quickly if router 1 becomes unavailable. With this configuration, when a router becomes unavailable, IRDP offers a convergence time of less than one minute.

IRDP is preferred over the Routing Information Protocol (RIP) and default gateways for the following reasons:

- RIP takes longer to converge.
- Configuration of router 1 as the default gateway on each Sun workstation on the trader networks would allow those Sun workstations to send unicast traffic to router 1, but would not provide an alternative route if router 1 becomes unavailable.

The figure below shows how data flows when the network is configured for UDP flooding.

Figure 4: Data Flow with UDP Flooding and IRDP



Note This topology is broadcast intensive--broadcasts sometimes consume 20 percent of the 10-MB Ethernet bandwidth. However, this is a favorable percentage when compared to the configuration of IP helper addressing, which, in the same network, causes broadcasts to consume up to 50 percent of the 10-MB Ethernet bandwidth.

If the hosts on the trader networks do not support IRDP, Hot Standby Routing Protocol (HSRP), Virtual Router Redundancy Protocol (VRRP), or Gateway Load Balancing Protocol (GLBP) can be used to select which router will handle unicast traffic. These protocols allow the standby router to take over quickly if the primary router becomes unavailable.

Enable turbo flooding on the routers to increase the performance of UDP flooding.

Note Turbo flooding increases the amount of processing that is done at interrupt level, which increases the CPU load on the router. Turbo flooding may not be appropriate on routers that are already under high CPU load or that must also perform other CPU-intensive activities.

How to Configure IP Broadcast Packet Handling

Enabling IP Directed Broadcasts Without an Access List

Perform this task to permit the forwarding of IP directed broadcasts from any source.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *address mask*
5. **ip directed-broadcast**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/1`	Specifies an interface and enters interface configuration mode.

	Command or Action	Purpose
Step 4	**ip address** *address mask* **Example:** `Router(config-if)# ip address 172.16.10.1 255.255.255.0`	Assigns an IP address to the interface.
Step 5	**ip directed-broadcast** **Example:** `Router(config-if)# ip directed-broadcast`	Enables IP directed broadcasts on the interface. • Configure this command on the interface that is connected to the IP network address of the directed broadcast packets. • In this example the directed broadcast packets are addressed to 172.16.10.255.

Step 6	**end**
	Example:
	`Router(config-if)# end`
		Exits the current configuration mode and returns to privileged EXEC mode.

Enabling IP Directed Broadcasts with an Access List

Perform this task to limit the forwarding of IP directed broadcasts by applying an access list to the **ip directed-broadcast** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **access-list** *100-199* **permit ip** *source-address mask destination-address mask*
4. **interface** *type number*
5. **ip address** *address mask*
6. **ip directed-broadcast** *access-list*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *100-199* **permit ip** *source-address mask destination-address mask* **Example:** `Router(config)# access-list 100 permit ip 10.4.9.167 0.0.0.0 172.16.10.0 0.0.0.255`	Creates an access list to limit the IP directed broadcasts that are forwarded. • In this example the IP directed broadcasts are sent by the host with the IP address of 10.4.9.167 to the IP directed broadcast address 172.16.10.255.
Step 4	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0`	Specifies an interface and enters interface configuration mode.
Step 5	**ip address** *address mask* **Example:** `Router(config-if)# ip address 172.16.10.1 255.255.255.0`	Assigns an IP address to the interface.
Step 6	**ip directed-broadcast** *access-list* **Example:** `Router(config-if)# ip directed-broadcast 100`	Enables IP directed broadcasts on the interface for broadcast packets that are allowed by the access list you assigned. Configure this command on the interface that is connected to the IP network address of the directed broadcast packets. • In this example the directed broadcast packets are addressed to 172.16.10.255.

Step 7	**end**
	Example:
	`Router(config-if)# end`
		Exits the current configuration mode and returns to privileged EXEC mode.

Enabling Forwarding of UDP Broadcast Packets to a Specific Host

Perform this task to enable UDP broadcast packet forwarding to a single host.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip forward-protocol udp**
4. **interface** *type number*
5. **ip address** *address mask*
6. **ip helper-address** *address*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip forward-protocol udp** **Example:** `Router(config)# ip forward-protocol udp`	Enables forwarding of UDP broadcast packets.
Step 4	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/1`	Specifies an interface and enters interface configuration mode.
Step 5	**ip address** *address mask* **Example:** `Router(config-if)# ip address 172.16.10.1 255.255.255.0`	Assigns an IP address to the interface.

	Command or Action	Purpose
Step 6	**ip helper-address** *address* **Example:** `Router(config-if)# ip helper-address 172.16.10.2`	Enables an IP helper address for the interface that is receiving the UDP broadcast packets. • In this example the IP destination address of the IP UDP broadcast packets is rewritten to 172.16.10.2.

Step 7	**end**
	Example:
	`Router(config-if)# end`
		Exits the current configuration mode and returns to privileged EXEC mode.

Enabling Forwarding of UDP Broadcast Packets to a Range of Hosts

Perform this task to enable UDP broadcast packet forwarding to a range of hosts to allow for load sharing between the destination hosts and to provide redundancy if one or more of the destination hosts fail.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip forward-protocol udp**
4. **interface** *type number*
5. **ip address** *address mask*
6. **ip helper-address** *address*
7. **exit**
8. **interface** *type number*
9. **ip address** *address mask*
10. **ip directed-broadcast**
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip forward-protocol udp** **Example:** `Router(config)# ip forward-protocol udp`	Enables forwarding of UDP broadcast packets.
Step 4	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0`	Specifies an interface and enters interface configuration mode.
Step 5	**ip address** *address mask* **Example:** `Router(config-if)# ip address 192.168.10.1 255.255.255.0`	Assigns an IP address to the interface.
Step 6	**ip helper-address** *address* **Example:** `Router(config-if)# ip helper-address 172.16.10.255`	Enables an IP helper address for the interface that is receiving the UDP broadcast packets. • In this example an IP directed broadcast address is used. The IP destination address of the IP UDP broadcast packets is rewritten to 172.16.10.255. • All of the hosts on the 172.16.10.0/24 network that support the application or service that the UDP broadcast packets are intended for will respond to the UDP broadcast packets. **Note** This often results in the source of the UDP broadcast packets receiving responses from two or more hosts. In most circumstances the source of the UDP broadcast packets accepts the first response and ignores any subsequent responses. In some situations the source of the UDP broadcast packets cannot handle duplicate responses and reacts by reloading, or other unexpected behavior.
Step 7	**exit** **Example:** `Router(config-if)# exit`	Returns to global configuration mode.

	Command or Action	Purpose
Step 8	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/1`	Specifies an interface and enters interface configuration mode.
Step 9	**ip address** *address mask* **Example:** `Router(config-if)# ip address 172.16.10.1 255.255.255.0`	Assigns an IP address to the interface.
Step 10	**ip directed-broadcast** **Example:** `Router(config-if)# ip directed-broadcast`	Enables IP directed broadcasts on the interface that is transmitting the UDP broadcasts.

Step 11	**end**
	Example:
	`Router(config-if)# end`
		Exits the current configuration mode and returns to privileged EXEC mode.

Changing the Default IP Broadcast Address for All Interfaces to 0.0.0.0 on Routers Without Nonvolatile Memory

If you router does not have NVRAM, and you need to change the IP broadcast address to 0.0.0.0, you must change the IP broadcast address manually by setting jumpers in the processor configuration register. Setting bit 10 causes the device to use all 0s. Bit 10 interacts with bit 14, which controls the network and host portions of the broadcast address. Setting bit 14 causes the device to include the network and host portions of its address in the broadcast address. The table below shows the combined effect of setting bits 10 and 14.

Table 5: Configuration Register Settings for Broadcast Address Destination

Bit 14	Bit 10	Address (<net><host>)
Out	Out	<ones><ones>
Out	In	<zeros><zeros>
In	In	<net><zeros>
In	Out	<net><ones>

For additional information on setting the hardware jumpers on your router, see the hardware documentation that was supplied with you router.

Changing the Default IP Broadcast Address for All Interfaces to 0.0.0.0 on Routers with Nonvolatile Memory

Cisco IOS-based routers with NVRAM have software configuration registers that allow you to modify several behaviors of the router such as where it looks for images to load, what IP broadcast address it uses, and the console line speed. The factory default value for the configuration register is 0x2102 where *0X* indicates this a hexadecimal number. The **config-register** command is used to modify the settings of the software configuration registers.

Information on configuring other behaviors with the software configuration registers using the **config-register** command is available in the following documentation:

- "Loading and Managing System Images" chapter of the *Cisco IOS Configuration Fundamentals Configuration Guide*
- *Cisco IOS Configuration Fundamentals Command Reference*

Caution You need to be very careful when you change the software configuration registers on your router because if you inadvertently alter the console port line speed, you will not be able to configure the router with a terminal server on the console port unless you know the speed that you set for the console port, and you know how to change the line speed for your terminal application. If your router is configured for alternate access to the CLI such as using Telnet or a web browser, you can use this method to log in to the router and change the software configuration register back to 0x2102.

Perform this task to set the IP broadcast address on every interface to 0.0.0.0 while maintaining the remainder of the default values for the software configuration register settings.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **config-register** *value*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**config-register** *value* **Example:** `Router(config)# config-register 0x2502`	Sets the IP broadcast address to 0.0.0.0 on every interface while maintaining the remainder of the default values for the other software configuration register settings.

Step 4	**end**
	Example:
	`Router(config)# end`
		Exits the current configuration mode and returns to privileged EXEC mode.

Changing the IP Broadcast Address to Any IP Address on One or More Interfaces in a Router

Perform this task if you network requires an IP broadcast address other than 255.255.255.255 or 0.0.0.0, or you want to change the IP broadcast address to 0.0.0.0 on a subset of the interfaces on the router instead of on all of the interfaces on the router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip address** *address mask*
5. **ip broadcast-address** *address*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/1`	Specifies an interface and enters interface configuration mode.
Step 4	**ip address** *address mask* **Example:** `Router(config-if)# ip address 172.16.10.1 255.255.255.0`	Assigns an IP address to the interface.
Step 5	**ip broadcast-address** *address* **Example:** `Router(config-if)# ip broadcast-address 172.16.10.255`	Specifies the IP broadcast address • In this example IP broadcasts are sent to 172.16.10.255.

Step 6	**end**
	Example:
	`Router(config-if)# end`
		Exits the current configuration mode and returns to privileged EXEC mode.

Configuring UDP Broadcast Packet Flooding

Before You Begin

The version of Cisco IOS software on your router must support transparent bridging.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **bridge** *number* **protocol ieee**
4. **ip forward-protocol spanning-tree**
5. **ip forward-protocol turbo-flood**
6. **ip forward-protocol udp**
7. **interface** *type number*
8. **ip address** *address mask*
9. **bridge-group** *number*
10. **interface** *type number*
11. **ip address** *address mask*
12. **bridge-group** *number*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**bridge** *number* **protocol ieee** **Example:** `Router(config)# bridge 1 protocol ieee`	Enables spanning-tree bridging and specifies the bridging protocol.
Step 4	**ip forward-protocol spanning-tree** **Example:** `Router(config)# ip forward-protocol spanning-tree`	Enables using the spanning-tree forwarding table to flood broadcast packets.
Step 5	**ip forward-protocol turbo-flood** **Example:** `Router(config)# ip forward-protocol turbo-flood`	(Optional) Enables fast forwarding of broadcast packets using the spanning-tree forwarding table.

	Command or Action	Purpose
Step 6	**ip forward-protocol udp** **Example:** `Router(config)# ip forward-protocol udp`	Enables forwarding of UDP broadcasts.
Step 7	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0`	Specifies an interface and enters interface configuration mode.
Step 8	**ip address** *address mask* **Example:** `Router(config-if)# ip address 192.168.10.1 255.255.255.0`	Assigns an IP address to the interface.
Step 9	**bridge-group** *number* **Example:** `Router(config-if)# bridge-group 1`	Places the interface in the spanning-tree bridge group specified.
Step 10	**interface** *type number* **Example:** `Router(config-if)# interface fastethernet 0/1`	Specifies an interface and enters interface configuration mode.
Step 11	**ip address** *address mask* **Example:** `Router(config-if)# ip address 172.16.10.1 255.255.255.0`	Assigns an IP address to the interface.
Step 12	**bridge-group** *number* **Example:** `Router(config-if)# bridge-group 1`	Places the interface in the spanning-tree bridge group specified.

Step 13	**end**
	Example:
	`Router(config-if)# end`
		Exits the current configuration mode and returns to privileged EXEC mode.

Configuration Examples for IP Broadcast Packet Handling

Example: Enabling IP Directed Broadcasts with an Access List

The following example shows how to enable IP directed broadcasts with an access list to control the directed broadcasts that are forwarded.

```
Router(config)# access-list 100 permit ip 10.4.9.167 0.0.0.0 172.16.10.0 0.0.0.255
Router(config)# interface fastethernet 0/0
Router(config-if)# ip address 172.16.10.1 255.255.255.0
Router(config-if)# ip directed-broadcast 100
```

Example: Configuring UDP Broadcast Packet Flooding

```
Router(config)# bridge 1 protocol ieee
Router(config)# ip forward-protocol spanning-tree
Router(config)# ip forward-protocol turbo-flood
Router(config)# ip forward-protocol udp
Router(config)# interface fastethernet 0/0
Router(config-if)# ip address 192.168.10.1 255.255.255.0
Router(config-if)# bridge-group 1
Router(config)# interface fastethernet 0/1
Router(config-if)# ip address 172.16.10.1 255.255.255.0
Router(config-if)# bridge-group 1
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Currently assigned IP multicast addresses	*Internet Multicast Addresses* http://www.iana.org/assignments/multicast-addresses
Configuration fundamentals configuration tasks	*Cisco IOS Configuration Fundamentals Configuration Guide*
Configuration fundamentals commands	*Cisco IOS Configuration Fundamentals Command Reference*
Cisco IOS bridging and IBM networking configuration tasks	*Cisco IOS Bridging and IBM Networking Configuration Guide*
Cisco IOS bridging and IBM networking commands	*Cisco IOS Bridging and IBM Networking Command Reference*

Related Topic	Document Title
Cisco IOS IP multicast configuration tasks	*Cisco IOS IP Multicast Configuration Guide*
Cisco IOS IP Multicast commands	*Cisco IOS IP Multicast Command Reference*

Standards

Standard	Title
IEEE Spanning-Tree Bridging	802.1D MAC Bridges http://www.ieee802.org/1/pages/802.1D-2003.html

MIBs

MIB	MIBs Link
—	No new or modified MIBs are supported, and support for existing MIBs has not been modified.

RFCs

RFC	Title
RFC 1812	*Requirements for IP Version 4 Routers* http://www.ietf.org/rfc/rfc1812.txt
RFC 2131	*Dynamic Host Configuration Protocol* http://www.ietf.org/rfc/rfc2131.txt .

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for IP Broadcast Packet Handling

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 6: Feature Information for IP Broadcast Packet Handling

Feature Name	Releases	Feature Information
Flooding Packets Using spanning-tree	15.0(1)SY	Enables the forwarding of UDP broadcast packets using the spanning-tree forwarding table. The following commands were introduced or modified by this feature: **ip forward-protocol spanning-tree**, **ip forward-protocol turbo-flood**.
IP Directed Broadcasts	15.0(1)SY	Enables the translation of a directed broadcast to physical broadcasts. The following command was introduced or modified by this feature: **ip directed-broadcast**.
Specifying an IP Broadcast Address	15.0(1)SY	Specifies the IP broadcast address for an interface. The following command was introduced or modified by this feature: **ip broadcast-address**.
UDP Broadcast Packet Forwarding	15.0(1)SY	Enables the forwarding of UDP broadcast packets. The following commands were introduced or modified by this feature: **ip forward-protocol**, **ip helper-address**.

CHAPTER 4

Configuring TCP

TCP is a protocol that specifies the format of data and acknowledgments used in data transfer. TCP is a connection-oriented protocol because participants must establish a connection before data can be transferred. By performing flow control and error correction, TCP guarantees reliable, in-sequence delivery of packets. TCP is considered a reliable protocol because it will continue to request an IP packet that is dropped or received out of order until it is received. This module explains concepts related to TCP and how to configure TCP in a network.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for TCP

TCP Time Stamp, TCP Selective Acknowledgment, and TCP Header Compression

Because TCP time stamps are always sent and echoed in both directions and the time-stamp value in the header is always changing, TCP header compression will not compress the outgoing packet. To allow TCP header compression over a serial link, the TCP time-stamp option is disabled. If you want to use TCP header compression over a serial line, TCP time stamp and TCP selective acknowledgment must be disabled. Both features are disabled by default. Use the **no ip tcp selective-ack** command to disable the TCP selective acknowledgment once it is enabled.

Information About TCP

TCP Services

TCP provides reliable transmission of data in an IP environment. TCP corresponds to the transport layer (Layer 4) of the Open Systems Interconnection (OSI) reference model. Among the services that TCP provides are stream data transfer, reliability, efficient flow control, full-duplex operation, and multiplexing.

With stream data transfer, TCP delivers an unstructured stream of bytes that are identified by sequence numbers. This service benefits applications because they do not have to divide data into blocks before handing it off to TCP. Instead, TCP groups bytes into segments and passes them to IP for delivery.

TCP offers reliability by providing connection-oriented, end-to-end reliable packet delivery through an internetwork. It does this by sequencing bytes with a forwarding acknowledgment number that indicates to the destination the next byte that the source expects to receive. Bytes that are not acknowledged within a specified time period are retransmitted. The reliability mechanism of TCP allows devices to handle lost, delayed, duplicate, or misread packets. A timeout mechanism allows devices to detect lost packets and request retransmission.

TCP offers efficient flow control, which means that the receiving TCP process indicates the highest sequence number that it can receive without overflowing its internal buffers when sending acknowledgments back to the source.

TCP offers full-duplex operation, and TCP processes can both send and receive data at the same time.

TCP multiplexing allows numerous simultaneous upper-layer conversations to be multiplexed over a single connection.

TCP Sliding Window

A TCP sliding window provides an efficient use of network bandwidth because it enables hosts to send multiple bytes or packets before waiting for an acknowledgment.

In TCP, the receiver specifies the current window size in every packet. Because TCP provides a byte-stream connection, window sizes are expressed in bytes. A window is the number of data bytes that the sender is allowed to send before waiting for an acknowledgment. Initial window sizes are indicated at connection setup, but might vary throughout the data transfer to provide flow control. A window size of zero means “Send no data.” The default TCP window size is 4128 bytes. We recommend that you keep the default value unless your device is sending large packets (greater than 536 bytes). Use the **ip tcp window-size** command to change the default window size.

In a TCP sliding-window operation, for example, the sender might have a sequence of bytes to send (numbered 1 to 10) to a receiver who has a window size of five. The sender then places a window around the first five bytes and transmits them together. The sender then waits for an acknowledgment.

The receiver responds with an ACK = 6, indicating that it has received bytes 1 to 5 and is expecting byte 6 next. In the same packet, the receiver indicates that its window size is 5. The sender then moves the sliding window five bytes to the right and transmits bytes 6 to 10. The receiver responds with an ACK = 11, indicating that it is expecting sequenced byte 11 next. In this packet, if the receiver indicates that its window size is 0, the sender cannot send any more bytes until the receiver sends another packet with a window size greater than 0.

TCP Connection Establishment

To use reliable transport services, TCP hosts must establish a connection-oriented session with one another. Connection establishment is performed by using a “three-way handshake” mechanism.

A three-way handshake synchronizes both ends of a connection by allowing both sides to agree upon the initial sequence numbers. This mechanism guarantees that both sides are ready to transmit data. The three-way handshake is necessary so that packets are not transmitted or retransmitted during session establishment or after session termination.

Each host randomly chooses a sequence number, which is used to track bytes within the stream that the host is sending. The three-way handshake proceeds in the following manner:

- The first host (Host A) initiates a connection by sending a packet with the initial sequence number (X) and the synchronize/start (SYN) bit set to indicate a connection request.
- The second host (Host B) receives the SYN, records the sequence number X, and replies by acknowledging (ACK) the SYN (with an ACK = X + 1). Host B includes its own initial sequence number (SEQ = Y). An ACK = 20 means that the host has received bytes 0 through 19 and expects byte 20 next. This technique is called forward acknowledgment.
- Host A acknowledges all bytes that Host B has sent with a forward acknowledgment indicating the next byte Host A expects to receive (ACK = Y + 1). Data transfer can then begin.

TCP Connection Attempt Time

You can set the amount of time the software will wait before attempting to establish a TCP connection. The connection attempt time is a host parameter and pertains to traffic that originated at the device and not to traffic going through the device. To set the TCP connection attempt time, use the **ip tcp synwait-time** command in global configuration mode. The default is 30 seconds.

TCP Selective Acknowledgment

The TCP Selective Acknowledgment feature improves performance if multiple packets are lost from one TCP window of data.

Prior to this feature, because of limited information available from cumulative acknowledgments, a TCP sender could learn about only one lost packet per-round-trip time. An aggressive sender could choose to resend packets early, but such re-sent segments might have already been successfully received.

The TCP selective acknowledgment mechanism helps improve performance. The receiving TCP host returns selective acknowledgment packets to the sender, informing the sender of data that has been received. In other words, the receiver can acknowledge packets received out of order. The sender can then resend only missing data segments (instead of everything since the first missing packet).

Prior to selective acknowledgment, if TCP lost packets 4 and 7 out of an 8-packet window, TCP would receive acknowledgment of only packets 1, 2, and 3. Packets 4 through 8 would need to be re-sent. With selective acknowledgment, TCP receives acknowledgment of packets 1, 2, 3, 5, 6, and 8. Only packets 4 and 7 must be re-sent.

TCP selective acknowledgment is used only when multiple packets are dropped within one TCP window. There is no performance impact when the feature is enabled but not used. Use the **ip tcp selective-ack** command in global configuration mode to enable TCP selective acknowledgment.

Refer to RFC 2018 for more details about TCP selective acknowledgment.

TCP Time Stamp

The TCP time-stamp option provides improved TCP round-trip time measurements. Because the time stamps are always sent and echoed in both directions and the time-stamp value in the header is always changing, TCP header compression will not compress the outgoing packet. To allow TCP header compression over a serial link, the TCP time-stamp option is disabled. Use the **ip tcp timestamp** command to enable the TCP time-stamp option.

Refer to RFC 1323 for more details on TCP time stamps.

TCP Maximum Read Size

The maximum number of characters that TCP reads from the input queue for Telnet and relogin at one time is very large (the largest possible 32-bit positive number) by default. To change the TCP maximum read size value, use the **ip tcp chunk-size** command in global configuration mode.

Note We do not recommend that you change this value.

TCP Path MTU Discovery

Path MTU Discovery is a method for maximizing the use of the available bandwidth in the network between endpoints of a TCP connection, which is described in RFC 1191. IP Path MTU Discovery allows a host to dynamically discover and cope with differences in the maximum allowable maximum transmission unit (MTU) size of the various links along the path. Sometimes a device is unable to forward a datagram because it requires fragmentation (the packet is larger than the MTU that you set for the interface with the **interface** configuration command), but the “do not fragment” (DF) bit is set. The intermediate gateway sends a “Fragmentation needed and DF bit set” Internet Control Message Protocol (ICMP) message to the sending host, alerting the host to the problem. On receiving this message, the host reduces its assumed path MTU and consequently sends a smaller packet that will fit the smallest packet size of all links along the path.

By default, TCP Path MTU Discovery is disabled. Existing connections are not affected irrespective of whether this feature is enabled or disabled.

Customers using TCP connections to move bulk data between systems on distinct subnets would benefit most by enabling this feature. Customers using remote source-route bridging (RSRB) with TCP encapsulation, serial tunnel (STUN), X.25 Remote Switching (also known as XOT or X.25 over TCP), and some protocol translation configurations might also benefit from enabling this feature.

Use the **ip tcp path-mtu-discovery** global configuration command to enable Path MTU Discovery for connections initiated by the device when the device is acting as a host.

For more information about Path MTU Discovery, refer to the “Configuring IP Services” module of the *IP Application Services Configuration Guide*.

TCP Outgoing Queue Size

The default TCP outgoing queue size per connection is five segments if the connection has a TTY associated with it (such as a Telnet connection). If no TTY connection is associated with a connection, the default queue size is 20 segments. Use the **ip tcp queuemax** command to change the five-segment default value.

TCP MSS Adjustment

The TCP MSS Adjustment feature enables the configuration of the maximum segment size (MSS) for transient packets that traverse a device, specifically TCP segments with the SYN bit set. Use the **ip tcp adjust-mss** command in interface configuration mode to specify the MSS value on the intermediate device of the SYN packets to avoid truncation.

When a host (usually a PC) initiates a TCP session with a server, the host negotiates the IP segment size by using the MSS option field in the TCP SYN packet. The value of the MSS field is determined by the MTU configuration on the host. The default MSS value for a PC is 1500 bytes.

The PPP over Ethernet (PPPoE) standard supports a Maximum Transmission Unit (MTU) of only 1492 bytes. The disparity between the host and PPPoE MTU size can cause the device in between the host and the server to drop 1500-byte packets and terminate TCP sessions over the PPPoE network. Even if the path MTU (which detects the correct MTU across the path) is enabled on the host, sessions may be dropped because system administrators sometimes disable ICMP error messages that must be relayed from the host for path MTU to work.

The **ip tcp adjust-mss** command helps prevent TCP sessions from being dropped by adjusting the MSS value of the TCP SYN packets.

The **ip tcp adjust-mss** command is effective only for TCP connections passing through the device.

In most cases, the optimum value for the *max-segment-size* argument of the **ip tcp adjust-mss** command is 1452 bytes. This value plus the 20-byte IP header, the 20-byte TCP header, and the 8-byte PPPoE header add up to a 1500-byte packet that matches the MTU size for the Ethernet link.

See the “Configuring the MSS Value and MTU for Transient TCP SYN Packets” section for configuration instructions.

IPv6 TCP Traffic

Due to the differences in the network layer (IP) headers between IPv4 and IPv6, extra overhead such as tunnel headers may be added during the IPv6 traffic path and this may cause IP fragmentation. IPv6 path MTU (PMTU) detects the MTU and then the sender does IPv6 fragmentation.

The **ipv6 tcp adjust-mss** command allows the TCP MSS Adjustment feature to be enabled on IPv6 traffic.

TCP MIB for RFC 4022 Support

The TCP MIB for RFC 4022 Support feature introduces support for RFC 4022, *Management Information Base for the Transmission Control Protocol (TCP)*. RFC 4022 is an incremental change of the TCP MIB to improve the manageability of TCP.

To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL:

http://www.cisco.com/go/mibs

How to Configure TCP

Configuring TCP Performance Parameters

Before You Begin

Both sides of the network link must be configured to support window scaling or the default of 65,535 bytes will be applied as the maximum window size. To support Explicit Congestion Notification (ECN), the remote peer must be ECN-enabled because the ECN capability is negotiated during a three-way handshake with the remote peer.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip tcp synwait-time** *seconds*
4. **ip tcp path-mtu-discovery** [**age-timer** {*minutes* | **infinite**}]
5. **ip tcp selective-ack**
6. **ip tcp timestamp**
7. **ip tcp chunk-size** *characters*
8. **ip tcp window-size** *bytes*
9. **ip tcp ecn**
10. **ip tcp queuemax** *packets*
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip tcp synwait-time** *seconds* **Example:** `Device(config)# ip tcp synwait-time 60`	(Optional) Sets the amount of time the Cisco software will wait before attempting to establish a TCP connection. • The default is 30 seconds.
Step 4	**ip tcp path-mtu-discovery** [**age-timer** {*minutes* | **infinite**}] **Example:** `Device(config)# ip tcp path-mtu-discovery age-timer 11`	(Optional) Enables Path MTU Discovery. • **age-timer —**Time interval, in minutes, TCP reestimates the Maximum Transmission Unit (MTU) with a larger Maximum Segment Size (MSS). The default is 10 minutes. The maximum is 30 minutes. • **infinite—**Disables the age timer.
Step 5	**ip tcp selective-ack** **Example:** `Device(config)# ip tcp selective-ack`	(Optional) Enables TCP selective acknowledgment.
Step 6	**ip tcp timestamp** **Example:** `Device(config)# ip tcp timestamp`	(Optional) Enables the TCP time stamp.
Step 7	**ip tcp chunk-size** *characters* **Example:** `Device(config)# ip tcp chunk-size 64000`	(Optional) Sets the TCP maximum read size for Telnet or rlogin. **Note** We do not recommend that you change this value.
Step 8	**ip tcp window-size** *bytes* **Example:** `Device(config)# ip tcp window-size 75000`	(Optional) Sets the TCP window size. • The *bytes* argument can be set to an integer from 68 to 1073741823. To enable window scaling to support Long Flat Networks (LFNs), the TCP window size must be more than 65535. The default window size is 4128 if window scaling is not configured.

	Command or Action	Purpose
		Note With CSCsw45317, the *bytes* argument can be set to an integer from 68 to 1073741823.
Step 9	**ip tcp ecn** **Example:** `Device(config)# ip tcp ecn`	(Optional) Enables ECN for TCP.
Step 10	**ip tcp queuemax** *packets* **Example:** `Device(config)# ip tcp queuemax 10`	(Optional) Sets the TCP outgoing queue size.

Step 11	**end**
	Example:
	`Device(config)# end`
		Exits to privileged EXEC mode.

Configuring the MSS Value and MTU for Transient TCP SYN Packets

Perform this task to configure the maximum size segment (MSS) for transient packets that traverse a device, specifically TCP segments with the SYN bit set, and to configure the MTU size of IP packets.

If you are configuring the **ip mtu** command on the same interface as the **ip tcp adjust-mss** command, we recommend that you use the following commands and values:

- **ip tcp adjust-mss 1452**
- **ip mtu 1492**

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip tcp adjust-mss** *max-segment-size*
5. **ip mtu** *bytes*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/0`	Configures an interface type and enters interface configuration mode.
Step 4	**ip tcp adjust-mss** *max-segment-size* **Example:** `Device(config-if)# ip tcp adjust-mss 1452`	Adjusts the MSS value of TCP SYN packets going through a device. • The *max-segment-size* argument is the maximum segment size, in bytes. The range is from 500 to 1460.
Step 5	**ip mtu** *bytes* **Example:** `Device(config-if)# ip mtu 1492`	Sets the MTU size of IP packets, in bytes, sent on an interface.

Step 6	**end**
	Example:
	`Device(config-if)# end`
		Exits to global configuration mode.

Configuring the MSS Value for IPv6 Traffic

Perform this task to configure the maximum size segment (MSS) for transient packets that traverse a device, specifically TCP segments with the DF bit set in IPv6 network layer (IP) header.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 tcp adjust-mss** *max-segment-size*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 1/0/0`	Configures an interface type and enters interface configuration mode.
Step 4	**ipv6 tcp adjust-mss** *max-segment-size* **Example:** `Device(config-if)# ipv6 tcp adjust-mss 1452`	Adjusts the MSS value of TCP DF packets going through a device. • The *max-segment-size* argument is the maximum segment size, in bytes. The range is from 40 to 1940.

Step 5	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Verifying TCP Performance Parameters

SUMMARY STEPS

1. **show tcp** [*line-number*] [**tcb** *address*]
2. **show tcp brief** [**all** | **numeric**]
3. **debug ip tcp transactions**
4. **debug ip tcp congestion**

DETAILED STEPS

Step 1 **show tcp** [*line-number*] [**tcb** *address*]

Displays the status of TCP connections. The arguments and keyword are as follows:

- *line-number*—(Optional) Absolute line number of the Telnet connection status.

- **tcb—**(Optional) Transmission control block (TCB) of the Explicit Congestion Notification (ECN)-enabled connection.
- *address***—**(Optional) TCB hexadecimal address. The valid range is from 0x0 to 0xFFFFFFFF.

The following sample output from the **show tcp tcb** command displays detailed information about an ECN-enabled connection that uses a hexadecimal address format:

Example:

```
Device# show tcp tcb 0x62CD2BB8

Connection state is LISTEN, I/O status: 1, unread input bytes: 0
Connection is ECN enabled
Local host: 10.10.10.1, Local port: 179
Foreign host: 10.10.10.2, Foreign port: 12000
Enqueued packets for retransmit: 0, input: 0 mis-ordered: 0 (0 bytes)
Event Timers (current time is 0x4F31940):
Timer          Starts    Wakeups            Next
Retrans             0          0             0x0
TimeWait            0          0             0x0
AckHold             0          0             0x0
SendWnd             0          0             0x0
KeepAlive           0          0             0x0
GiveUp              0          0             0x0
PmtuAger            0          0             0x0
DeadWait            0          0             0x0
iss:          0 snduna:          0 sndnxt:          0     sndwnd:      0
irs:          0 rcvnxt:          0 rcvwnd:       4128  delrcvwnd:      0
SRTT: 0 ms, RTTO: 2000 ms, RTV: 2000 ms, KRTT: 0 ms
minRTT: 60000 ms, maxRTT: 0 ms, ACK hold: 200 ms
Flags: passive open, higher precedence, retransmission timeout
TCB is waiting for TCP Process (67)
Datagrams (max data segment is 516 bytes):
Rcvd: 6 (out of order: 0), with data: 0, total data bytes: 0
Sent: 0 (retransmit: 0, fastretransmit: 0), with data: 0, total data
bytes: 0
```

Cisco Software Modularity

The following sample output from the **show tcp tcb** command displays a Software Modularity image:

Example:

```
Device# show tcp tcb 0x1059C10

Connection state is ESTAB, I/O status: 0, unread input bytes: 0
Local host: 10.4.2.32, Local port: 23
Foreign host: 10.4.2.39, Foreign port: 11000
VRF table id is: 0
Current send queue size: 0 (max 65536)
Current receive queue size: 0 (max 32768)  mis-ordered: 0 bytes
Event Timers (current time is 0xB9ACB9):
Timer          Starts    Wakeups            Next(msec)
Retrans             6          0                0
SendWnd             0          0                0
TimeWait            0          0                0
AckHold             8          4                0
KeepAlive          11          0          7199992
PmtuAger            0          0                0
GiveUp              0          0                0
Throttle            0          0                0
irs:    1633857851  rcvnxt: 1633857890  rcvadv: 1633890620  rcvwnd:  32730
iss:    4231531315  snduna: 4231531392  sndnxt: 4231531392  sndwnd:   4052
sndmax: 4231531392  sndcwnd:     10220
SRTT: 84 ms,  RTTO: 650 ms,  RTV: 69 ms,  KRTT: 0 ms
minRTT: 0 ms,  maxRTT: 200 ms, ACK hold: 200 ms
```

```
Keepalive time: 7200 sec, SYN wait time: 75 sec
Giveup time: 0 ms, Retransmission retries: 0, Retransmit forever: FALSE
State flags: none
Feature flags: Nagle
Request flags: none
Window scales: rcv 0, snd 0, request rcv 0, request snd 0
Timestamp option: recent 0, recent age 0, last ACK sent          0
Datagrams (in bytes): MSS 1460, peer MSS 1460, min MSS 1460, max MSS 1460
Rcvd: 14 (out of order: 0), with data: 10, total data bytes: 38
Sent: 10 (retransmit: 0, fastretransmit: 0), with data: 5, total data bytes: 76
Header prediction hit rate: 72 %
Socket states: SS_ISCONNECTED, SS_PRIV
Read buffer flags: SB_WAIT, SB_SEL, SB_DEL_WAKEUP
Read notifications: 4
Write buffer flags: SB_DEL_WAKEUP
Write notifications: 0
Socket status: 0
```

Step 2 **show tcp brief** [**all** | **numeric**]

(Optional) Displays addresses in IP format.

Use the **show tcp brief** command to display a concise description of TCP connection endpoints. Use the optional **all** keyword to display the status for all endpoints with addresses in a Domain Name System (DNS) hostname format. If this keyword is not used, endpoints in the LISTEN state are not shown. Use the optional **numeric** keyword to display the status for all endpoints with addresses in IP format.

Note If the **ip domain-lookup** command is enabled on the device, and you execute the **show tcp brief** command, the response time of the device to display the output will be very slow. To get a faster response, you should disable the **ip domain-lookup** command.

The following is sample output from the **show tcp brief** command while a user is connected to the system by using Telnet:

Example:

```
Device# show tcp brief

TCB       Local Address           Foreign Address        (state)
609789AC  Device.cisco.com.23     cider.cisco.com.3733   ESTAB
```

The following example shows the IP activity after the **numeric** keyword is used to display addresses in IP format:

Example:

```
Device# show tcp brief numeric

TCB           Local Address          Foreign Address     (state)
6523A4FC      10.1.25.3.11000        10.1.25.3.23         ESTAB
65239A84      10.1.25.3.23           10.1.25.3.11000      ESTAB
653FCBBC      *.1723 *.* LISTEN
```

Step 3 **debug ip tcp transactions**

Use the **debug ip tcp transactions** command to display information about significant TCP transactions such as state changes, retransmissions, and duplicate packets. The TCP/IP network isolated above the data link layer might encounter performance issues. The **debug ip tcp transactions** command can be useful in debugging these performance issues.

The following is sample output from the **debug ip tcp transactions** command:

Example:

```
Device# debug ip tcp transactions

TCP: sending SYN, seq 168108, ack 88655553
TCP0: Connection to 10.9.0.13:22530, advertising MSS 966
```

```
TCP0: state was LISTEN -> SYNRCVD [23 -> 10.9.0.13(22530)]
TCP0: state was SYNSENT -> SYNRCVD [23 -> 10.9.0.13(22530)]
TCP0: Connection to 10.9.0.13:22530, received MSS 956
TCP0: restart retransmission in 5996
TCP0: state was SYNRCVD -> ESTAB [23 -> 10.9.0.13(22530)]
TCP2: restart retransmission in 10689
TCP2: restart retransmission in 10641
TCP2: restart retransmission in 10633
TCP2: restart retransmission in 13384 -> 10.0.0.13(16151)]
TCP0: restart retransmission in 5996 [23 -> 10.0.0.13(16151)]
```

The following line from the **debug ip tcp transactions** command sample output shows that TCP has entered Fast Recovery mode:

Example:

```
fast re-transmit - sndcwnd - 512, snd_last - 33884268765
```

The following lines from the **debug ip tcp transactions** command sample output show that a duplicate acknowledgment is received when TCP is in Fast Recovery mode (first line) and a partial acknowledgment has been received (second line):

Example:

```
TCP0:ignoring second congestion in same window sndcwn - 512, snd_1st - 33884268765
TCP0:partial ACK received sndcwnd:338842495
```

Step 4 **debug ip tcp congestion**

Use the **debug ip tcp congestion** command to display information about TCP congestion events. The TCP/IP network isolated above the data link layer might encounter performance issues. The **debug ip tcp congestion** command can be used to debug these performance issues. The command also displays information related to variations in the TCP send window, congestion window, and congestion threshold window.

The following is sample output from the **debug ip tcp congestion** command:

Example:

```
Device# debug ip tcp congestion

*May 20 22:49:49.091: Setting New Reno as congestion control algorithm
*May 22 05:21:47.281: Advance cwnd by 12
*May 22 05:21:47.281: TCP85FD0C10: sndcwnd: 1472
*May 22 05:21:47.285: Advance cwnd by 3
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1475
*May 22 05:21:47.285: Advance cwnd by 3
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1478
*May 22 05:21:47.285: Advance cwnd by 9
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1487
*May 20 22:50:32.559: [New Reno] sndcwnd: 8388480 ssthresh: 65535 snd_mark: 232322
*May 20 22:50:32.559: 10.168.10.10:42416 <---> 10.168.30.11:49100 congestion window changes
*May 20 22:50:32.559: cwnd from 8388480 to 2514841, ssthresh from 65535 to 2514841
```

For Cisco TCP, New Reno is the default congestion control algorithm. However, an application can also use Binary Increase Congestion Control (BIC) as the congestion control algorithm. The following is sample output from the **debug ip tcp congestion** command using BIC:

Example:

```
Device# debug ip tcp congestion
```

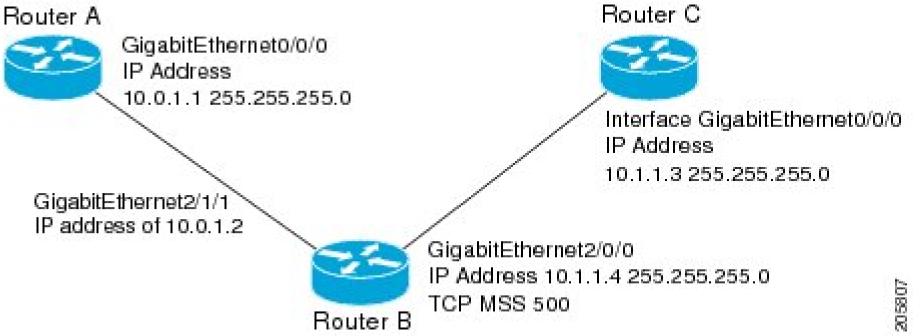
```
*May 22 05:21:42.281: Setting BIC as congestion control algorithm
*May 22 05:21:47.281: Advance cwnd by 12
*May 22 05:21:47.281: TCP85FD0C10: sndcwnd: 1472
*May 22 05:21:47.285: Advance cwnd by 3
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1475
*May 22 05:21:47.285: Advance cwnd by 3
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1478
*May 22 05:21:47.285: Advance cwnd by 9
*May 22 05:21:47.285: TCP85FD0C10: sndcwnd: 1487
*May 20 22:50:32.559: [BIC] sndcwnd: 8388480 ssthresh: 65535 bic_last_max_cwnd: 0 last_cwnd: 8388480
*May 20 22:50:32.559: 10.168.10.10:42416 <---> 10.168.30.11:49100 congestion window changes
*May 20 22:50:32.559: cwnd from 8388480 to 2514841, ssthresh from 65535 to 2514841
*May 20 22:50:32.559: bic_last_max_cwnd changes from 0 to 8388480
```

Configuration Examples for TCP

Example: Configuring the TCP MSS Adjustment

The following example shows how to configure and verify the interface adjustment value for the example topology displayed in the figure below:

Figure 5: Example Topology for TCP MSS Adjustment



Configure the interface adjustment value on router B:

```
Router_B(config)# interface GigabitEthernet 2/0/0
Router_B(config-if)# ip tcp adjust-mss 500
```

Telnet from router A to router C with B having the Maximum Segment Size (MSS) adjustment configured:

```
Router_A# telnet 192.168.1.1

Trying 192.168.1.1... Open
```

Observe the debug output from router C:

```
Router_C# debug ip tcp transactions

Sep 5 18:42:46.247: TCP0: state was LISTEN -> SYNRCVD [23 -> 10.0.1.1(38437)]
Sep 5 18:42:46.247: TCP: tcb 32290C0 connection to 10.0.1.1:38437, peer MSS 500, MSS is 500
Sep 5 18:42:46.247: TCP: sending SYN, seq 580539401, ack 6015751
Sep 5 18:42:46.247: TCP0: Connection to 10.0.1.1:38437, advertising MSS 500
Sep 5 18:42:46.251: TCP0: state was SYNRCVD -> ESTAB [23 -> 10.0.1.1(38437)]
```

The MSS gets adjusted to 500 on Router B as configured.

The following example shows the configuration of a Point-to-Point Protocol over Ethernet (PPPoE) client with the MSS value set to 1452:

```
Device(config)# vpdn enable
Device(config)# no vpdn logging
Device(config)# vpdn-group 1
Device(config-vpdn)# request-dialin
Device(config-vpdn-req-in)# protocol pppoe
Device(config-vpdn-req-in)# exit
Device(config-vpdn)# exit
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# ip address 192.168.100.1.255.255.255.0
Device(config-if)# ip tcp adjust-mss 1452
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface ATM 0
Device(config-if)# no ip address
Device(config-if)# no atm ilmi-keepalive
Device(config-if)# pvc 8/35
Device(config-if)# pppoe client dial-pool-number 1
Device(config-if)# dsl equipment-type CPE
Device(config-if)# dsl operating-mode GSHDSL symmetric annex B
Device(config-if)# dsl linerate AUTO
Device(config-if)# exit
Device(config)# interface Dialer 1
Device(config-if)3 ip address negotiated
Device(config-if)# ip mtu 1492
Device(config-if)# ip nat outside
Device(config-if)# encapsulation ppp
Device(config-if)# dialer pool 1
Device(config-if)# dialer-group 1
Device(config-if)# ppp authentication pap callin
Device(config-if)# ppp pap sent-username sohodyn password 7 141B1309000528
Device(config-if)# ip nat inside source list 101 Dialer1 overload
Device(config-if)# exit
Device(config)# ip route 0.0.0.0.0.0.0.0 Dialer1
Device(config)# access-list permit ip 192.168.100.0.0.0.0.255 any
```

The following example shows the configuration of interface adjustment value for IPv6 traffic:

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 0/0/0
Device(config)# ipv6 tcp adjust-mss 1452
Device(config)# end
```

Example: Configuring the TCP Application Flags Enhancement

The following output shows the flags (status and option) displayed using the **show tcp** command:

```
Device# show tcp
.
.
.
Status Flags: passive open, active open, retransmission timeout
 App closed
Option Flags: vrf id set
IP Precedence value: 6
.
.
```

```
.
SRTT: 273 ms, RTTO: 490 ms, RTV: 217 ms, KRTT: 0 ms
minRTT: 0 ms, maxRTT: 300 ms, ACK hold: 200 ms
```

Example: Displaying Addresses in IP Format

The following example shows the IP activity by using the **numeric** keyword to display the addresses in IP format:

```
Device# show tcp brief numeric

TCB           Local Address          Foreign Address     (state)
6523A4FC      10.1.25.3.11000        10.1.25.3.23         ESTAB
65239A84      10.1.25.3.23           10.1.25.3.11000      ESTAB
653FCBBC      *.1723 *.* LISTEN
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
IP Application Services commands	IP Application Services Command Reference

Standards and RFCs

Standard/RFC	Title
RFC 793	Transmission Control Protocol
RFC 1191	Path MTU discovery
RFC 1323	TCP Extensions for High Performance
RFC 2018	TCP Selective Acknowledgment Options
RFC 2581	TCP Congestion Control
RFC 3168	The Addition of Explicit Congestion Notification (ECN) to IP
RFC 3782	The NewReno Modification to TCP's Fast Recovery Algorithm
RFC 4022	Management Information Base for the Transmission Control Protocol (TCP)

MIBs

MIB	MIBs Link
CISCO-TCP-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for TCP

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 7: Feature Information for TCP

Feature Name	Releases	Feature Information
TCP MIB for RFC4022 Support	15.0(1)SY 15.1(1)SY	The TCP MIB for RFC 4022 Support feature introduces support for RFC 4022, *Management Information Base for the Transmission Control Protocol (TCP)*. RFC 4022 is an incremental change of the TCP MIB to improve the manageability of TCP. There are no new or modified commands for this feature.

Feature Name	Releases	Feature Information
TCP MSS Adjust	15.0(1)SY	The TCP MSS Adjust feature enables the configuration of the maximum segment size (MSS) for transient packets that traverse a router, specifically TCP segments in the SYN bit set. The following command was introduced by this feature: **ip tcp adjust-mss**.

CHAPTER 5

Configuring UDP Forwarding Support for IP Redundancy Virtual Router Groups

User Datagram Protocol (UDP) forwarding is a feature used in Cisco IOS software to forward broadcast and multicast packets received for a specific IP address. Virtual Router Group (VRG) support, implemented with the Hot Standby Routing Protocol (HSRP), allows a set of routers to be grouped as a logical router that answers to a well-known IP address. The UDP Forwarding Support for IP Redundancy Virtual Router Groups feature enables UDP forwarding to be VRG aware; this results in packets getting forwarde only to the active router in the VRG.

This module explains the concepts related UDP forwarding and VRG support and describes how to configure UDP forwarding support for IP Redundancy Virtual Router Groups in a network.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for UDP Forwarding Support for IP Redundancy Virtual Router Groups

- The UDP Forwarding Support for Virtual Router Groups feature is available only on platforms that support VRGs.

Information About UDP Forwarding Support for IP Redundancy Virtual Router Groups

Benefits of the UDP Forwarding Support for Virtual Router Groups Feature

Forwarding is limited to the active router in the VRG instead of all routers within the VRG. Prior to the implementation of this feature, the only VRG support was HSRP. Within a VRG that is formed by HSRP, the forwarding of UDP-based broadcast and multicast packets is done by all the routers within the VRG. This process can cause some DHCP servers to operate incorrectly. The UDP Forwarding Support for VRGs feature limits forwarding to the active router in the VRG.

VRG awareness is achieved with IP Redundancy Service (IRS). The IRS application programming interface (API) provides notification updates of a specific VRG, addition and deletion of a VRG, and querying of the current state of a VRG. A state change notification is provided to avoid the performance impact of querying the state of the VRG each time it is needed. The UDP forwarding code caches the VRG state for each required helper address that is defined. Each time the UDP forwarding code needs to execute, it checks the current state of the VRG associated with the helper address and forwards packets only to VRGs that are active.

How to Configure UDP Forwarding Support for IP Redundancy Virtual Router Groups

Configuring UDP Forwarding Support for IP Redundancy Virtual Router Groups

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **no shutdown**
5. **ip address** *ip-address mask*
6. **ip helper-address** *address* **redundancy** *vrg-name*
7. **standby** *group-number* **ip** *ip-address*
8. **standby** *group-number* **name** *group-name*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface fastethernet 0/0`	Specifies an interface and enters interface configuration mode.
Step 4	**no shutdown** **Example:** `Router(config-if)# no shutdown`	Restarts a disabled interface.

	Command or Action	Purpose
Step 5	**ip address** *ip-address mask* **Example:** `Router(config-if)# ip address 172.16.10.1 255.255.255.0`	Sets a primary address for the interface.
Step 6	**ip helper-address** *address* **redundancy** *vrg-name* **Example:** `Router(config-if)# ip helper-address 10.1.1.1 redundancy vrg1`	Enables UDP forwarding support for the VRG.
Step 7	**standby** *group-number* **ip** *ip-address* **Example:** `Router(config-if)# standby 1 ip 172.16.10.254`	Activates HSRP.
Step 8	**standby** *group-number* **name** *group-name* **Example:** `Router(config-if)# standby 1 name vrg1`	Configures the name of the standby group.

Step 9	**end**
	Example:
	`Router(config-if)# end`
		Exits the current configuration mode and returns to privileged EXEC mode.

Configuration Examples for UDP Forwarding Support for IP Redundancy Virtual Router Groups

Example: Configuring UDP Forwarding Support for IP Redundancy Virtual Router Groups

The following example shows how to configure UDP Forwarding Support for IP Redundancy Virtual Router Groups:

```
Router(config)# interface fastethernet 0/0
Router(config-if)# no shutdown
Router(config-if)# ip address 172.16.10.1 255.255.255.0
Router(config-if)# ip helper-address 10.1.1.1 redundancy vrg1
Router(config-if)# standby 1 ip 172.16.10.254
Router(config-if)# standby 1 name vrg1
Router(config-if)# end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
IP application services commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IP Application Services Command Reference*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified	—

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified	—

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

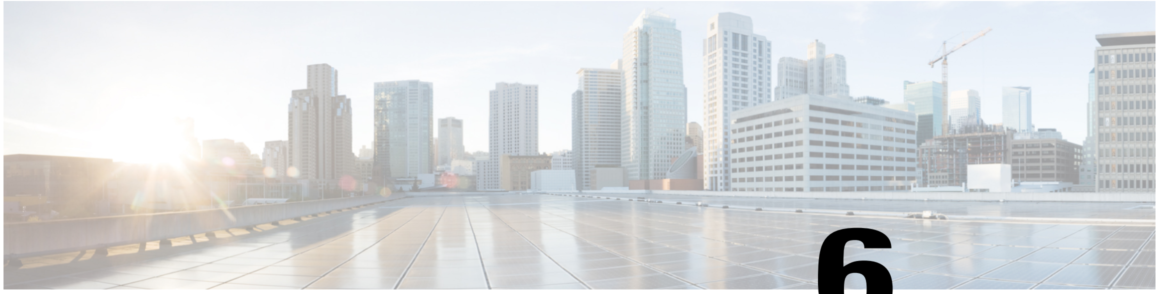
Feature Information for UDP Forwarding Support for IP Redundancy Virtual Router Groups

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 8: Feature Information for UDP Forwarding Support for IP Redundancy Virtual Router Groups

Feature Name	Releases	Feature Information
UDP Forwarding Support for IP Redundancy Virtual Router Group	Cisco IOS XE 3.1.0SG 12.2(50)SY 12.2(15)T 15.0(1)SY 15.2(1)S	UDP forwarding is a feature used in Cisco IOS software to forward broadcast and multicast packets received for a specific IP address. Virtual Router Group (VRG) support is implemented with the Hot Standby Routing Protocol (HSRP) and it allows a set of routers to be grouped as a logical router that answers to a well-known IP address. The UDP Forwarding Support for IP Redundancy Virtual Router Groups feature enables UDP forwarding to be VRG aware, resulting in forwarding only to the active router in the VRG. The following command was introduced or modified: **ip helper-address**.



CHAPTER 6

Configuring WCCP

The Web Cache Communication Protocol (WCCP) is a Cisco-developed content-routing technology that intercepts IP packets and redirects those packets to a destination other than that specified in the IP packet. Typically the packets are redirected from their destination web server on the Internet to a content engine that is local to the client. In some WCCP deployment scenarios, redirection of traffic may also be required from the web server to the client. WCCP enables you to integrate content engines into your network infrastructure.

Cisco IOS Release 12.1 and later releases allow the use of either WCCP Version 1 (WCCPv1) or Version 2 (WCCPv2).

The tasks in this document assume that you have already configured content engines on your network. For specific information on hardware and network planning associated with Cisco Content Engines and WCCP, see the Cisco Content Engines documentation at the following URL:

http://www.cisco.com/univercd/cc/td/doc/product/webscale/content/index.htm

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for WCCP

- To use WCCP, IP must be configured on the interface connected to the Internet and another interface must be connected to the content engine.
- The interface connected to the content engine must be a Fast Ethernet or Gigabit Ethernet interface.
- Only Catalyst 6500 series switches with a PFC4 support the following hardware capabilities:
    - WCCP generic routing encapsulation (GRE) decapsulation in hardware
    - WCCP Egress Mask assignment in hardware
    - WCCP Exclude capability in hardware

Restrictions for WCCP

General

The following limitations apply to Web Cache Communication Protocol version1 (WCCPv1) and WCCP version 2 (WCCPv2):

- WCCP works only with IPv4 networks.
- WCCP bypasses Network Address Translation (NAT) when Cisco Express Forwarding is enabled.
- WCCP interoperability with Network Address Translation (NAT) in conjunction with Zone-Based Firewall (ZBF) is not supported.

WCCPv1

The following limitations apply to WCCPv1:

- WCCPv1 supports the redirection of HTTP (TCP port 80) traffic only.
- WCCPv1 does not allow multiple routers to be attached to a cluster of content engines.

WCCPv2

The following limitations apply to WCCPv2:

- WCCP works only with IPv4 networks.
- For routers servicing a multicast cluster, the Time To Live (TTL) value must be set at 15 or fewer.
- Service groups can comprise up to 32 content engines and 32 routers.
- All content engines in a cluster must be configured to communicate with all routers servicing the cluster.
- Multicast addresses must be from 224.0.0.0 to 239.255.255.255.

WCCP VRF Support

In Cisco IOS Release 12.2(33)SRE, this feature is supported only on Cisco 7200 NPE-G2 and Cisco 7304-NPE-G100 routers.

This feature is supported in Cisco IOS Release 12.2(50)SY on Catalyst 6000 series switches with a PFC4.

Layer 2 Forwarding and Return

The following limitations apply to WCCP Layer 2 Forwarding and Return:

In a Cisco Wide Area Application Services (WAAS) and Cisco IOS XE firewall configuration, all packets processed by a Wide Area Application Engine (WAE) device must go over the Cisco IOS XE firewall in both directions to support the Web Cache Coordination Protocol (WCCP) generic routing encapsulation (GRE) redirect. This situation occurs when a Layer 2 redirect is not available. If a Layer 2 redirect is configured on the WAE, the system defaults to the GRE redirect to continue to function.

The client device and a WAE device or a cache engine cannot be connected to a Cisco device with the same interface and WCCP redirect configured on the interface.

The following two configurations are supported:

1. For WCCP Layer 2 return, the client and WAE are connected to a Cisco device with same interface and WCCP output is configured on the interface.
2. For WCCP Layer 2 return, the client and WAE are connected to a Cisco device with same physical interface but in different VLANs and sub-interfaces.

Cisco Catalyst 4500 Series Switches

The following limitations apply to Cisco Catalyst 4500 series switches:

- Catalyst 4500 series switches do not support WCCPv1.
- Up to eight service groups are supported at the same time on the same client interface.
- The Layer 2 (L2) rewrite forwarding method is supported, but generic routing encapsulation (GRE) is not.
- Direct L2 connectivity to content engines is required; Layer 3 (L3) connectivity of one or more hops away is not supported.
- Ternary content addressable memory (TCAM) friendly mask-based assignment is supported, but the hash bucket-based method is not.
- Redirect ACL for WCCP on a client interface is not supported.
- Incoming traffic redirection on an interface is supported, but outgoing traffic redirection is not.
- When TCAM space is exhausted, traffic is not redirected; it is forwarded normally.
- The WCCP version 2 standard allows for support of up to 256 distinct masks. However, a Catalyst 4500 series switch supports only mask assignment tables with a single mask.

Cisco Catalyst 6500 Series Switches

The following limitation apply to Cisco Catalyst 6500 series switches:

- With a Policy Feature Card 2 (PFC2), Cisco IOS Release 12.2(17d)SXB and later releases support WCCP.

- With a PFC3, Cisco IOS Release 12.2(18)SXD1 and later releases support WCCP.
- With a PFC4, Cisco IOS Release 12.2(50)SY and later releases support WCCP and introduce support for WCCP GRE decapsulation, WCCP mask assignment, and WCCP exclude capability in hardware.
- To use the WCCP Layer 2 PFC redirection feature, configure WCCP on the Catalyst 6500 series switch and configure accelerated WCCP on the cache engine as described in the Transparent Caching document.
- Cisco Application and Content Networking System (ACNS) software releases later than Release 4.2.2 support WCCP Layer 2 Policy Feature Card (PFC) redirection hardware acceleration.
- A content engine configured for mask assignment that tries to join a farm where the selected assignment method is hash remains out of the farm as long as the cache engine assignment method does not match that of the existing farm.
- When WCCP Layer 2 PFC redirection is the forwarding method for a service group, the packet counters in the **show ip wccp** *service-number* command output display flow counts instead of packet counts.

Catalyst 6500 Series Switches and Cisco 7600 Series Routers Access Control Lists

When WCCP is using the mask assignment, any redirect list is merged with the mask information from the appliance and the resulting merged ACL is passed down to the Catalyst 6500 series switch or Cisco 7600 series router hardware. Only Permit or Deny ACL entries from the redirect list in which the protocol is IP or exactly matches the service group protocol are merged with the mask information from the appliance.

The following restrictions apply to the redirect-list ACL:

- The ACL must be an IPv4 simple or extended ACL.
- Only individual source or destination port numbers may be specified; port ranges cannot be specified.
- The only valid matching criteria in addition to individual source or destination port numbers are **dscp** or **tos**.
- The use of **fragments**, **time-range**, or **options** keywords, or any TCP flags is not permitted.

If the redirect ACL does not meet the restrictions shown, the system will log the following error message:

```
WCCP-3-BADACE: Service <service group>, invalid access-list entry (seq:<sequence>,
reason:<reason>)
```

WCCP continues to redirect packets, but the redirection is carried out in software (NetFlow Switching) until the access list is adjusted.

Information About WCCP

WCCP Overview

WCCP uses Cisco Content Engines (or other content engines running WCCP) to localize web traffic patterns in the network, enabling content requests to be fulfilled locally. Traffic localization reduces transmission costs and download time.

WCCP enables routing platforms to transparently redirect content requests. The main benefit of transparent redirection is that users need not configure their browsers to use a web proxy. Instead, they can use the target URL to request content, and have their requests automatically redirected to a content engine. The word "transparent" in this case means that the end user does not know that a requested file (such as a web page) came from the content engine instead of from the originally specified server.

A content engine receiving a request attempts to service it from its own local cache. If the requested information is not present, the content engine issues its own request to the originally targeted server to get the required information. A content engine retrieving the requested information forwards it to the requesting client and caches it to fulfill future requests, thus maximizing download performance and substantially reducing transmission costs.

WCCP enables a series of content engines, called a content engine cluster, to provide content to a router or multiple routers. Network administrators can easily scale their content engines to manage heavy traffic loads through these clustering capabilities. Cisco clustering technology enables each cluster member to work in parallel, resulting in linear scalability. Clustering content engines greatly improves the scalability, redundancy, and availability of your caching solution. You can cluster up to 32 content engines to scale to your desired capacity.

Layer 2 Forwarding Redirection and Return

WCCP uses either generic routing encapsulation (GRE) or Layer 2 (L2) to redirect or return IP traffic. When WCCP forwards traffic via GRE, the redirected packets are encapsulated within a GRE header. The packets also have a WCCP redirect header. When WCCP forwards traffic using L2, the original MAC header of the IP packet is overwritten and replaced with the MAC header for the WCCP client.

Using L2 as a forwarding method allows direct forwarding to the content engine without further lookup. Layer 2 redirection requires that the router and content engines are directly connected, that is, on the same IP subnetwork.

When WCCP returns traffic via GRE, the returned packets are encapsulated within a GRE header. The destination IP address is the address of the router and the source address is the address of the WCCP client. When WCCP returns traffic via L2, the original IP packet is returned without any added header information. The router to which the packet is returned will recognize the source of the packet and prevent redirection.

The WCCP redirection method does not have to match the return method.

L2 forwarding, return, or redirection are typically used for hardware-accelerated platforms. Depending on your release, L2 forwarding, return, and redirection can also be used for software-switching platforms.

On Cisco ASR 1000 Series Aggregation Services Routers, both the GRE and L2 forward and return methods use the hardware. Therefore, there is no significant performance degradation between them.

For content engines running Application and Content Networking System (ACNS) software, use the **wccp custom-web-cache** command with the **l2-redirect** keyword to configure L2 redirection. For content engines running Cisco Wide Area Application Services (WAAS) software, use the **wccp tcp-promiscuous** command with the **l2-redirect** keyword to configure L2 redirection.

Note Before configuring a GRE tunnel, configure a loopback interface (that is not attached to a VRF) with an IP address so that the internally created tunnel interface is enabled for IPv4 forwarding by unnumbering itself to this dummy loopback interface. You do not need to configure a loopback interface if the system has at least one interface that is not attached to a VRF and that is configured with an IPv4 address.

For information about Cisco ACNS commands used to configure Cisco Content Engines, see the Cisco ACNS Software Command Reference.

For more information about WAAS commands used to configure Cisco Content Engines, see the Cisco Wide Area Application Services Command Reference.

WCCP Mask Assignment

The WCCP Mask Assignment feature enables mask assignment as the load-balancing method (instead of the default hash assignment method) for a WCCP service.

For content engines running Application and Content Networking System (ACNS) software, use the **wccp custom-web-cache** command with the **mask-assign** keyword to configure mask assignment. For content engines running Cisco Wide Area Application Services (WAAS) software, use the **wccp tcp-promiscuous** command with the **mask-assign** keyword to configure mask assignment.

For information about Cisco ACNS commands used to configure Cisco Content Engines, see the Cisco ACNS Software Command Reference.

For more information about WAAS commands used to configure Cisco Content Engines, see the Cisco Wide Area Application Services Command Reference.

Hardware Acceleration

Catalyst 6500 series switches and Cisco 7600 series routers provide WCCP Layer 2 Policy Feature Card (PFC) redirection hardware acceleration. Hardware acceleration allows Cisco Content Engines to perform a L2 MAC address rewrite redirection method when directly connected to a compatible switch or router.

Redirection processing is accelerated in the switching or routing hardware, which is more efficient than L3 redirection with Generic Routing Encapsulation (GRE). L2 redirection takes place on the switch or router, and is not visible to the Multilayer Switch Feature Card (MSFC). The WCCP L2 PFC redirection feature requires no configuration on the MSFC. The **show ip wccp** {*service-number* | **web-cache**} **detail** command displays which redirection method is in use for each content engine.

In order for the router or switch to make complete use of hardware redirection, the content engine must be configured with L2 redirection and mask assignment.

Use the **ip wccp web-cache accelerated** command on hardware-based platforms to enforce the use of L2 redirection and mask assignment. Using this command configures the router to form a service group and redirect packets with an appliance only if the appliance is configured for L2 and mask assignment.

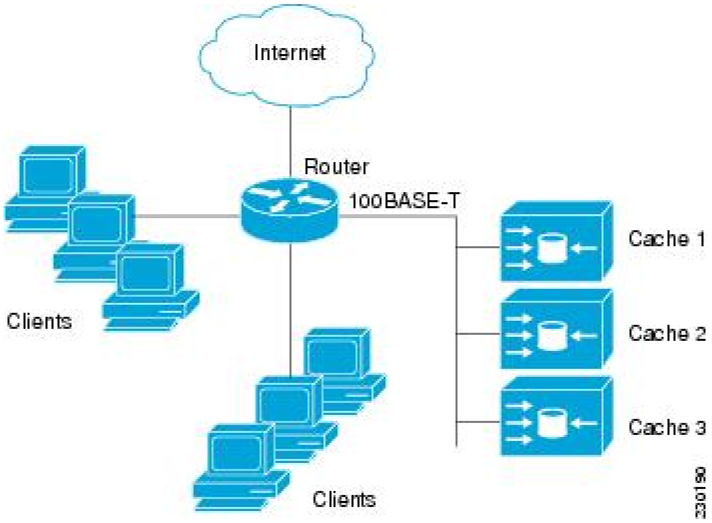
The following guidelines apply to WCCP Layer 2 PFC redirection:

- The WCCP Layer 2 PFC redirection feature sets the IP flow mask to full-flow mode.
- You can configure the Cisco Cache Engine software Release 2.2 or later releases to use the WCCP Layer 2 PFC redirection feature.
- L2 redirection takes place on the PFC and is not visible to the MSFC. The **show ip wccp** {*service-number* | **web-cache**} **detail** command on the MSFC displays statistics for only the first packet of an L2 redirected flow, which provides an indication of how many flows, rather than packets, are using L2 redirection. You can view information about L2 redirected flows by entering the **show platform flow ip** command. The PFC3 provides hardware acceleration for GRE. If you use WCCP Layer 3 redirection with GRE, there is hardware support for encapsulation, but the PFC3 does not provide hardware support for decapsulation of WCCP GRE traffic.

WCCPv1 Configuration

With WCCPv1, only a single router services a cluster. In this scenario, this router is the device that performs all the IP packet redirection. The figure below illustrates the WCCPv1 configuration.

Figure 6: WCCPv1 Configuration



Content is not duplicated on the content engines. The benefit of using multiple content engines is that you can scale a caching solution by clustering multiple physical content engines to appear as one logical cache.

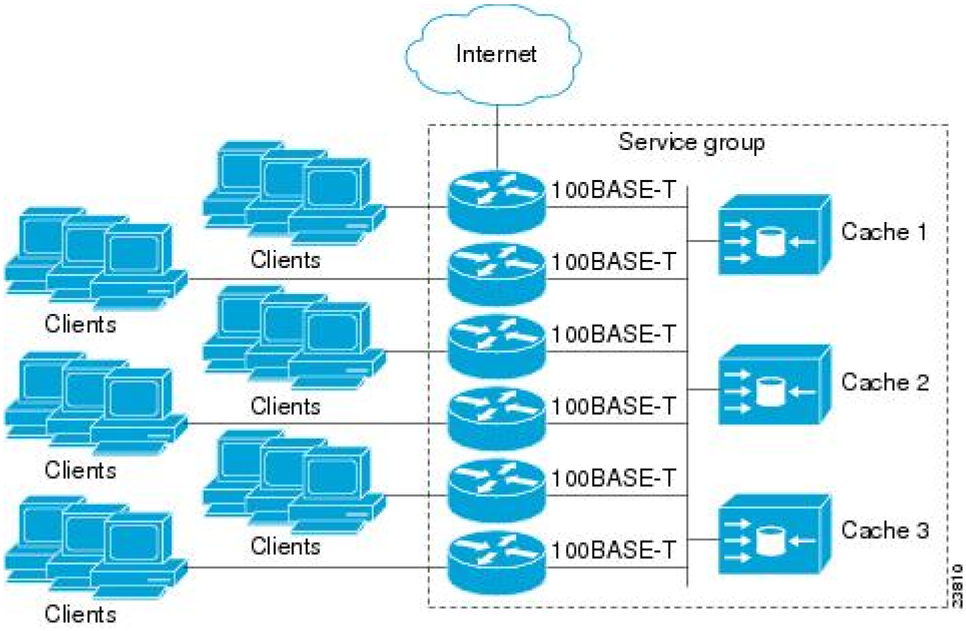
The following sequence of events details how WCCPv1 configuration works:

1. Each content engine is configured by the system administrator with the IP address of the control router. Up to 32 content engines can connect to a single control router.
2. The content engines send their IP addresses to the control router using WCCP, indicating their presence. Routers and content engines communicate to each other via a control channel; this channel is based on UDP port 2048.
3. This information is used by the control router to create a cluster view (a list of caches in the cluster). This view is sent to each content engine in the cluster, essentially making all the content engines aware of each other. A stable view is established after the membership of the cluster remains the same for a certain amount of time.
4. When a stable view has been established, one content engine is elected as the lead content engine. (The lead is defined as the content engine seen by all the content engines in the cluster with the lowest IP address). This lead content engine uses WCCP to indicate to the control router how IP packet redirection should be performed. Specifically, the lead content engine designates how redirected traffic should be distributed across the content engines in the cluster.

WCCPv2 Configuration

Multiple routers can use WCCPv2 to service a content engine cluster. In WCCPv1, only one router could redirect content requests to a cluster. The figure below illustrates a sample configuration using multiple routers.

Figure 7: Cisco Content Engine Network Configuration Using WCCPv2



The subset of content engines within a cluster and routers connected to the cluster that are running the same service is known as a service group. Available services include TCP and UDP redirection.

In WCCPv1, the content engines were configured with the address of the single router. WCCPv2 requires that each content engine be aware of all the routers in the service group. To specify the addresses of all the routers in a service group, you must choose one of the following methods:

- Unicast—A list of router addresses for each of the routers in the group is configured on each content engine. In this case the address of each router in the group must be explicitly specified for each content engine during configuration.
- Multicast—A single multicast address is configured on each content engine. In the multicast address method, the content engine sends a single-address notification that provides coverage for all routers in the service group. For example, a content engine could indicate that packets should be sent to a multicast address of 224.0.0.100, which would send a multicast packet to all routers in the service group configured for group listening using WCCP (see the **ip wccp group-listen** or the **ipv6 wccp group-listen** interface configuration command for details).

The multicast option is easier to configure because you need only specify a single address on each content engine. This option also allows you to add and remove routers from a service group dynamically, without needing to reconfigure the content engines with a different list of addresses each time.

The following sequence of events details how WCCPv2 configuration works:

1 Each content engine is configured with a list of routers.

2 Each content engine announces its presence and a list of all routers with which it has established communications. The routers reply with their view (list) of content engines in the group.

3 When the view is consistent across all content engines in the cluster, one content engine is designated as the lead and sets the policy that the routers need to deploy in redirecting packets.

WCCPv2 Support for Services Other Than HTTP

WCCPv2 allows redirection of traffic other than HTTP (TCP port 80 traffic), including a variety of UDP and TCP traffic. WCCPv2 supports the redirection of packets intended for other ports, including those used for proxy-web cache handling, File Transfer Protocol (FTP) caching, FTP proxy handling, web caching for ports other than 80, and Real Audio, video, and telephony applications.

To accommodate the various types of services available, WCCPv2 introduced the concept of multiple *service groups*. Service information is specified in the WCCP configuration commands using dynamic services identification numbers (such as 98) or a predefined service keyword (such as **web-cache**). This information is used to validate that service group members are all using or providing the same service.

The content engines in a service group specify traffic to be redirected by protocol (TCP or UDP) and up to eight source or destination ports. Each service group has a priority status assigned to it. The priority of a dynamic service is assigned by the content engine. The priority value is in the range of 0 to 255 where 0 is the lowest priority. The predefined web-cache service has an assigned priority of 240.

WCCPv2 Support for Multiple Routers

WCCPv2 allows multiple routers to be attached to a cluster of cache engines. The use of multiple routers in a service group allows for redundancy, interface aggregation, and distribution of the redirection load. WCCPv2 supports up to 32 routers per service group. Each service group is established and maintained independently.

WCCPv2 MD5 Security

WCCPv2 provides optional authentication that enables you to control which routers and content engines become part of the service group using passwords and the Hashed Message Authentication Code—Message Digest (HMAC MD5) standard. Shared-secret MD5 one-time authentication (set using the **ip wccp** [**password** [**0** | **7**] *password*] global configuration command) enables messages to be protected against interception, inspection, and replay.

WCCPv2 Web Cache Packet Return

If a content engine is unable to provide a requested object it has cached due to error or overload, the content engine will return the request to the router for onward transmission to the originally specified destination server. WCCPv2 provides a check on packets that determines which requests have been returned from the content engine unserviced. Using this information, the router can then forward the request to the originally targeted server (rather than attempting to resend the request to the content engine cluster). This process provides error handling transparency to clients.

Typical reasons why a content engine would reject packets and initiate the packet return feature include the following:

- Instances when the content engine is overloaded and has no room to service the packets
- Instances when the content engine is filtering for certain conditions that make caching packets counterproductive (for example, when IP authentication has been turned on)

WCCPv2 Load Distribution

WCCPv2 can be used to adjust the load being offered to individual content engines to provide an effective use of the available resources while helping to ensure high quality of service (QoS) to the clients. WCCPv2 allows the designated content engine to adjust the load on a particular content engine and balance the load across the content engines in a cluster. WCCPv2 uses three techniques to perform load distribution:

- Hot spot handling—Allows an individual hash bucket to be distributed across all the content engines. Prior to WCCPv2, information from one hash bucket could go to only one content engine.
- Load balancing—Allows the set of hash buckets assigned to a content engine to be adjusted so that the load can be shifted from an overwhelmed content engine to other members that have available capacity.
- Load shedding—Enables the router to selectively redirect the load to avoid exceeding the capacity of a content engine.

The use of these hashing parameters prevents one content engine from being overloaded and reduces the potential for bottlenecking.

WCCP Mask Assignment

The WCCP Mask Assignment feature enables mask assignment as the load-balancing method (instead of the default hash assignment method) for a WCCP service.

For content engines running Application and Content Networking System (ACNS) software, use the **wccp custom-web-cache** command with the **mask-assign** keyword to configure mask assignment. For content engines running Cisco Wide Area Application Services (WAAS) software, use the **wccp tcp-promiscuous** command with the **mask-assign** keyword to configure mask assignment.

For information about Cisco ACNS commands used to configure Cisco Content Engines, see the Cisco ACNS Software Command Reference.

For more information about WAAS commands used to configure Cisco Content Engines, see the Cisco Wide Area Application Services Command Reference.

WCCP VRF Support

The WCCP VRF Support feature enhances the WCCPv2 protocol by implementing support for virtual routing and forwarding (VRF).

The WCCP VRF Support feature allows service groups to be configured on a per-VRF basis in addition to those defined globally.

Along with the service identifier, the VRF of WCCP protocol packets arriving at the router is used to associate cache-engines with a configured service group.

The same VRF must have the interface on which redirection is applied, the interface which is connected to cache engine, and the interface on which the packet would have left if it had not been redirected.

WCCP VRF Tunnel Interfaces

In releases that support the WCCP VRF Support feature, the use of GRE redirection results in the creation of new tunnel interfaces. You can display these tunnel interfaces by entering the **show ip interface brief | include tunnel** command:

```
Device# show ip interface brief | include tunnel

Tunnel0                172.16.0.1      YES unset  up                     up
Tunnel1                172.16.0.1      YES unset  up                     up
Tunnel2                172.16.0.1      YES unset  up                     up
Tunnel3                172.16.0.1      YES unset  up                     up
Device#
```

The tunnel interfaces are automatically created in order to process outgoing GRE-encapsulated traffic for WCCP. The tunnel interfaces appear when a content engine connects and requests GRE redirection. The tunnel interfaces are not created directly by WCCP, but are created indirectly via a tunnel application programming interface (API). WCCP does not have direct knowledge of the tunnel interfaces, but can redirect packets to them, resulting in the appropriate encapsulation being applied to the packets. After the appropriate encapsulation is applied, the packet is then sent to the content engine.

Note The tunnel interfaces are not used to connect with incoming WCCP GRE return packets.

One tunnel is created for each service group that is using GRE redirection. One additional tunnel is created to provide an IP address that allows the other tunnel group interfaces to be unnumbered but still enabled for IPv4.

You can confirm the connection between the tunnels and WCCP by entering the **show tunnel groups wccp** command:

```
Device# show tunnel groups wccp

WCCP : service group 0 in "Default", ver v2, assgnmnt: hash-table
   intf: Tunnel0, locally sourced
 WCCP : service group 317 in "Default", ver v2, assgnmnt: hash-table
   intf: Tunnel3, locally sourced
 WCCP : service group 318 in "Default", ver v2, assgnmnt: hash-table
   intf: Tunnel2, locally sourced
```

You can display additional information about each tunnel interface by entering the **show tunnel interface** *interface-number* command:

```
Device# show tunnel interface t0

Tunnel0
   Mode:multi-GRE/IP, Destination UNKNOWN, Source 10.1.1.80
   Application ID 2: WCCP : service group 0 in "Default", ver v2, assgnmnt: hash-table
   Linestate - current up
   Internal linestate - current up, evaluated up

Device# show tunnel interface t1

Tunnel1
   Mode:multi-GRE/IP, Destination UNKNOWN, Source 172.16.0.1
   Application ID 2: unspecified
   Linestate - current up
   Internal linestate - current up, evaluated up

Device# show tunnel interface t2
```

```
Tunnel2
   Mode:multi-GRE/IP, Destination UNKNOWN, Source 10.1.1.80
   Application ID 2: WCCP : service group 318 in "Default", ver v2, assgnmnt: hash-table
   Linestate - current up
   Internal linestate - current up, evaluated up

Device# show tunnel interface t3

Tunnel3
   Mode:multi-GRE/IP, Destination UNKNOWN, Source 10.1.1.80
   Application ID 2: WCCP : service group 317 in "Default", ver v2, assgnmnt: hash-table
   Linestate - current up
   Internal linestate - current up, evaluated up
Device#
```

Note that the service group number shown in the examples is the internal tunnel representation of the WCCP service group number. Group 0 is the web-cache service. To determine the dynamic services, subtract 256 from the displayed service group number to convert to the WCCP service group number. For interfaces that are used for redirection, the source address shown is the WCCP router ID.

You can display information about the connected content engines and encapsulation, including software packet counters, by entering the **show adjacency** [*tunnel-interface*] [**encapsulation**] [**detail**] [**internal**] command:

```
Device# show adjacency t0

Protocol Interface                 Address
IP       Tunnel0                   10.1.1.82(3)

Device# show adjacency t0 encapsulation

Protocol Interface                 Address
IP       Tunnel0                   10.1.1.82(3)
  Encap length 28
  4500000000000000FF2F7D2B1E010150
  1E0101520000883E00000000
  Provider: TUNNEL
  Protocol header count in macstring: 3
    HDR 0: ipv4
       dst: static, 10.1.1.82
       src: static, 10.1.1.80
      prot: static, 47
       ttl: static, 255
        df: static, cleared
      per packet fields: tos ident tl chksm
    HDR 1: gre
      prot: static, 0x883E
      per packet fields: none
    HDR 2: wccpv2
       dyn: static, cleared
      sgID: static, 0
      per packet fields: alt altB priB

Device# show adjacency t0 detail

Protocol Interface                 Address
IP       Tunnel0                   10.1.1.82(3)
                                   connectionid 1
                                   0 packets, 0 bytes
                                   epoch 0
                                   sourced in sev-epoch 1
                                   Encap length 28
                                   4500000000000000FF2F7D2B1E010150
                                   1E0101520000883E00000000
                                   Tun endpt
                                   Next chain element:
                                    IP adj out of Ethernet0/0, addr 10.1.1.82
Device# show adjacency t0 internal

Protocol Interface                 Address
IP       Tunnel0                   10.1.1.82(3)
```

```
                                   connectionid 1
                                   0 packets, 0 bytes
                                   epoch 0
                                   sourced in sev-epoch 1
                                   Encap length 28
                                   4500000000000000FF2F7D2B1E010150
                                   1E0101520000883E00000000
                                   Tun endpt
                                   Next chain element:
                                    IP adj out of Ethernet0/0, addr 10.1.1.82
                                    parent oce 0x4BC76A8
                                    frame originated locally (Null0)
                                   L3 mtu 17856
                                   Flags (0x2808C4)
                                   Fixup enabled (0x40000000)
                                         GRE WCCP redirection
                                   HWIDB/IDB pointers 0x55A13E0/0x35F5A80
                                   IP redirect disabled
                                   Switching vector: IPv4 midchain adj oce
                                   IP Tunnel stack to 10.1.1.82 in Default (0x0)
                                    nh tracking enabled: 10.1.1.82/32
                                    IP adj out of Ethernet0/0, addr 10.1.1.82
                                   Adjacency pointer 0x4BC74D8
                                   Next-hop 10.1.1.82
Device#
```

WCCP Bypass Packets

WCCP intercepts IP packets and redirects those packets to a destination other than the destination that is specified in the IP header. Typically the packets are redirected from a web server on the Internet to a web cache that is local to the destination.

Occasionally a web cache cannot manage the redirected packets appropriately and returns the packets unchanged to the originating router. These packets are called bypass packets and are returned to the originating router using either Layer 2 forwarding without encapsulation (L2) or encapsulated in generic routing encapsulation (GRE). The router decapsulates and forwards the packets normally. The VRF associated with the ingress interface (or the global table if there is no VRF associated) is used to route the packet to the destination.

GRE is a tunneling protocol developed by Cisco that encapsulates packet types from a variety of protocols inside IP tunnels, creating a virtual point-to-point link over an IP network.

WCCP Closed Services and Open Services

In applications where packets are intercepted and redirected by a Cisco IOS router to external WCCP client devices, it may be necessary to block the packets for the application when a WCCP client device is not available. This blocking is achieved by configuring a WCCP closed service. When a WCCP service is configured as closed, the packets that fulfill the services, but do not have an active client device, are discarded.

By default, WCCP operates as an open service, wherein communication between clients and servers proceeds normally in the absence of an intermediary device.

The **ip wccp service-list** or the **ipv6 wccp service-list** command can be used for both closed-mode and open-mode services. Use the **service-list** keyword and *service-access-list* argument to register an application protocol type or port number. Use the **mode** keyword to select an open or closed service.

WCCP Service Groups

WCCP is a component of Cisco IOS software that redirects traffic with defined characteristics from its original destination to an alternative destination. The typical application of WCCP is to redirect traffic bound for a remote web server to a local web cache to improve response time and optimize network resource usage.

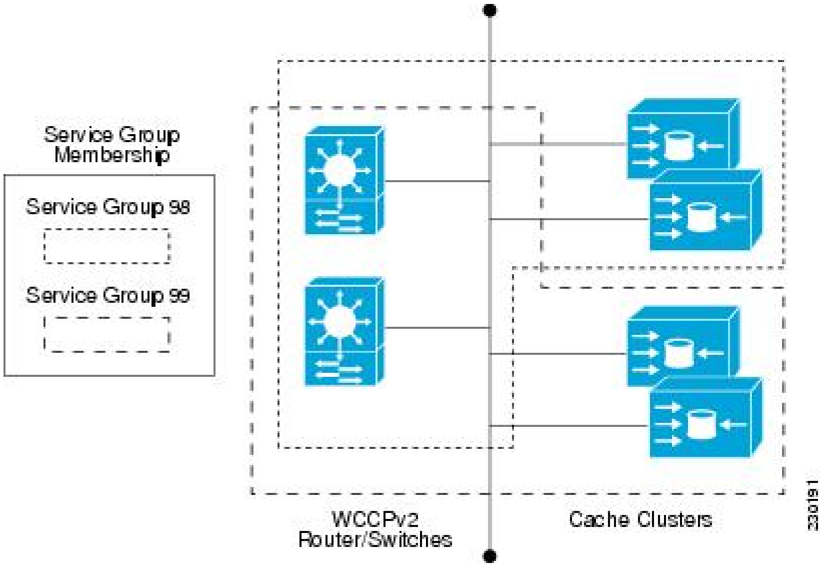
The nature of the selected traffic for redirection is defined by service groups (see figure below) specified on content engines and communicated to routers by using WCCP. The maximum number of service groups allowed across all VRFs is 256.

WCCPv2 supports up to 32 routers per service group. Each service group is established and maintained independently.

WCCPv2 uses service groups based on logical redirection services, deployed for intercepting and redirecting traffic. The standard service is web cache, which intercepts TCP port 80 (HTTP) traffic and redirects that traffic to the content engines. This service is referred to as a *well-known service*, because the characteristics of the web cache service are known by both the router and content engines. A description of a well-known service is not required beyond a service identification. To specify the standard web cache service, use the **ip wccp** or the **ipv6 wccp** command with the **web-cache** keyword.

Note More than one service can run on a router at the same time, and routers and content engines can be part of multiple service groups at the same time.

Figure 8: WCCP Service Groups



The dynamic services are defined by the content engines; the content engine instructs the router which protocol or ports to intercept, and how to distribute the traffic. The router itself does not have information on the characteristics of the dynamic service group's traffic, because this information is provided by the first content engine to join the group. In a dynamic service, up to eight ports can be specified within a single protocol.

Cisco Content Engines, for example, use dynamic service 99 to specify a reverse-proxy service. However, other content engine devices may use this service number for some other service.

WCCP—Check All Services

An interface may be configured with more than one WCCP service. When more than one WCCP service is configured on an interface, the precedence of a service depends on the relative priority of the service compared to the priority of the other configured services. Each WCCP service has a priority value as part of its definition. When an interface is configured with more than one WCCP service, the precedence of the packets is matched against service groups in priority order.

Note The priority of a WCCP service group cannot be configured via Cisco IOS software.

With the **ip wccp check services all** or the **ipv6 wccp check services all** command, WCCP can be configured to check all configured services for a match and perform redirection for those services if appropriate. The caches to which packets are redirected can be controlled by a redirect ACL and by the service priority.

If no WCCP services are configured with a redirect ACL, the services are considered in priority order until a service is found that matches the IP packet. If no services match the packet, the packet is not redirected. If a service matches the packet and the service has a redirect ACL configured, then the IP packet will be checked against the ACL. If the packet is rejected by the ACL, the packet will not be passed down to lower priority services unless the **ip wccp check services all** or the **ipv6 wccp check services all** command is configured. When the **ip wccp check services all** or the **ipv6 wccp check services all** command is configured, WCCP will continue to attempt to match the packet against any remaining lower priority services configured on the interface.

WCCP Interoperability with NAT

To redirect traffic using WCCP to a router running WAAS software that is also configured with NAT, enable the **ip nat inside** or the **ipv6 nat inside** command on the WAAS interface. If you are not able to configure the **ip nat inside** or the**ipv6 nat inside** command on the WAAS interface, disable Cisco Express Forwarding. You must also update the WCCP redirect ACL to include a private address to ensure that pretranslated traffic is redirected.

WCCP Troubleshooting Tips

CPU usage may be very high when WCCP is enabled. The WCCP counters enable a determination of the bypass traffic directly on the router and can indicate whether the cause is high CPU usage due to enablement of WCCP. In some situations, 10 percent bypass traffic may be normal; in other situations, 10 percent may be high. However, any figure above 25 percent should prompt a closer investigation of what is occurring in the web cache.

If the counters suggest that the level of bypass traffic is high, the next step is to examine the bypass counters in the content engine and determine why the content engine is choosing to bypass the traffic. You can log in to the content engine console and use the CLI to investigate further. The counters allow you to determine the percent of traffic being bypassed.

You can use the **clear ipv6 wccp***service-id* command to remove the IPv6 WCCP statistics (counts) maintained on the router for a particular service.

You can use the **clear wccp** command to remove all (IPv4 and IPv6) WCCP statistics (counts) maintained on the router for a particular service.

You can use the **show ipv6 wccp** command to display the IPv6 WCCP global statistics (counts).

You can use the **show wccp** command to display all (IPv4 and IPv6) WCCP global statistics (counts).

How to Configure WCCP

The following configuration tasks assume that you have already installed and configured the content engines you want to include in your network. You must configure the content engines in the cluster before configuring WCCP functionality on your routers or switches. Refer to the Cisco Cache Engine User Guide for content engine configuration and setup tasks.

Configuring WCCP

Perform this task to configure WCCP.

Until you configure a WCCP service using the **ip wccp**{**web-cache** | *service-number*} global configuration command, WCCP is disabled on the router. The first use of a form of the **ip wccp** command enables WCCP. By default WCCPv2 is used for services, but you can use WCCPv1 functionality instead. To change the running version of WCCP from Version 2 to Version 1, or to return to WCCPv2 after an initial change, use the **ip wccp version** command in global configuration mode.

If a function is not allowed in WCCPv1, an error prompt will be printed to the screen. For example, if WCCPv1 is running on the router and you try to configure a dynamic service, the following message will be displayed: “WCCP V1 only supports the web-cache service.” The **show ip wccp** EXEC command will display the WCCP protocol version number that is running on your router.

Use the **ip wccp web-cache password** command to set a password for a router and the content engines in a service group. MD5 password security requires that each router and content engine that wants to join a service group be configured with the service group password. The password must be up to eight characters in length. Each content engine or router in the service group will authenticate the security component in a received WCCP packet immediately after validating the WCCP message header. Packets failing authentication will be discarded.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip wccp version** {**1** | **2**}
4. **ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} [**group-address** *multicast-address*] [**redirect-list** *access-list*] [**group-list** *access-list*] [**password** *password* [**0** | **7**]]
5. **interface** *type number*
6. **ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **redirect** {**in** | **out**}
7. **exit**
8. **interface** *type number*
9. **ip wccp redirect exclude in**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**ip wccp version** {**1** | **2**}
	Example:
	`Router(config)# ip wccp version 2`
		Specifies which version of WCCP to configure on a router.
		• WCCPv2 is the default running version.

Step 4	**ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} [**group-address** *multicast-address*] [**redirect-list** *access-list*] [**group-list** *access-list*] [**password** *password* [**0** | **7**]] **Example:** `Router(config)# ip wccp web-cache password password1`	Specifies a web-cache or dynamic service to enable on the router, specifies a VRF-name to associate with the service group, specifies the IP multicast address used by the service group, specifies any access lists to use, specifies whether to use MD5 authentication, and enables the WCCP service.
Step 5	**interface** *type number* **Example:** `Router(config)# interface ethernet0/0`	Targets an interface number for which the web cache service will run, and enters interface configuration mode.

	Command or Action	Purpose

Step 6	**ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **redirect** {**in** | **out**}
	Example:
	`Router(config-if)# ip wccp web-cache redirect in`
		Enables packet redirection on an outbound or inbound interface using WCCP.
		• As indicated by the **out** and **in** keyword options, redirection can be specified for outbound interfaces or inbound interfaces.

Step 7	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 8	**interface** *type number* **Example:** `Router(config)# interface GigabitEthernet 0/2/0`	Targets an interface number on which to exclude traffic for redirection, and enters interface configuration mode.
Step 9	**ip wccp redirect exclude in** **Example:** `Router(config-if)# ip wccp redirect exclude in`	(Optional) Excludes traffic on the specified interface from redirection.

Configuring Closed Services

Perform this task to specify the number of service groups for WCCP, to configure a service group as a closed or open service, and to optionally specify a check of all services.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Enter one of the following commands:
   - **ip wccp** [**vrf** *vrf-name*] *service-number* [**service-list** *service-access-list* **mode** {**open** | **closed**}]
   - or
   - **ip wccp** [**vrf** *vrf-name*] **web-cache mode** {**open** | **closed**}
4. **ip wccp check services all**
5. **ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*}
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	Enter one of the following commands:
	• **ip wccp** [**vrf** *vrf-name*] *service-number* [**service-list** *service-access-list* **mode** {**open** | **closed**}]
	• or
	• **ip wccp** [**vrf** *vrf-name*] **web-cache mode** {**open** | **closed**}
	Example:
	`Device(config)# ip wccp 90 service-list 120 mode closed`
	or
	`Device(config)# ip wccp web-cache mode closed`
		Configures a dynamic WCCP service as closed or open.
		or
		Configures a web-cache service as closed or open.
		Note When configuring the web-cache service as a closed service, you cannot specify a service access list.
		Note When configuring a dynamic WCCP service as a closed service, you must specify a service access list.

Step 4	**ip wccp check services all**	(Optional) Enables a check of all WCCP services.

	Command or Action	Purpose
	Example: `Device(config)# ip wccp check services all`	• Use this command to configure WCCP to check the other configured services for a match and perform redirection for those services if appropriate. The caches to which packets are redirected can be controlled by the redirect ACL and not just the service description. **Note** The **ip wccp check services all** command is a global WCCP command that applies to all services and is not associated with a single service.

Step 5	**ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*}
	Example:
	`Device(config)# ip wccp 201`
		Specifies the WCCP service identifier.
		• You can specify the standard web-cache service or a dynamic service number from 0 to 255.
		• The maximum number of services that can be specified is 256.

Step 6	**exit** **Example:** `Device(config)# exit`	Exits to privileged EXEC mode.

Registering a Router to a Multicast Address

If you decide to use the multicast address option for your service group, you must configure the router to listen for the multicast broadcasts on an interface.

For network configurations where redirected traffic needs to traverse an intervening router, the router being traversed must be configured to perform IP multicast routing. You must configure the following two components to enable traversal over an intervening router:

- Enable IP multicast routing using the **ip multicast-routing** global configuration command.
- Enable the interfaces to which the cache engines will connect to receive multicast transmissions using the **ip wccp group-listen** interface configuration command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip multicast-routing** [**vrf** *vrf-name*] [**distributed**]
4. **ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **group-address** *multicast-address*
5. **interface** *type number*
6. **ip pim** {**sparse-mode** | **sparse-dense-mode** | **dense-mode** [**proxy-register** {**list** *access-list* | **route-map** *map-name*}]}
7. **ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **group-listen**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip multicast-routing** [**vrf** *vrf-name*] [**distributed**] **Example:** `Device(config)# ip multicast-routing`	Enables IP multicast routing.
Step 4	**ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **group-address** *multicast-address* **Example:** `Device(config)# ip wccp 99 group-address 239.1.1.1`	Specifies the multicast address for the service group.
Step 5	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Enables the interfaces to which the content engines will connect to receive multicast transmissions for which the web cache service will run, and enters interface configuration mode.
Step 6	**ip pim** {**sparse-mode** | **sparse-dense-mode** | **dense-mode** [**proxy-register** {**list** *access-list* | **route-map** *map-name*}]}	(Optional) Enables Protocol Independent Multicast (PIM) on an interface.

	Command or Action	Purpose
	Example: `Device(config-if)# ip pim dense-mode`	**Note** To ensure correct operation of the **ip wccp group-listen** command on Catalyst 6500 series switches and Cisco 7600 series routers, you must enter the **ip pim** command in addition to the **ip wccp group-listen** command.
Step 7	**ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **group-listen** **Example:** `Device(config-if)# ip wccp 99 group-listen`	Configures an interface to enable or disable the reception of IP multicast packets for WCCP.

Using Access Lists for a WCCP Service Group

Perform this task to configure the device to use an access list to determine which traffic should be directed to which content engines.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **access-list** *access-list-number* **remark** *remark*
4. **access-list** *access-list-number* **permit** {*source* [*source-wildcard*] | **any**} [**log**]
5. **access-list** *access-list-number* **remark** *remark*
6. **access-list** *access-list-number* **deny** {*source* [*source-wildcard*] | **any**} | [**log**]
7. Repeat some combination of Steps 3 through 6 until you have specified the sources on which you want to base your access list.
8. **ip wccp** [**vrf** *vrf-name*] **web-cache group-list** *access-list*
9. **ip wccp** [**vrf** *vrf-name*] **web-cache redirect-list** *access-list*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *access-list-number* **remark** *remark* **Example:** `Device(config)# access-list 1 remark Give access to user1`	(Optional) Adds a user-friendly comment about an access list entry. • A remark of up to 100 characters can precede or follow an access list entry.
Step 4	**access-list** *access-list-number* **permit** {*source* [*source-wildcard*] | **any**} [**log**] **Example:** `Device(config)# access-list 1 permit 172.16.5.22 0.0.0.0`	Creates an access list that enables or disables traffic redirection to the cache engine and permits the specified source based on a source address and wildcard mask. • Every access list needs at least one permit statement; it does not need to be the first entry. • Standard IP access lists are numbered 1 to 99 or 1300 to 1999. • If the *source-wildcard* is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source address. • Optionally use the keyword **any** as a substitute for the *source source-wildcard* to specify the source and source wildcard of 0.0.0.0 255.255.255.255. • In this example, host 172.16.5.22 is allowed to pass the access list.
Step 5	**access-list** *access-list-number* **remark** *remark* **Example:** `Device(config)# access-list 1 remark Give access to user1`	(Optional) Adds a user-friendly comment about an access list entry. • A remark of up to 100 characters can precede or follow an access list entry.
Step 6	**access-list** *access-list-number* **deny** {*source* [*source-wildcard*] | **any**} | [**log**] **Example:** `Device(config)# access-list 1 deny 172.16.7.34 0.0.0.0`	Denies the specified source based on a source address and wildcard mask. • If the *source-wildcard* is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source address. • Optionally use the abbreviation any as a substitute for the *source source-wildcard* to specify the source and source wildcard of 0.0.0.0 255.255.255.255. • In this example, host 172.16.7.34 is denied passing the access list.

	Command or Action	Purpose
Step 7	Repeat some combination of Steps 3 through 6 until you have specified the sources on which you want to base your access list.	Remember that all sources not specifically permitted are denied by an implicit **deny** statement at the end of the access list.
Step 8	**ip wccp** [**vrf** *vrf-name*] **web-cache group-list** *access-list* **Example:** `Device(config) ip wccp web-cache group-list 1`	Indicates to the device from which IP addresses of content engines to accept packets.
Step 9	**ip wccp** [**vrf** *vrf-name*] **web-cache redirect-list** *access-list* **Example:** `Device(config)# ip wccp web-cache redirect-list 1`	(Optional) Disables caching for certain clients.

Enabling WCCP Interoperability with NAT

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip nat inside**
5. **ip wccp** *service-number* **redirect in**
6. **exit**
7. **interface** *type number*
8. **ip nat outside**
9. **ip wccp** *service-number* **redirect in**
10. **exit**
11. **interface** *type number*
12. **ip nat inside**
13. **ip wccp redirect exclude in**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface ethernet 1`	Specifies an interface on which to enable NAT and enters interface configuration mode. - This is the LAN-facing interface.
Step 4	**ip nat inside** **Example:** `Router(config-if)# ip nat inside`	Designates that traffic originating from or destined for the interface is subject to NAT and indicates that the interface is connected to the inside network (the network subject to NAT translation).
Step 5	**ip wccp** *service-number* **redirect in** **Example:** `Router(config-if)# ip wccp 61 redirect in`	Enables packet redirection on an inbound interface using WCCP.
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 7	**interface** *type number* **Example:** `Router(config)# interface ethernet 2`	Specifies an interface on which to enable NAT and enters interface configuration mode. - This is the WAN-facing interface.
Step 8	**ip nat outside** **Example:** `Router(config-if)# ip nat outside`	Designates that traffic originating from or destined for the interface is subject to NAT and indicates that the interface is connected to the outside network.

	Command or Action	Purpose
Step 9	**ip wccp** *service-number* **redirect in** **Example:** `Router(config-if)# ip wccp 62 redirect in`	Enables packet redirection on an inbound interface using WCCP.
Step 10	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 11	**interface** *type number* **Example:** `Router(config)# interface ethernet 3`	Specifies an interface on which to enable NAT and enters interface configuration mode. • This is the WAAS-facing interface.
Step 12	**ip nat inside** **Example:** `Router(config-if)# ip nat inside`	Designates that traffic originating from or destined for the interface is subject to NAT and indicates that the interface is connected to the inside network (the network subject to NAT translation).
Step 13	**ip wccp redirect exclude in** **Example:** `Router(config-if)# ip wccp redirect exclude in`	Configures an interface to exclude packets received on an interface from being checked for redirection..

Verifying and Monitoring WCCP Configuration Settings

SUMMARY STEPS

1. **enable**
2. **show ip wccp** [**vrf** *vrf-name*] [**web-cache** |*service-number*] [**detail view**]
3. **show ip interface**
4. **more system:running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip wccp** [**vrf** *vrf-name*] [**web-cache** |*service-number*] [**detail view**] **Example:** `Router# show ip wccp 24 detail`	Displays global information related to WCCP, including the protocol version running, the number of content engines in the router service group, which content engine group is allowed to connect to the router, and which access list is being used. • **vrf** *vrf-name*—(Optional) Virtual routing and forwarding (VRF) instance associated with a service group. • *service-number*—(Optional) Dynamic number of the web-cache service group being controlled by the content engine. The range is from 0 to 99. For web caches that use Cisco Content Engines, the reverse proxy service is indicated by a value of 99. • **web-cache**—(Optional) statistics for the web-cache service. • **detail**—(Optional) other members of a particular service group or web cache that have or have not been detected. • **view**—(Optional) information about a router or all web caches.
Step 3	**show ip interface** **Example:** `Router# show ip interface`	Displays status about whether any **ip wccp redirection** commands are configured on an interface; for example, “Web Cache Redirect is enabled / disabled.”
Step 4	**more system:running-config** **Example:** `Router# more system:running-config`	(Optional) Displays contents of the running configuration file (equivalent to the **show running-config** command).

Configuration Examples for WCCP

Example: Changing the Version of WCCP on a Router

The following example shows how to change the WCCP version from the default of WCCPv2 to WCCPv1, and enabling the web-cache service in WCCPv1:

```
Router# show ip wccp

% WCCP version 2 is not enabled
Router# configure terminal

Router(config)# ip wccp version 1

Router(config)# end
Router# show ip wccp

% WCCP version 1 is not enabled
Router# configure terminal

Router(config)# ip wccp web-cache
Router(config)# end
Router# show ip wccp

Global WCCP information:
    Router information:
        Router Identifier:                   10.4.9.8
        Protocol Version:                    1.0
.
.
.
```

Example: Configuring a General WCCPv2 Session

```
Router# configure terminal
Router(config)# ip wccp web-cache group-address 224.1.1.100 password password1
Router(config)# ip wccp source-interface GigabitEthernet 0/1/0
Router(config)# ip wccp check services all
 Configures a check of all WCCP services.
Router(config)# interface GigabitEthernet 0/1/0
Router(config-if)# ip wccp web-cache redirect in
Router(config-if)# exit
Router(config)# interface GigabitEthernet 0/2/0
Router(config-if)# ip wccp redirect exclude in
Router(config-if)# exit
```

Example: Setting a Password for a Router and Content Engines

```
Router# configure terminal
Router(config)# ip wccp web-cache password password1
```

Example: Configuring a Web Cache Service

```
Device# configure terminal
Device(config)# ip wccp web-cache
Device(config)# interface GigabitEthernet 0/1/0
Device(config-if)# ip wccp web-cache redirect in
Device(config-if)# exit
Device# copy running-config startup-config
```

The following example shows how to configure a session in which redirection of HTTP traffic arriving on Gigabit Ethernet interface 0/1/0 is enabled:

```
Device# configure terminal
Device(config)# interface GigabitEthernet 0/1/0
Device(config-if)# ip wccp web-cache redirect in
Device(config-if)# exit
Device# show ip interface GigabitEthernet 0/1/0
.
.
.
WCCP Redirect inbound is enabled
WCCP Redirect exclude is disabled
.
.
.
```

Example: Running a Reverse Proxy Service

The following example assumes that you are configuring a service group using Cisco cache engines, which use dynamic service 99 to run a reverse proxy service:

```
Router# configure terminal
Router(config)# ip wccp 99
Router(config)# interface gigabitethernet 0/1/0
Router(config-if)# ip wccp 99 redirect out
```

Example: Registering a Router to a Multicast Address

```
Device# configure terminal
Device(config)# ip wccp web-cache group-address 224.1.1.100
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ip wccp web cache group-listen
```

The following example shows a router configured to run a reverse proxy service, using the multicast address of 224.1.1.1. Redirection applies to packets outgoing via Gigabit Ethernet interface 0/1/0:

```
Device# configure terminal
Device(config)# ip wccp 99 group-address 224.1.1.1
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ip wccp 99 redirect out
```

Example: Using Access Lists

To achieve better security, you can use a standard access list to notify the device which IP addresses are valid addresses for a content engine attempting to register with the current device. The following example shows a standard access list configuration session where the access list number is 10 for some sample hosts:

```
Device(config)# access-list 10 permit host 10.1.1.1
Device(config)# access-list 10 permit host 10.1.1.2
Device(config)# access-list 10 permit host 10.1.1.3
Device(config)# ip wccp web-cache group-list 10
```

To disable caching for certain clients, servers, or client/server pairs, you can use WCCP access lists. The following example shows that any requests coming from 10.1.1.1 to 10.3.1.1 will bypass the cache, and that all other requests will be serviced normally:

```
Device(config)# ip wccp web-cache redirect-list 120
Device(config)# access-list 120 deny tcp host 10.1.1.1 any
Device(config)# access-list 120 deny tcp any host 10.3.1.1
Device(config)# access-list 120 permit ip any any
```

The following example configures a device to redirect web-related packets received via Gigabit Ethernet interface 0/1/0, destined to any host except 209.165.200.224:

```
Device(config)# access-list 100 deny ip any host 209.165.200.224
Device(config)# access-list 100 permit ip any any
Device(config)# ip wccp web-cache redirect-list 100
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ip wccp web-cache redirect in
```

Example: Enabling WCCP Interoperability with NAT

```
Router(config)# interface ethernet1 ! This is the LAN-facing interface
Router(config-if)# ip nat inside
Router(config-if)# ip wccp 61 redirect in
Router(config-if)# exit
Router(config)# interface ethernet2 ! This is the WAN-facing interface
Router(config-if)# ip nat outside
Router(config-if)# ip wccp 62 redirect in
Router(config-if)# exit
Router(config)# interface ethernet3 ! This is the WAAS-facing interface
Router(config-if)# ip nat inside
Router(config-if)# ip wccp redirect exclude in
```

Example: Verifying WCCP Settings

The following example shows how to verify your configuration changes by using the **more system:running-config** command in privileged EXEC mode. The following example shows that both the web cache service and dynamic service 99 are enabled on the device:

```
Device# more system:running-config

 Building configuration...
 Current configuration:
 !
 version 12.0
 service timestamps debug uptime
 service timestamps log uptime
 no service password-encryption
 service udp-small-servers
 service tcp-small-servers
 !
 hostname router4
 !
 enable secret 5 $1$nSVy$faliJsVQXVPW.KuCxZNTh1
 enable password password1
 !
 ip subnet-zero
 ip wccp web-cache
 ip wccp 99
 ip domain-name cisco.com
 ip name-server 10.1.1.1
 ip name-server 10.1.1.2
 ip name-server 10.1.1.3
 !
 !
 !
 interface GigabitEthernet0/1/1
 ip address 10.3.1.2 255.255.255.0
 no ip directed-broadcast
 ip wccp web-cache redirect in
 ip wccp 99 redirect in
 no ip route-cache
 no ip mroute-cache
 !
 interface GigabitEthernet0/1/0
 ip address 10.4.1.1 255.255.255.0
 no ip directed-broadcast
 ip wccp 99 redirect in
 no ip route-cache
 no ip mroute-cache
 !
 interface Serial0
 no ip address
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
 shutdown
 !
 interface Serial1
 no ip address
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
 shutdown
 !
 ip default-gateway 10.3.1.1
 ip classless
 ip route 0.0.0.0 0.0.0.0 10.3.1.1
 no ip http server
 !
 !
 !
 line con 0
 transport input none
 line aux 0
 transport input all
 line vty 0 4
 password password1
 login
 !
 end
```

The following example shows how to display global statistics related to WCCP:

```
Device# show ip wccp web-cache detail

WCCP Client information:
WCCP Client ID:      10.1.1.2
Protocol Version:    2.0
State:               Usable
Redirection:         L2
```

```
Packet Return:       L2
Packets Redirected:  0
Connect Time:        00:20:34
Assignment:          MASK
Mask  SrcAddr    DstAddr    SrcPort DstPort
----  -------    -------    ------- -------
0000: 0x00000000 0x00001741 0x0000  0x0000
Value SrcAddr    DstAddr   SrcPort DstPort CE-IP
----- -------    -------   ------- ------- -----
0000: 0x00000000 0x00000000 0x0000 0x0000 0x3C010102 (10.1.1.2)
0001: 0x00000000 0x00000001 0x0000 0x0000 0x3C010102 (10.1.1.2)
0002: 0x00000000 0x00000040 0x0000 0x0000 0x3C010102 (10.1.1.2)
0003: 0x00000000 0x00000041 0x0000 0x0000 0x3C010102 (10.1.1.2)
0004: 0x00000000 0x00000100 0x0000 0x0000 0x3C010102 (10.1.1.2)
0005: 0x00000000 0x00000101 0x0000 0x0000 0x3C010102 (10.1.1.2)
0006: 0x00000000 0x00000140 0x0000 0x0000 0x3C010102 (10.1.1.2)
```

For more information about the **show ip wccp web-cache** command, see the *Cisco IOS IP Application Services Command Reference*.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Cisco ACNS software configuration information	• *Cisco ACNS Software Caching Configuration Guide, Release 4.2* • Cisco ACNS Software listing page on Cisco.com
IP access list overview, configuration tasks, and commands	*Cisco IOS Security Command Reference*
IP addressing and services commands and configuration tasks	• *Cisco IOS IP Addressing Services Configuration Guide* • *Cisco IOS IP Addressing Services Command Reference*
WCCP commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IP Application Services Command Reference*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	—

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	—

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for WCCP

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

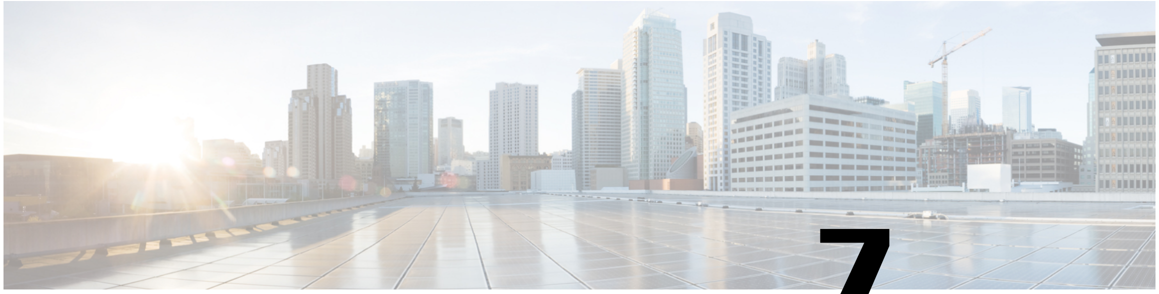
Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 9: Feature Information for WCCP

Feature Name	Releases	Feature Information
WCCP Bypass Counters	15.0(1)SY	The WCCP Bypass Counters feature allows you to display a count of packets that have been bypassed by a web cache and returned to the originating router to be forwarded normally. The **show ip wccp** command was modified by this feature.
WCCP Closed Services	15.0(1)SY	The WCCP Closed Services feature permits WCCP services to be configured so that WCCP always intercepts traffic for such services but, if no WCCP client (such as a content engine) has registered to receive this traffic, packets are discarded. This behavior supports Application-Oriented Network Services (AONS) applications, which require traffic to be transparently intercepted using WCCP but do not want the packets to be forwarded to their destination if the WCCP client is unavailable to perform its processing. (This is contrary to the traditional use of WCCP to assist caches where the absence of a cache does not change the behavior as observed by the user.) The **ip wccp** command was modified by this feature.
WCCP Increased Services	15.0(1)SY	The WCCP Increased Services feature increases the number of services supported by WCCP to a maximum of 256 across all VRFs. The following commands were modified by this feature: **ip wccp**, **ip wccp check services all**, **show ip wccp**.

Feature Name	Releases	Feature Information
WCCP L2 Return	15.0(1)SY	The WCCP L2 Return feature allows content engines to return packets to WCCP routers directly connected at Layer 2 by swapping the source and destination MAC addresses rather than tunneling packets back to the router inside a Layer 3 GRE tunnel. There are no new or modified commands associated with this feature.
WCCP Mask Assignment	15.0(1)SY	The WCCP Mask Assignment feature introduces support for ACNS/WAAS devices using mask assignment as a cache engine assignment method. There are no new or modified commands associated with this feature.
WCCP Redirection on Inbound Interfaces	15.0(1)SY	The WCCP Redirection on Inbound Interfaces feature enables interfaces to be configured for input redirection for a particular WCCP service. When this feature is enabled on an interface, all packets arriving at that interface are compared against the specified WCCP service. If the packets match, they will be redirected. The following commands were introduced or modified by this feature: **ip wccp redirect-list**.

Feature Name	Releases	Feature Information
WCCP Version 2	15.0(1)SY	The WCCP Version 2 feature provides several enhancements and features to the WCCP protocol, including: • The ability of multiple routers to service a content engine cluster. • Redirection of traffic other than HTTP (TCP port 80 traffic), including a variety of UDP and TCP traffic. • Optional authentication that enables you to control which routers and content engines become part of the service group using passwords and the HMAC MD5 standard. • A check on packets that determines which requests have been returned from the content engine unserviced. • Load adjustments for individual content engines to provide an effective use of the available resources while helping to ensure high quality of service (QoS) to the clients. The following commands were introduced or modified by this feature: **clear ip wccp**, **ip wccp**, **ip wccp group-listen**, **ipwccp redirect**, **ip wccp redirect exclude in**, **ip wccp version**, **show ip wccp**.
WCCP VRF Support	15.0(1)SY	The WCCP VRF Support feature provides enhancements to the existing WCCPv2 protocol which support VRF awareness. The following commands were introduced or modified by this feature: **clear ip wccp**, **debug ip wccp**, **ip wccp**, **ip wccp group-listen**, **ip wccp redirect**, **show ip wccp**.



CHAPTER 7

WCCP—Configurable Router ID

The WCCP—Configurable Router ID feature enables the configuration of a Web Cache Communication Protocol (WCCP) source interface. The IP address of this configured source interface is then used as the preferred WCCP router ID and WCCP GRE source address. When a WCCP router ID is manually configured, router IDs are no longer automatically generated when the current router ID is no longer valid and the router ID does not change when another IP address is added to the system.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About WCCP—Configurable Router ID

WCCP—Configurable Router ID Overview

WCCP uses a router ID in its control messages that a WCCP client can use to uniquely identify a particular WCCP server. The router ID is an IP address and is used as the source address of any WCCP-generated

Generic Routing Encapsulation (GRE) frames. Prior to the WCCP—Configurable Router ID feature, WCCP selected a router ID using an automatic mechanism; the highest reachable IP address on the system (or the highest loopback IP address, if there is one) was used as the WCCP router ID. The highest IP address on the system is not always the best choice as the router ID or as the source address of GRE frames. A change in addressing information on the system may cause the WCCP router ID to change unexpectedly. During this changeover period, WCCP clients briefly advertise the existence of two routers (the old router ID and the new router ID) and GRE frames are sourced from a different address.

The WCCP—Configurable Router ID feature enables you to define a WCCP source interface from which the router ID will be obtained. The IP address of this configured source interface is then used as the preferred WCCP router ID and WCCP GRE source address. When a WCCP router ID is manually configured, the router ID does not change when another IP address is added to the system. The router ID changes only when a new router ID is manually configured using the **ip wccp source- interface** or the **ipv6 wccp source- interface** command, or when the address on the manually configured interface is no longer valid.

How to Configure WCCP—Configurable Router ID

Configuring a Preferred WCCP Router ID

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip wccp** [**vrf** *vrf-name*] **source-interface** *source-interface*
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip wccp** [**vrf** *vrf-name*] **source-interface** *source-interface* **Example:** `Device(config)# ip wccp source-interface GigabitEthernet 0/0/0`	Configures a preferred WCCP router ID.

	Command or Action	Purpose
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Configuration Examples for WCCP—Configurable Router ID

Example: Configuring a Preferred WCCP Router ID

The following example displays the configuration for a preferred WCCP router ID:

```
! Configure a preferred WCCP router ID
ip wccp source-interface GigabitEthernet 0/0/0
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Currently assigned IP multicast addresses	*Internet Multicast Addresses* http://www.iana.org/assignments/multicast-addresses
Configuration fundamentals configuration tasks	*Cisco IOS Configuration Fundamentals Configuration Guide*
Configuration fundamentals commands	*Cisco IOS Configuration Fundamentals Command Reference*
Cisco IOS bridging and IBM networking configuration tasks	*Cisco IOS Bridging and IBM Networking Configuration Guide*
Cisco IOS bridging and IBM networking commands	*Cisco IOS Bridging and IBM Networking Command Reference*
Cisco IOS IP multicast configuration tasks	*Cisco IOS IP Multicast Configuration Guide*
Cisco IOS IP Multicast commands	*Cisco IOS IP Multicast Command Reference*

Standards

Standard	Title
IEEE Spanning-Tree Bridging	802.1D MAC Bridges http://www.ieee802.org/1/pages/802.1D-2003.html

MIBs

MIB	MIBs Link
—	No new or modified MIBs are supported, and support for existing MIBs has not been modified.

RFCs

RFC	Title
RFC 1812	*Requirements for IP Version 4 Routers* http://www.ietf.org/rfc/rfc1812.txt
RFC 2131	*Dynamic Host Configuration Protocol* http://www.ietf.org/rfc/rfc2131.txt .

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for WCCP—Configurable Router ID

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 10: Feature Information for WCCP—Configurable Router ID

Feature Name	Releases	Feature Information
WCCP—Configurable Router ID	15.1(1)SG 15.2(3)T Cisco IOS XE Release 3.1S Cisco IOS XE Release 3.3SG	The WCCP—Configurable Router ID feature enables the configuration of a Web Cache Communication Protocol (WCCP) source interface. The IP address of this configured source interface is then used as the preferred WCCP router ID and WCCP GRE source address. When a WCCP router ID is manually configured, router IDs are no longer automatically generated when the current router ID is no longer valid and the router ID does not change when another IP address is added to the system. The following command was added: **ip wccp source-interface**.

CHAPTER 8

WCCP—Fast Timers

The Web Cache Communication Protocol (WCCP)—Fast Timers feature enables WCCP to establish redirection using a configurable message interval when a WCCP client is added to a service group or when a WCCP client fails.

The WCCP message interval capability introduced by the WCCP-Fast Timers feature defines the transmission interval that WCCP clients and WCCP routers use when sending keepalive messages and defines a scaling factor used when calculating the timeout value. The WCCP router uses the timeout value to determine if a WCCP client is no longer available and to redirect traffic as a result.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About WCCP—Fast Timers

WCCP—Fast Timers Overview

The WCCP—Fast Timers feature enables WCCP to establish redirection using a configurable message interval when a WCCP client is added to a service group or when a WCCP client fails. WCCP routers and WCCP clients exchange keepalive messages at a fixed interval. Prior to the introduction of the WCCP—Fast Timers feature, the WCCP message interval was fixed at 10 seconds. The WCCP—Fast Timers feature enables use of message intervals ranging from 0.5 seconds to 60 seconds and a timeout value scaling factor of 1 to 5. The default is 10 seconds. The timer interval is driven by the WCCP client which is being redirected to. The WCCP clients must support variable message interval timers in order for the WCCP—Fast Timers feature to function correctly.

The WCCP message interval capability introduced by the WCCP—Fast Timers feature defines the transmission interval that WCCP clients and WCCP routers use when sending keepalive messages and defines a scaling factor used when calculating the timeout value. The WCCP router uses the timeout value to determine if a WCCP client is no longer available and to redirect traffic as a result. The WCCP router enforces a single message interval per service group. WCCP clients with incompatible message intervals are prevented from joining a service group. If a default message interval that is smaller than the default 10 seconds is used, CPU usage will increase.

You can use the **show ip wccp service** *service-number* **detail** command to display information about the message interval.

How to Configure WCCP—Fast Timers

Displaying WCCP—Fast Timers Information

SUMMARY STEPS

1. **enable**
2. **show ip wccp** [[*service-number*][**detail**]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**show ip wccp** [[*service-number*][**detail**]] **Example:** `Device# show ip wccp 24 detail`	Displays WCCP client information that includes the message interval information. • The message interval is the fixed time interval between successive keepalive messages sent from a WCCP client to a WCCP router. The default time interval is 10 seconds. If the default time interval is configured, the "Message Interval" field is not displayed. **Note** You configure the time interval on the WCCP client device. Details of the client configuration are specific to each type of client, so you should consult the documentation of your WCCP client device. Client devices may choose not support the full range of settings that are supported by the router.

Configuration Examples for WCCP—Fast Timers

Example: Displaying WCCP-Fast Timers Information

The following example displays WCCP client information that includes the message interval information:

```
Device# show ip wccp 91 detail

WCCP Client information:
 WCCP Client ID: 10.1.1.14
 Protocol Version: 2.0
 State: Usable
 Redirection: GRE
 Packet Return: GRE
 Assignment: MASK


Message Interval: 2.500 seconds (2.354 since last message)
 Client timeout: 15 seconds
 Assignment timeout: 25 seconds
 Packets Redirected: 0
 Connect Time: 00:01:56
 Bypassed Packets
 Process: 0
 CEF: 0
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases

Related Topic	Document Title
IP application services commands	Cisco IOS IP Application Services Command Reference

Standards and RFCs

Standard	Title
RFC 1256	ICMP Router Discovery Messages

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for WCCP—Fast Timers

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 11: Feature Information for WCCP—Fast Timers

Feature Name	Releases	Feature Information
WCCP—Fast Timers	15.1(1)SG 15.2(3)T Cisco IOS XE Release 3.3SG	The Web Cache Communication Protocol (WCCP)—Fast Timers feature enables WCCP to establish redirection using a configurable message interval when a WCCP client is added to a service group or when a WCCP client fails. The WCCP message interval capability introduced by the WCCP-Fast Timers feature defines the transmission interval that WCCP clients and WCCP routers use when sending keepalive messages and defines a scaling factor used when calculating the timeout value. The WCCP router uses the timeout value to determine if a WCCP client is no longer available and to redirect traffic as a result. The following command were modified: **show ip wccp**.

CHAPTER 9

WCCPv2—IPv6 Support

This feature introduces support for Web Cache Communication Protocol version 2 (WCCPv2) in an IPv6 environment.

WCCP is a Cisco-developed content-routing technology that intercepts IP packets and redirects those packets to a destination other than that specified in the IP packet. Typically the packets are redirected from their destination web server on the Internet to a content engine that is local to the client. In some WCCP deployment scenarios, redirection of traffic may also be required from the web server to the client. WCCP enables you to integrate content engines into your network infrastructure.

Multiple routers can use WCCPv2 to service a content engine cluster. In WCCPv1, only one router can redirect content requests to a cluster.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for WCCPv2—IPv6 Support

- IPv6 must be configured on the interface used for redirection and on the interface facing the content engine.
- The interface connected to the content engine must be a Fast Ethernet or Gigabit Ethernet interface.

Restrictions for WCCPv2—IPv6 Support

WCCPv2

- For routers servicing a multicast cluster, the Time To Live (TTL) value must be set at 15 or lower.
- Service groups can comprise up to 32 content engines and 32 routers.
- All content engines in a cluster must be configured to communicate with all routers servicing the cluster.
- Multicast addresses must be in the range from 224.0.0.0 to 239.255.255.255.
- Effective from Cisco IOS XE Release 3.10, the Cisco ASR 1000 Series Aggregation Services Routers support hash assignment for IPv6 load balance across content engines, and does not support mask assignment. However, it supports both hash assignment and mask assignment for IPv4.
- Effective from Cisco IOS XE Release 3.7, WCCP Interoperability with Network Address Translation (NAT) is supported.

Layer 2 Forwarding and Return

- Layer 2 redirection requires that content engines be directly connected to an interface on each WCCP router. Unless multicast IP addresses are used, WCCP configuration of the content engine must reference the directly connected interface IP address of the WCCP router and not a loopback IP address or any other IP address configured on the WCCP router.

Information About WCCPv2—IPv6 Support

WCCP Overview

WCCP uses Cisco Content Engines (or other content engines running WCCP) to localize traffic patterns in the network, enabling content requests to be fulfilled locally. Traffic localization reduces transmission costs and download time.

WCCP enables routing platforms to transparently redirect content requests. With transparent redirection, users can fulfill content requests locally without configuring their browsers to use a web proxy. Instead, they can use the target URL to request content, and have their requests automatically redirected to a content engine. The word "transparent" in this case means that the end user does not know that a requested file (such as a web page) came from the content engine instead of from the originally specified server.

A content engine receiving a request attempts to service it from its own local cache. If the requested information is not present, the content engine issues its own request to the originally targeted server to get the required information. A content engine retrieving the requested information forwards it to the requesting client and caches it to fulfill future requests, thus maximizing download performance and substantially reducing transmission costs.

WCCP enables a series of content engines, called a content engine cluster, to provide content to a router or multiple routers. Network administrators can easily scale their content engines to manage heavy traffic loads through these clustering capabilities. Cisco clustering technology enables each cluster member to work in parallel, resulting in linear scalability. Clustering content engines greatly improves the scalability, redundancy, and availability of your caching solution. You can cluster up to 32 content engines to scale to your desired capacity.

Layer 2 Forwarding Redirection and Return

WCCP uses either generic routing encapsulation (GRE) or Layer 2 (L2) to redirect or return IP traffic. When WCCP forwards traffic via GRE, the redirected packets are encapsulated within a GRE header. The packets also have a WCCP redirect header. When WCCP forwards traffic using L2, the original MAC header of the IP packet is overwritten and replaced with the MAC header for the WCCP client.

Using L2 as a forwarding method allows direct forwarding to the content engine without further lookup. Layer 2 redirection requires that the router and content engines are directly connected, that is, on the same IP subnetwork.

When WCCP returns traffic via GRE, the returned packets are encapsulated within a GRE header. The destination IP address is the address of the router and the source address is the address of the WCCP client. When WCCP returns traffic via L2, the original IP packet is returned without any added header information. The router to which the packet is returned will recognize the source of the packet and prevent redirection.

The WCCP redirection method does not have to match the return method.

L2 forwarding, return, or redirection are typically used for hardware-accelerated platforms. Depending on your release, L2 forwarding, return, and redirection can also be used for software-switching platforms.

On Cisco ASR 1000 Series Aggregation Services Routers, both the GRE and L2 forward and return methods use the hardware. Therefore, there is no significant performance degradation between them.

For content engines running Application and Content Networking System (ACNS) software, use the **wccp custom-web-cache** command with the **l2-redirect** keyword to configure L2 redirection. For content engines running Cisco Wide Area Application Services (WAAS) software, use the **wccp tcp-promiscuous** command with the **l2-redirect** keyword to configure L2 redirection.

Note Before configuring a GRE tunnel, configure a loopback interface (that is not attached to a VRF) with an IP address so that the internally created tunnel interface is enabled for IPv4 forwarding by unnumbering itself to this dummy loopback interface. You do not need to configure a loopback interface if the system has at least one interface that is not attached to a VRF and that is configured with an IPv4 address.

For information about Cisco ACNS commands used to configure Cisco Content Engines, see the Cisco ACNS Software Command Reference.

For more information about WAAS commands used to configure Cisco Content Engines, see the Cisco Wide Area Application Services Command Reference.

WCCP Mask Assignment

The WCCP Mask Assignment feature enables mask assignment as the load-balancing method (instead of the default hash assignment method) for a WCCP service.

For content engines running Application and Content Networking System (ACNS) software, use the **wccp custom-web-cache** command with the **mask-assign** keyword to configure mask assignment. For content engines running Cisco Wide Area Application Services (WAAS) software, use the **wccp tcp-promiscuous** command with the **mask-assign** keyword to configure mask assignment.

For information about Cisco ACNS commands used to configure Cisco Content Engines, see the Cisco ACNS Software Command Reference.

For more information about WAAS commands used to configure Cisco Content Engines, see the Cisco Wide Area Application Services Command Reference.

WCCP Hash Assignment

The Cisco ASR 1000 Series Aggregation Services Routers support hash assignment for IPv6 load balance across different content engines, but does not support mask assignment. However, it supports both hash assignment and mask assignment for IPv4.

For content engines running the Cisco Application and Content Networking System (ACNS) software, use the **wccp custom-web-cache** command with the **hash-assign** keyword to configure hash assignment. For content engines running Cisco Wide Area Application Services (WAAS) software, use the **wccp tcp-promiscuous** command with the **hash-assign** keyword to configure hash assignment.

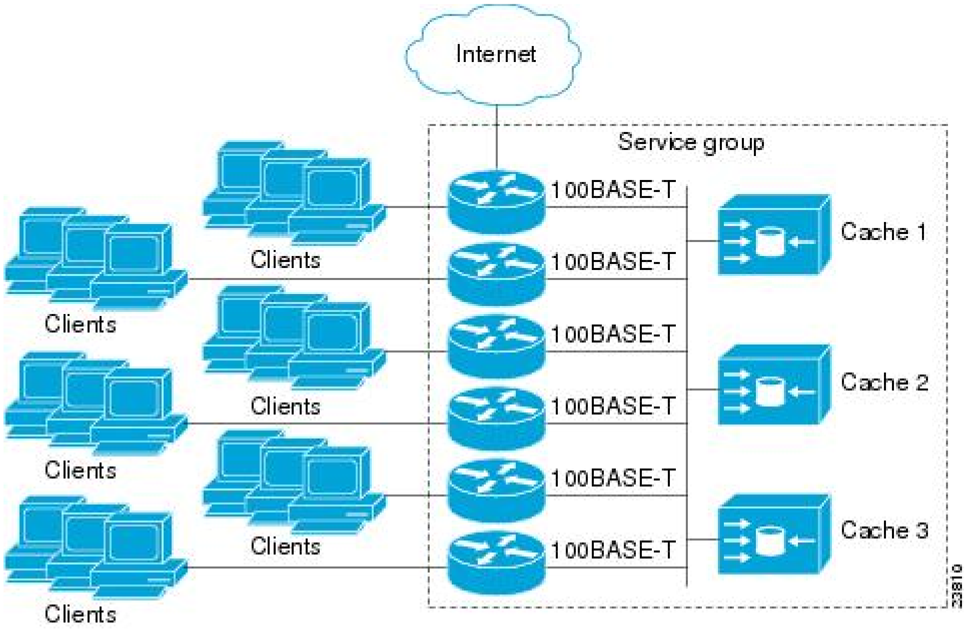
For information about Cisco ACNS commands used to configure Cisco Content Engines, see the Cisco ACNS Software Command Reference.

For more information about WAAS commands used to configure Cisco Content Engines, see the Cisco Wide Area Application Services Command Reference.

WCCPv2 Configuration

Multiple routers can use WCCPv2 to service a content engine cluster. In WCCPv1, only one router could redirect content requests to a cluster. The figure below illustrates a sample configuration using multiple routers.

Figure 9: Cisco Content Engine Network Configuration Using WCCPv2



The subset of content engines within a cluster and routers connected to the cluster that are running the same service is known as a service group. Available services include TCP and UDP redirection.

In WCCPv1, the content engines were configured with the address of the single router. WCCPv2 requires that each content engine be aware of all the routers in the service group. To specify the addresses of all the routers in a service group, you must choose one of the following methods:

- Unicast—A list of router addresses for each of the routers in the group is configured on each content engine. In this case the address of each router in the group must be explicitly specified for each content engine during configuration.
- Multicast—A single multicast address is configured on each content engine. In the multicast address method, the content engine sends a single-address notification that provides coverage for all routers in the service group. For example, a content engine could indicate that packets should be sent to a multicast address of 224.0.0.100, which would send a multicast packet to all routers in the service group configured for group listening using WCCP (see the **ip wccp group-listen** or the **ipv6 wccp group-listen** interface configuration command for details).

The multicast option is easier to configure because you need only specify a single address on each content engine. This option also allows you to add and remove routers from a service group dynamically, without needing to reconfigure the content engines with a different list of addresses each time.

The following sequence of events details how WCCPv2 configuration works:

1 Each content engine is configured with a list of routers.

2 Each content engine announces its presence and a list of all routers with which it has established communications. The routers reply with their view (list) of content engines in the group.

3 When the view is consistent across all content engines in the cluster, one content engine is designated as the lead and sets the policy that the routers need to deploy in redirecting packets.

WCCPv2 Support for Services Other Than HTTP

WCCPv2 allows redirection of traffic other than HTTP (TCP port 80 traffic), including a variety of UDP and TCP traffic. WCCPv2 supports the redirection of packets intended for other ports, including those used for proxy-web cache handling, File Transfer Protocol (FTP) caching, FTP proxy handling, web caching for ports other than 80, and Real Audio, video, and telephony applications.

To accommodate the various types of services available, WCCPv2 introduced the concept of multiple *service groups*. Service information is specified in the WCCP configuration commands using dynamic services identification numbers (such as 98) or a predefined service keyword (such as **web-cache**). This information is used to validate that service group members are all using or providing the same service.

The content engines in a service group specify traffic to be redirected by protocol (TCP or UDP) and up to eight source or destination ports. Each service group has a priority status assigned to it. The priority of a dynamic service is assigned by the content engine. The priority value is in the range of 0 to 255 where 0 is the lowest priority. The predefined web-cache service has an assigned priority of 240.

WCCPv2 Support for Multiple Routers

WCCPv2 allows multiple routers to be attached to a cluster of cache engines. The use of multiple routers in a service group allows for redundancy, interface aggregation, and distribution of the redirection load. WCCPv2 supports up to 32 routers per service group. Each service group is established and maintained independently.

WCCPv2 MD5 Security

WCCPv2 provides optional authentication that enables you to control which routers and content engines become part of the service group using passwords and the Hashed Message Authentication Code—Message Digest (HMAC MD5) standard. Shared-secret MD5 one-time authentication (set using the **ip wccp** [**password** [**0** | **7**] *password*] global configuration command) enables messages to be protected against interception, inspection, and replay.

WCCPv2 Web Cache Packet Return

If a content engine is unable to provide a requested object it has cached due to error or overload, the content engine will return the request to the router for onward transmission to the originally specified destination server. WCCPv2 provides a check on packets that determines which requests have been returned from the content engine unserviced. Using this information, the router can then forward the request to the originally targeted server (rather than attempting to resend the request to the content engine cluster). This process provides error handling transparency to clients.

Typical reasons why a content engine would reject packets and initiate the packet return feature include the following:

- Instances when the content engine is overloaded and has no room to service the packets
- Instances when the content engine is filtering for certain conditions that make caching packets counterproductive (for example, when IP authentication has been turned on)

WCCPv2 Load Distribution

WCCPv2 can be used to adjust the load being offered to individual content engines to provide an effective use of the available resources while helping to ensure high quality of service (QoS) to the clients. WCCPv2 allows the designated content engine to adjust the load on a particular content engine and balance the load across the content engines in a cluster. WCCPv2 uses three techniques to perform load distribution:

- Hot spot handling—Allows an individual hash bucket to be distributed across all the content engines. Prior to WCCPv2, information from one hash bucket could go to only one content engine.
- Load balancing—Allows the set of hash buckets assigned to a content engine to be adjusted so that the load can be shifted from an overwhelmed content engine to other members that have available capacity.
- Load shedding—Enables the router to selectively redirect the load to avoid exceeding the capacity of a content engine.

The use of these hashing parameters prevents one content engine from being overloaded and reduces the potential for bottlenecking.

WCCP VRF Support

The WCCP VRF Support feature enhances the WCCPv2 protocol by implementing support for virtual routing and forwarding (VRF).

The WCCP VRF Support feature allows service groups to be configured on a per-VRF basis in addition to those defined globally.

Along with the service identifier, the VRF of WCCP protocol packets arriving at the router is used to associate cache-engines with a configured service group.

The same VRF must have the interface on which redirection is applied, the interface which is connected to cache engine, and the interface on which the packet would have left if it had not been redirected.

IPv6 WCCP Tunnel Interface

The use of GRE redirection results in the creation of new tunnel interfaces. You can display these tunnel interfaces by entering the **show ipv6 interface brief | include tunnel** command:

```
Device# show ipv6 interface brief | include tunnel

Tunnel0                2001::DB8:1::1      YES unset  up                    up
Tunnel1                2001::DB8:1::1      YES unset  up                    up
Tunnel2                2001::DB8:1::1      YES unset  up                    up
Tunnel3                2001::DB8:1::1      YES unset  up                    up
Device#
```

The tunnel interfaces are automatically created in order to process outgoing GRE-encapsulated traffic for WCCP. The tunnel interfaces appear when a content engine connects and requests GRE redirection. The tunnel interfaces are not created directly by WCCP, but are created indirectly via a tunnel application programming interface (API). WCCP does not have direct knowledge of the tunnel interfaces, but can redirect packets to them, resulting in the appropriate encapsulation being applied to the packets. After the appropriate encapsulation is applied, the packet is then sent to the content engine.

Note The tunnel interfaces are not used to connect with incoming WCCP GRE return packets.

One tunnel is created for each service group that is using GRE redirection. One additional tunnel is created to provide an IP address that allows the other tunnel group interfaces to be unnumbered but still enabled for IPv6.

You can confirm the connection between the tunnels and WCCP by entering the **show tunnel groups wccp** command:

```
Device# show tunnel groups wccp

WCCP : service group 0 in "Default", ver v2, assgnmnt: hash-table
   intf: Tunnel0, locally sourced
 WCCP : service group 317 in "Default", ver v2, assgnmnt: hash-table
   intf: Tunnel3, locally sourced
 WCCP : service group 318 in "Default", ver v2, assgnmnt: hash-table
   intf: Tunnel2, locally sourced
```

You can display additional information about each tunnel interface by entering the **show tunnel interface** *interface-number* command:

```
Device# show tunnel interface t0

Tunnel0
   Mode:multi-GRE/IP, Destination UNKNOWN, Source 2001::DB8:1::2
   Application ID 2: WCCP : service group 0 in "Default", ver v2, assgnmnt: hash-table
   Linestate - current up
   Internal linestate - current up, evaluated up

Device# show tunnel interface t1

Tunnel1
   Mode:multi-GRE/IP, Destination UNKNOWN, Source 2001::DB8:1::1
   Application ID 2: unspecified
   Linestate - current up
   Internal linestate - current up, evaluated up

Device# show tunnel interface t2

Tunnel2
   Mode:multi-GRE/IP, Destination UNKNOWN, Source 2001::DB8:1::1
   Application ID 2: WCCP : service group 318 in "Default", ver v2, assgnmnt: hash-table
   Linestate - current up
   Internal linestate - current up, evaluated up

Device# show tunnel interface t3

Tunnel3
   Mode:multi-GRE/IP, Destination UNKNOWN, Source 2001::DB8:1::1
   Application ID 2: WCCP : service group 317 in "Default", ver v2, assgnmnt: hash-table
   Linestate - current up
   Internal linestate - current up, evaluated up
Device#
```

Note that the service group number shown in the examples is the internal tunnel representation of the WCCP service group number. Group 0 is the web-cache service. To determine the dynamic services, subtract 256 from the displayed service group number to convert to the WCCP service group number. For interfaces that are used for redirection, the source address shown is the WCCP router ID.

You can display information about the connected content engines and encapsulation, including software packet counters, by entering the **show adjacency** [*tunnel-interface*] [**encapsulation**] [**detail**] [**internal**] command:

```
Device# show adjacency t0

Protocol Interface                 Address
IP       Tunnel0                   2001::DB8:1::1(3)

Device# show adjacency t0 encapsulation

Protocol Interface                 Address
IPV6     Tunnel1                   2001:DB8:1::11(2)
  Encap length 48
  6000000000002FFF20010DB801000000
  000000000000000120010DB800010000
  00000000000000110000883E00000000
  Provider: TUNNEL
IPV6     Tunnel1                   2001:DB8:1::12(2)
  Encap length 48
  6000000000002FFF20010DB801000000
  000000000000000120010DB800010000
  00000000000000120000883E00000000
  Provider: TUNNEL

Device# show adjacency t0 detail

Protocol Interface                 Address
IPV6     Tunnel1                   2001:DB8:1::11(2)
                                   0 packets, 0 bytes
                                   epoch 0
                                   sourced in sev-epoch 22
                                   Encap length 48
                                   6000000000002FFF20010DB801000000
                                   000000000000000120010DB800010000
                                   00000000000000110000883E00000000
                                   Tun endpt
                                   Next chain element:
                                    punt

Device# show adjacency t0 internal

Protocol Interface                 Address
IPV6     Tunnel1                   2001:DB8:1::11(2)
                                   0 packets, 0 bytes
                                   epoch 0
                                   sourced in sev-epoch 22
                                   Encap length 48
                                   6000000000002FFF20010DB801000000
                                   000000000000000120010DB800010000
                                   00000000000000110000883E00000000
                                   Tun endpt
                                   Next chain element:
                                    punt
                                    parent oce 0x68C55B00
                                    frame originated locally (Null0)
                                   L3 mtu 0
                                   Flags (0x2808C6)
                                   Fixup disabled
                                   HWIDB/IDB pointers 0x200900DC/0x20090D98
                                   IP redirect disabled
                                   Switching vector: IPv6 midchain adjacency oce
                                   Next-hop cannot be inferred
                                   IP Tunnel stack to 2001:DB8:1::11 in Default (0x0)

Device#
```

WCCP Bypass Packets

WCCP intercepts IP packets and redirects those packets to a destination other than the destination that is specified in the IP header. Typically the packets are redirected from a web server on the Internet to a web cache that is local to the destination.

Occasionally a web cache cannot manage the redirected packets appropriately and returns the packets unchanged to the originating router. These packets are called bypass packets and are returned to the originating router using either Layer 2 forwarding without encapsulation (L2) or encapsulated in generic routing encapsulation (GRE). The router decapsulates and forwards the packets normally. The VRF associated with the ingress interface (or the global table if there is no VRF associated) is used to route the packet to the destination.

GRE is a tunneling protocol developed by Cisco that encapsulates packet types from a variety of protocols inside IP tunnels, creating a virtual point-to-point link over an IP network.

WCCP Closed Services and Open Services

In applications where packets are intercepted and redirected by a Cisco IOS router to external WCCP client devices, it may be necessary to block the packets for the application when a WCCP client device is not available. This blocking is achieved by configuring a WCCP closed service. When a WCCP service is configured as closed, the packets that fulfill the services, but do not have an active client device, are discarded.

By default, WCCP operates as an open service, wherein communication between clients and servers proceeds normally in the absence of an intermediary device.

The **ip wccp service-list** or the **ipv6 wccp service-list** command can be used for both closed-mode and open-mode services. Use the **service-list** keyword and *service-access-list* argument to register an application protocol type or port number. Use the **mode** keyword to select an open or closed service.

WCCP Outbound ACL Check

When WCCP is enabled for redirection on an ingress interface, the packets are redirected by WCCP and instead egress on an interface other than the destination that is specified in the IP header. The packets are still subject to ACLs configured on the ingress interface. However, redirection can cause the packets to bypass the ACL configured on the original egress interface. Packets that would have been dropped because of the ACL configured on the original egress interface can be sent out on the redirect egress interface, which poses a possible security problem. Enabling the WCCP Outbound ACL check feature ensures that redirected packets are subject to any ACL conditions configured on the original egress interface.

WCCP Service Groups

WCCP is a component of Cisco IOS software that redirects traffic with defined characteristics from its original destination to an alternative destination. The typical application of WCCP is to redirect traffic bound for a remote web server to a local web cache to improve response time and optimize network resource usage.

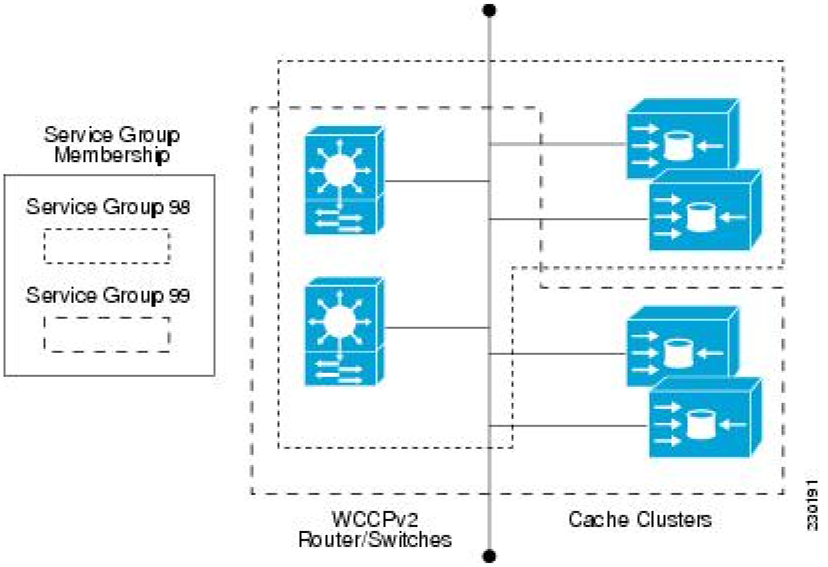
The nature of the selected traffic for redirection is defined by service groups (see figure below) specified on content engines and communicated to routers by using WCCP. The maximum number of service groups allowed across all VRFs is 256.

WCCPv2 supports up to 32 routers per service group. Each service group is established and maintained independently.

WCCPv2 uses service groups based on logical redirection services, deployed for intercepting and redirecting traffic. The standard service is web cache, which intercepts TCP port 80 (HTTP) traffic and redirects that traffic to the content engines. This service is referred to as a *well-known service*, because the characteristics of the web cache service are known by both the router and content engines. A description of a well-known service is not required beyond a service identification. To specify the standard web cache service, use the **ip wccp** or the **ipv6 wccp** command with the **web-cache** keyword.

Note More than one service can run on a router at the same time, and routers and content engines can be part of multiple service groups at the same time.

Figure 10: WCCP Service Groups



The dynamic services are defined by the content engines; the content engine instructs the router which protocol or ports to intercept, and how to distribute the traffic. The router itself does not have information on the characteristics of the dynamic service group's traffic, because this information is provided by the first content engine to join the group. In a dynamic service, up to eight ports can be specified within a single protocol.

Cisco Content Engines, for example, use dynamic service 99 to specify a reverse-proxy service. However, other content engine devices may use this service number for some other service.

WCCP—Check All Services

An interface may be configured with more than one WCCP service. When more than one WCCP service is configured on an interface, the precedence of a service depends on the relative priority of the service compared to the priority of the other configured services. Each WCCP service has a priority value as part of its definition. When an interface is configured with more than one WCCP service, the precedence of the packets is matched against service groups in priority order.

Note The priority of a WCCP service group cannot be configured via Cisco IOS software.

With the **ip wccp check services all** or the **ipv6 wccp check services all** command, WCCP can be configured to check all configured services for a match and perform redirection for those services if appropriate. The caches to which packets are redirected can be controlled by a redirect ACL and by the service priority.

If no WCCP services are configured with a redirect ACL, the services are considered in priority order until a service is found that matches the IP packet. If no services match the packet, the packet is not redirected. If a service matches the packet and the service has a redirect ACL configured, then the IP packet will be checked against the ACL. If the packet is rejected by the ACL, the packet will not be passed down to lower priority services unless the **ip wccp check services all** or the **ipv6 wccp check services all** command is configured. When the **ip wccp check services all** or the **ipv6 wccp check services all** command is configured, WCCP will continue to attempt to match the packet against any remaining lower priority services configured on the interface.

WCCP Interoperability with NAT

To redirect traffic using WCCP to a router running WAAS software that is also configured with NAT, enable the **ip nat inside** or the **ipv6 nat inside** command on the WAAS interface. If you are not able to configure the **ip nat inside** or the**ipv6 nat inside** command on the WAAS interface, disable Cisco Express Forwarding. You must also update the WCCP redirect ACL to include a private address to ensure that pretranslated traffic is redirected.

WCCP—Configurable Router ID Overview

WCCP uses a router ID in its control messages that a WCCP client can use to uniquely identify a particular WCCP server. The router ID is an IP address and is used as the source address of any WCCP-generated Generic Routing Encapsulation (GRE) frames. Prior to the WCCP—Configurable Router ID feature, WCCP selected a router ID using an automatic mechanism; the highest reachable IP address on the system (or the highest loopback IP address, if there is one) was used as the WCCP router ID. The highest IP address on the system is not always the best choice as the router ID or as the source address of GRE frames. A change in addressing information on the system may cause the WCCP router ID to change unexpectedly. During this changeover period, WCCP clients briefly advertise the existence of two routers (the old router ID and the new router ID) and GRE frames are sourced from a different address.

The WCCP—Configurable Router ID feature enables you to define a WCCP source interface from which the router ID will be obtained. The IP address of this configured source interface is then used as the preferred WCCP router ID and WCCP GRE source address. When a WCCP router ID is manually configured, the router ID does not change when another IP address is added to the system. The router ID changes only when a new router ID is manually configured using the **ip wccp source- interface** or the **ipv6 wccp source- interface** command, or when the address on the manually configured interface is no longer valid.

WCCP Troubleshooting Tips

CPU usage may be very high when WCCP is enabled. The WCCP counters enable a determination of the bypass traffic directly on the router and can indicate whether the cause is high CPU usage due to enablement of WCCP. In some situations, 10 percent bypass traffic may be normal; in other situations, 10 percent may be high. However, any figure above 25 percent should prompt a closer investigation of what is occurring in the web cache.

If the counters suggest that the level of bypass traffic is high, the next step is to examine the bypass counters in the content engine and determine why the content engine is choosing to bypass the traffic. You can log in to the content engine console and use the CLI to investigate further. The counters allow you to determine the percent of traffic being bypassed.

You can use the **clear ipv6 wccp***service-id* command to remove the IPv6 WCCP statistics (counts) maintained on the router for a particular service.

You can use the **clear wccp** command to remove all (IPv4 and IPv6) WCCP statistics (counts) maintained on the router for a particular service.

You can use the **show ipv6 wccp** command to display the IPv6 WCCP global statistics (counts).

You can use the **show wccp** command to display all (IPv4 and IPv6) WCCP global statistics (counts).

How to Configure WCCPv2—IPv6 Support

Configuring a General WCCPv2—IPv6 Session

Perform this task to configure a general IPv6 WCCPv2 session.

Until you configure a WCCP service using the **ipv6 wccp** {**web-cache** | *service-number*} global configuration command, WCCP is disabled on the router. The first use of a form of the **ipv6 wccp** command enables WCCP. By default WCCPv2 is used for services.

Using the **ipv6 wccp web-cache password** command, you can set a password for a router and the content engines in a service group. MD5 password security requires that each router and content engine that wants to join a service group be configured with the service group password. The password can be up to eight characters in length. Each content engine or router in the service group will authenticate the security component in a received WCCP packet immediately after validating the WCCP message header. Packets failing authentication will be discarded.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 wccp** [**vrf** *vrf-name*] **source-interface** *source-interface*
4. **ipv6 wccp** [**vrf** *vrf-name*] { **web-cache** | *service-number*} [**group-address** *group-address*] [**redirect-list** *access-list*] [**group-list** *access-list*] [**password** *password* [**0** | **7**]]
5. **interface** *type number*
6. **ipv6 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **redirect** {**out** | **in**}
7. **exit**
8. **interface** *type number*
9. **ipv6 wccp redirect exclude in**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. - Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 wccp** [**vrf** *vrf-name*] **source-interface** *source-interface* **Example:** `Device(config)# ipv6 wccp source-interface GigabitEthernet 0/0/0`	Configures a preferred WCCP router ID.
Step 4	**ipv6 wccp** [**vrf** *vrf-name*] { **web-cache** | *service-number*} [**group-address** *group-address*] [**redirect-list** *access-list*] [**group-list** *access-list*] [**password** *password* [**0** | **7**]] **Example:** `Device(config)# ipv6 wccp web-cache password password1`	Specifies a web-cache or dynamic service to enable on the router, specifies a VRF-name to associate with the service group, specifies the IP multicast address used by the service group, specifies any access lists to use, specifies whether to use MD5 authentication, and enables the WCCP service.
Step 5	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Targets an interface number for which the web cache service will run, and enters interface configuration mode.

Step 6	**ipv6 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **redirect** {**out** | **in**}
	Example:
	`Device(config-if)# ipv6 wccp web-cache redirect in`
		Enables packet redirection on an outbound or inbound interface using WCCP.
		- As indicated by the **out** and **in** keyword options, redirection can be specified for outbound interfaces or inbound interfaces.

Step 7	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.

	Command or Action	Purpose
Step 8	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/2/0`	Targets an interface number on which to exclude traffic for redirection, and enters interface configuration mode.
Step 9	**ipv6 wccp redirect exclude in** **Example:** î `Device(config-if)# ipv6 wccp redirect exclude in`	(Optional) Excludes traffic on the specified interface from redirection.

Configuring Services for WCCPv2—IPv6

Perform this task to specify the number of service groups for WCCP, to configure a service group as a closed or open service, and to optionally specify a check of all services.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Enter one of the following commands:
   - **ipv6 wccp** [**vrf** *vrf-name*] *service-number* [**service-list** *service-access-list* **mode** {**open** | **closed**}]
   - **ipv6 wccp** [**vrf** *vrf-name*] **web-cache mode** {**open** | **closed**}
4. **ipv6 wccp check services all**
5. **ipv6 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*}
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	Enter one of the following commands:
	• **ipv6 wccp** [**vrf** *vrf-name*] *service-number* [**service-list** *service-access-list* **mode** {**open** | **closed**}]
	• **ipv6 wccp** [**vrf** *vrf-name*] **web-cache mode** {**open** | **closed**}
	Example:
	`Device(config)# ipv6 wccp 90 service-list 120 mode closed`
	or
	`Device(config)# ipv6 wccp web-cache mode closed`
		Configures a dynamic WCCP service as closed or open.
		or
		Configures a web-cache service as closed or open.
		Note When configuring the web-cache service as a closed service, you cannot specify a service access list.
		Note When configuring a dynamic WCCP service as a closed service, you must specify a service access list.

Step 4	**ipv6 wccp check services all** **Example:** `Device(config)# ipv6 wccp check services all`	(Optional) Enables a check of all WCCP services. • Use this command to configure WCCP to check the other configured services for a match and perform redirection for those services if appropriate. The caches to which packets are redirected can be controlled by the redirect ACL and not just the service description. **Note** The **ipv6 wccp check services all** command is a global WCCP command that applies to all services and is not associated with a single service.

Step 5	**ipv6 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*}
	Example:
	`Device(config)# ipv6 wccp 201`
		Specifies the WCCP service identifier.
		• You can specify the standard web-cache service or a dynamic service number from 0 to 255.
		• The maximum number of services that can be specified is 256.

Step 6	**exit** **Example:** `Device(config)# exit`	Exits to privileged EXEC mode.

Registering a Router to a Multicast Address for WCCPv2— IPv6

If you decide to use the multicast address option for your service group, you must configure the router to listen for the multicast broadcasts on an interface.

For network configurations where redirected traffic needs to traverse an intervening router, the router being traversed must be configured to perform IP multicast routing. You must configure the following two components to enable traversal over an intervening router:

- Enable IP multicast routing using the **ipv6 multicast-routing** global configuration command.
- Enable the interfaces to which the cache engines will connect to receive multicast transmissions using the **ipv6 wccp group-listen** interface configuration command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 multicast-routing** [**vrf** *vrf-name*] [**distributed**]
4. **ipv6 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **group-address** *multicast-address*
5. **interface** *type number*
6. **ip pim** {**sparse-mode** | **sparse-dense-mode** | **dense-mode** [**proxy-register** {**list** *access-list* | **route-map** *map-name*}]}
7. **ipv6 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **group-listen**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 multicast-routing** [**vrf** *vrf-name*] [**distributed**] **Example:** `Device(config)# ipv6 multicast-routing`	Enables IP multicast routing.
Step 4	**ipv6 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **group-address** *multicast-address*	Specifies the multicast address for the service group.

	Command or Action	Purpose
	Example: `Device(config)# ipv6 wccp 99 group-address FF15::8000:1`	
Step 5	**interface** *type number* **Example:** `Device(config)# interface ethernet 0/0`	Enables the interfaces to which the content engines will connect to receive multicast transmissions for which the web cache service will run, and enters interface configuration mode.

Step 6	**ip pim** {**sparse-mode** | **sparse-dense-mode** | **dense-mode** [**proxy-register** {**list** *access-list* | **route-map** *map-name*}]}
	Example:
	`Device(config-if)# ip pim dense-mode`
		(Optional) Enables Protocol Independent Multicast (PIM) on an interface.
		Note To ensure correct operation of the **ipv6 wccp group-listen** command, you must enter the **ip pim** command in addition to the **ipv6 wccp group-listen** command.

Step 7	**ipv6 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **group-listen** **Example:** `Device(config-if)# ipv6 wccp 99 group-listen`	Configures an interface to enable or disable the reception of IP multicast packets for WCCP.

Using Access Lists for WCCPv2—IPv6 Service Group

Perform this task to configure the router to use an access list to determine which traffic should be directed to which content engines.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **access-list** *access-list-number* **remark** *remark*
4. **access-list** *access-list-number* **permit** {*source* [*source-wildcard*] | **any**} [**log**]
5. **access-list** *access-list-number* **remark** *remark*
6. **access-list** *access-list-number* **deny** {*source* [*source-wildcard*] | **any**} [**log**]
7. Repeat some combination of Steps 3 through 6 until you have specified the sources on which you want to base your access list.
8. **ipv6 wccp** [**vrf** *vrf-name*] **web-cache group-list** *access-list*
9. **ipv6 wccp** [**vrf** *vrf-name*] **web-cache redirect-list** *access-list*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**access-list** *access-list-number* **remark** *remark* **Example:** `Device(config)# access-list 1 remark Give access to user1`	(Optional) Adds a user-friendly comment about an access list entry. • A remark of up to 100 characters in length can precede or follow an access list entry.
Step 4	**access-list** *access-list-number* **permit** {*source* [*source-wildcard*] | **any**} [**log**] **Example:** `Device(config)# access-list 1 permit 172.16.5.22 0.0.0.0`	Creates an access list that enables or disables traffic redirection to the cache engine and permits the specified source based on a source address and wildcard mask. • Every access list needs at least one permit statement; it does not need to be the first entry. • Standard IP access lists are numbered 1 to 99 or 1300 to 1999. • If the *source-wildcard* string is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source address. • Optionally use the keyword **any** as a substitute for the *source source-wildcard* to specify the source and source wildcard of 0.0.0.0 255.255.255.255. • In this example, host 172.16.5.22 is allowed to pass the access list.
Step 5	**access-list** *access-list-number* **remark** *remark* **Example:** `Device(config)# access-list 1 remark Give access to user1`	(Optional) Adds a user-friendly comment about an access list entry. • A remark of up to 100 characters can precede or follow an access list entry.
Step 6	**access-list** *access-list-number* **deny** {*source* [*source-wildcard*] | **any**} [**log**]	Denies the specified source based on a source address and wildcard mask.

	Command or Action	Purpose
	Example: `Device(config)# access-list 1 deny 172.16.7.34 0.0.0.0`	• If the *source-wildcard* string is omitted, a wildcard mask of 0.0.0.0 is assumed, meaning match on all bits of the source address. • Optionally use the abbreviation any as a substitute for the *source source-wildcard* to specify the source and source wildcard of 0.0.0.0 255.255.255.255. • In this example, host 172.16.7.34 is denied passing the access list.
Step 7	Repeat some combination of Steps 3 through 6 until you have specified the sources on which you want to base your access list.	Remember that all sources not specifically permitted are denied by an implicit **deny** statement at the end of the access list.
Step 8	**ipv6 wccp** [**vrf** *vrf-name*] **web-cache group-list** *access-list* **Example:** `Device(config) ipv6 wccp web-cache group-list 1`	Indicates to the router from which IP addresses of content engines to accept packets.
Step 9	**ipv6 wccp** [**vrf** *vrf-name*] **web-cache redirect-list** *access-list* **Example:** `Router(config)# ipv6 wccp web-cache redirect-list 1`	(Optional) Disables caching for certain clients.

Enabling the WCCP—IPv6 Outbound ACL Check

Note When all redirection is performed in the hardware, the mode of redirection will change when outbound ACL checking is enabled. The first packet is switched in software to allow the extra ACL check to be performed before a shortcut is installed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} [**group-address** *multicast-address*] [**redirect-list** *access-list*] [**group-list** *access-list*] [**password** *password*]
4. **ipv6 wccp check acl outbound**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} [**group-address** *multicast-address*] [**redirect-list** *access-list*] [**group-list** *access-list*] [**password** *password*] **Example:** `Device(config)# ipv6 wccp web-cache`	Enables support for a Cisco content engine service group or any content engine service group and configures a redirect ACL list or group ACL.
Step 4	**ipv6 wccp check acl outbound** **Example:** `Device(config)# ipv6 wccp check acl outbound`	Checks the access control list (ACL) for egress interfaces for packets redirected by WCCP.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration.

Enabling WCCPv2—IPv6 Interoperability with NAT

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ipv6 nat inside**
5. **ipv6 wccp** *service-number* **redirect in**
6. **exit**
7. **interface** *type number*
8. **ipv6 nat outside**
9. **ipv6 wccp** *service-number* **redirect in**
10. **exit**
11. **interface** *type number*
12. **ipv6 nat inside**
13. **ipv6 wccp redirect exclude in**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface ethernet 1`	Specifies an interface on which to enable NAT and enters interface configuration mode. • This is the LAN-facing interface.
Step 4	**ipv6 nat inside** **Example:** `Device(config-if)# ipv6 nat inside`	Designates that traffic originating from or destined for the interface is subject to NAT and indicates that the interface is connected to the inside network (the network subject to NAT translation).

	Command or Action	Purpose
Step 5	**ipv6 wccp** *service-number* **redirect in** **Example:** `Device(config-if)# ipv6 wccp 61 redirect in`	Enables packet redirection on an inbound interface using WCCP.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 7	**interface** *type number* **Example:** `Device(config)# interface ethernet 2`	Specifies an interface on which to enable NAT and enters interface configuration mode. • This is the WAN-facing interface.
Step 8	**ipv6 nat outside** **Example:** `Device(config-if)# ipv6 nat outside`	Designates that traffic originating from or destined for the interface is subject to NAT and indicates that the interface is connected to the outside network.
Step 9	**ipv6 wccp** *service-number* **redirect in** **Example:** `Device(config-if)# ipv6 wccp 62 redirect in`	Enables packet redirection on an inbound interface using WCCP.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 11	**interface** *type number* **Example:** `Device(config)# interface ethernet 3`	Specifies an interface on which to enable NAT and enters interface configuration mode. • This is the WAAS-facing interface.
Step 12	**ipv6 nat inside** **Example:** `Device(config-if)# ipv6 nat inside`	Designates that traffic originating from or destined for the interface is subject to NAT and indicates that the interface is connected to the inside network (the network subject to NAT translation).

	Command or Action	Purpose
Step 13	**ipv6 wccp redirect exclude in** **Example:** `Device(config-if)# ipv6 wccp redirect exclude in`	Configures an interface to exclude packets received on an interface from being checked for redirection.

Verifying and Monitoring WCCPv2—IPv6 Configuration Settings

SUMMARY STEPS

1. **enable**
2. **show ipv6 wccp** [**vrf** *vrf-name*] [*service-number* | **web-cache**] [**detail** | **view**]
3. **show ipv6 interface**
4. **more system:running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ipv6 wccp** [**vrf** *vrf-name*] [*service-number* | **web-cache**] [**detail** | **view**] **Example:** `Device# show ipv6 wccp 24 detail`	(Optional) Displays global information related to WCCP, including the protocol version currently running, the number of content engines in the router service group, which content engine group is allowed to connect to the router, and which access list is being used. The argument and keywords are as follows: • *service-number*—(Optional) Dynamic number of the web-cache service group being controlled by the content engine. The range is from 0 to 99. For web caches that use Cisco Content Engines, the reverse proxy service is indicated by a value of 99. • **web-cache**—(Optional) Statistics for the web-cache service. • **detail**—(Optional) Other members of a particular service group or web cache that have or have not been detected. • **view**—(Optional) Information about a router or all web caches.

	Command or Action	Purpose
Step 3	**show ipv6 interface** **Example:** `Device# show ipv6 interface`	(Optional) Displays status about whether any **ip wccp redirection** commands are configured on an interface; for example, "Web Cache Redirect is enabled / disabled."
Step 4	**more system:running-config** **Example:** `Device# more system:running-config`	(Optional) Displays contents of the currently running configuration file (equivalent to the **show running-config** command).

Configuration Examples for WCCPv2—IPv6 Support

Example: Configuring a General WCCPv2—IPv6 Session

```
Device# configure terminal
Device(config)# ipv6 wccp web-cache password password1
Device(config)# ipv6 wccp source-interface GigabitEthernet 0/1/0
Device(config)# ipv6 wccp check services all
 Configures a check of all WCCP services.
Device(config)# interface GigabitEthernet 0/1/0
Device(config-if)# ipv6 wccp web-cache redirect in
Device(config-if)# exit
Device(config)# interface GigabitEthernet 0/2/0
Device(config-if)# ipv6 wccp redirect exclude in
Device(config-if)# exit
```

Example: WCCPv2—IPv6—Setting a Password for a Router and Content Engines

```
Device# configure terminal
Device(config)# ipv6 wccp web-cache password password1
```

Example: WCCPv2—IPv6—Configuring a Web Cache Service

```
Device# configure terminal
Device(config)# ipv6 wccp web-cache
Device(config)# interface GigabitEthernet 0/1/0
Device(config-if)# ipv6 wccp web-cache redirect in
Device(config-if)# exit
Device# copy running-config startup-config
```

The following example shows how to configure a session in which redirection of HTTP traffic arriving on Gigabit Ethernet interface 0/1/0 is enabled:

```
Device# configure terminal
Device(config)# interface GigabitEthernet 0/1/0
Device(config-if)# ipv6 wccp web-cache redirect in
Device(config-if)# exit
Device# show ip interface GigabitEthernet 0/1/0
.
.
.
WCCP Redirect inbound is enabled
WCCP Redirect exclude is disabled
.
.
.
```

Example: WCCPv2—IPv6—Running a Reverse Proxy Service

The following example assumes that you are configuring a service group using Cisco cache engines, which use dynamic service 99 to run a reverse proxy service:

```
Device# configure terminal
Device(config)# ipv6 wccp 99
Device(config)# interface GigabitEthernet 0/1/0
Device(config-if)# ipv6 wccp 99 redirect out
```

Example: WCCPv2—IPv6—Registering a Router to a Multicast Address

```
Device# configure terminal
Device(config)# ipv6 wccp web-cache group-address 224.1.1.100
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ipv6 wccp web cache group-listen
```

The following example shows a device configured to run a reverse proxy service, using the multicast address of 224.1.1.1. Redirection applies to packets outgoing via Gigabit Ethernet interface 0/1/0:

```
Device# configure terminal
Device(config)# ipv6 wccp 99 group-address 224.1.1.1
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ipv6 wccp 99 redirect out
```

Example: WCCPv2—IPv6—Using Access Lists for a WCCPv2 IPv6 Service Group

To achieve better security, you can use a standard access list to notify the device which IP addresses are valid addresses for a content engine attempting to register with the current device. The following example shows a standard access list configuration session where the access list number is 10 for some sample hosts:

```
Device(config)# access-list 10 permit host 10.1.1.1
Device(config)# access-list 10 permit host 10.1.1.2
Device(config)# access-list 10 permit host 10.1.1.3
Device(config)# ipv6 wccp web-cache group-list 10
```

To disable caching for certain clients, servers, or client/server pairs, you can use WCCP access lists. The following example shows that any requests coming from 10.1.1.1 to 10.3.1.1 will bypass the cache, and that all other requests will be serviced normally:

```
Device(config)# ipv6 wccp web-cache redirect-list 120
Device(config)# access-list 120 deny tcp host 10.1.1.1 any
Device(config)# access-list 120 deny tcp any host 10.3.1.1
Device(config)# access-list 120 permit ip any any
```

The following example configures a device to redirect web-related packets received via Gigabit Ethernet interface 0/1/0, destined to any host except 209.165.200.224:

```
Device(config)# access-list 100 deny ip any host 209.165.200.224
Device(config)# access-list 100 permit ip any any
Device(config)# ipv6 wccp web-cache redirect-list 100
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ipv6 wccp web-cache redirect in
```

Example: WCCPv2—IPv6—Configuring Outbound ACL Check

The following configuration example shows that the access list prevents traffic from network 10.0.0.0 leaving Gigabit Ethernet interface 0/1/0. Because the outbound ACL check is enabled, WCCP does not redirect that traffic. WCCP checks packets against the ACL before they are redirected.

```
Device(config)# ipv6 wccp web-cache
Device(config)# ipv6 wccp check acl outbound
Device(config)# interface gigabitethernet 0/1/0
Device(config-if)# ip access-group 10 out
Device(config-if)# exit
Device(config)# ipv6 wccp web-cache redirect-list redirect-out
Device(config)# access-list 10 deny 10.0.0.0 0.255.255.255
Device(config)# access-list 10 permit any
```

If the outbound ACL check is disabled, the HTTP packets from network 10.0.0.0 would be redirected to a web cache. Users with that network address could retrieve web pages even though the network administrator wanted to prevent it.

Example: WCCPv2—IPv6—Enabling WCCP Interoperability with NAT

```
Device(config)# interface ethernet1 ! This is the LAN-facing interface
Device(config-if)# ipv6 nat inside
Device(config-if)# ipv6 wccp 61 redirect in
Device(config-if)# exit
Device(config)# interface ethernet2 ! This is the WAN-facing interface
Device(config-if)# ipv6 nat outside
Device(config-if)# ipv6 wccp 62 redirect in
Device(config-if)# exit
Device(config)# interface ethernet3 ! This is the WAAS-facing interface
Device(config-if)# ipv6 nat inside
Device(config-if)# ipv6 wccp redirect exclude in
```

Example: WCCPv2—IPv6—Verifying WCCP Settings

The following example shows how to verify your configuration changes by using the **more system:running-config** command in privileged EXEC mode. The following example shows that both the web cache service and dynamic service 99 are enabled on the device:

```
Device# more system:running-config

 Building configuration...
 Current configuration:
 !
 version 12.0
 service timestamps debug uptime
 service timestamps log uptime
 no service password-encryption
 service udp-small-servers
 service tcp-small-servers
 !
 hostname router4
 !
 enable secret 5 $1$nSVy$faliJsVQXVPW.KuCxZNTh1
 enable password password1
 !
 ip subnet-zero
 ipv6 wccp web-cache
 ipv6 wccp 99
 ip domain-name cisco.com
 ip name-server 10.1.1.1
 ip name-server 10.1.1.2
 ip name-server 10.1.1.3
 !
 !
 !
 interface GigabitEthernet0/1/1
 ip address 10.3.1.2 255.255.255.0
 no ip directed-broadcast
 ipv6 wccp web-cache redirect in
 ipv6 wccp 99 redirect in
 no ip route-cache
 no ip mroute-cache
 !
 interface GigabitEthernet0/1/0
 ip address 10.4.1.1 255.255.255.0
 no ip directed-broadcast
 ipv6 wccp 99 redirect in
 no ip route-cache
 no ip mroute-cache
 !
 interface Serial0
 no ip address
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
 shutdown
 !
 interface Serial1
 no ip address
 no ip directed-broadcast
 no ip route-cache
 no ip mroute-cache
 shutdown
 !
 ip default-gateway 10.3.1.1
 ip classless
 ip route 0.0.0.0 0.0.0.0 10.3.1.1
 no ip http server
 !
 !
 !
 line con 0
 transport input none
 line aux 0
 transport input all
 line vty 0 4
 password password1
 login
 !
 end
```

The following example shows how to display global statistics related to WCCP:

```
Device# show ipv6 wccp web-cache detail

WCCP Client information:
WCCP Client ID:      10.1.1.2
Protocol Version:    2.0
State:               Usable
Redirection:         L2
Packet Return:       L2
Packets Redirected:  0
Connect Time:        00:20:34
Assignment:          MASK
Mask  SrcAddr    DstAddr    SrcPort DstPort
----  -------    -------    ------- -------
0000: 0x00000000 0x00001741 0x0000  0x0000
Value SrcAddr    DstAddr   SrcPort DstPort CE-IP
----- -------    -------   ------- ------- -----
0000: 0x00000000 0x00000000 0x0000 0x0000 0x3C010102 (10.1.1.2)
0001: 0x00000000 0x00000001 0x0000 0x0000 0x3C010102 (10.1.1.2)
0002: 0x00000000 0x00000040 0x0000 0x0000 0x3C010102 (10.1.1.2)
0003: 0x00000000 0x00000041 0x0000 0x0000 0x3C010102 (10.1.1.2)
0004: 0x00000000 0x00000100 0x0000 0x0000 0x3C010102 (10.1.1.2)
0005: 0x00000000 0x00000101 0x0000 0x0000 0x3C010102 (10.1.1.2)
0006: 0x00000000 0x00000140 0x0000 0x0000 0x3C010102 (10.1.1.2)
```

For more information about the **show ip wccp web-cache** command, see the *Cisco IOS IP Application Services Command Reference* document.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
IP addressing and services commands and configuration tasks	• *Cisco IOS IP Addressing Services Configuration Guide* • *Cisco IOS IP Addressing Services Command Reference*
WCCP commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS IP Application Services Command Reference*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for WCCPv2—IPv6 Support

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 12: Feature Information for WCCPv2 —IPv6 Support

Feature Name	Releases	Feature Information
WCCPv2—IPv6 Support	15.1(1)SY1 15.2(3)T	This feature introduces support for Web Cache Communication Protocol version 2 (WCCPv2) in an IPv6 environment. WCCP is a Cisco-developed content-routing technology that intercepts IP packets and redirects those packets to a destination other than that specified in the IP packet. Multiple routers can use WCCPv2 to service a content engine cluster. In WCCPv1, only one router could redirect content requests to a cluster. The following commands were added: **clear ipv6 wccp**, **clear wccp**, **debug ipv6 wccp**, **debug wccp**, **ipv6 wccp**, **ipv6 wccp check acl outbound**, **ipv6 wccp check services all**, **ipv6 wccp group-listen**, **ipv6 wccp redirect**, **ipv6 wccp redirect exclude in ipv6 wccp source-interface**, **show ipv6 wccp**, **show ipv6 wccp global counters**, **show wccp**, **show wccp global counters**, **show platform software wccp** *service-number* **ipv6 counters**, **show platform software wccp rp active** *service-number* **ipv6** , **show platform software wccp fp active** *service-number* **ipv6** , **show platform hardware qfp active feature wccp service id** *service-number* **ipv6** .

CHAPTER 10

Object Tracking: IPv6 Route Tracking

The Object Tracking: IPv6 Route Tracking feature expands the Enhanced Object Tracking (EOT) functionality to allow the tracking of IPv6 routes.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Restrictions for Object Tracking: IPv6 Route Tracking

Object Tracking: IPv6 Route Tracking is not Stateful Switchover (SSO)-aware and cannot be used with Hot Standby Router Protocol (HSRP), Virtual Router Redundancy Protocol (VRRP), or Gateway Load Balancing Protocol (GLBP) in SSO mode.

Information About Object Tracking: IPv6 Route Tracking

Enhanced Object Tracking and IPv6 Route Tracking

Enhanced Object Tracking (EOT) provides complete separation between the objects to be tracked and the action to be taken by a client when a tracked object changes. Thus, several clients such as Hot Standby Router Protocol (HSRP), Virtual Router Redundancy Protocol (VRRP), or Gateway Load Balancing Protocol (GLBP) can register interest with a tracking process, track the same object, and each take different a action when the object changes.

Each tracked object is identified by a unique number that is specified on the tracking CLI. Client processes use this number to track a specific object.

A tracking process periodically polls tracked objects and notes any change in value. The changes in the tracked object are communicated to interested client processes, either immediately or after a specified delay. The object values are reported as either up or down.

The Object Tracking: IPv6 Route Tracking feature expands EOT functionality to allow the tracking of IPv6 routes.

How to Configure Object Tracking: IPv6 Route Tracking

Tracking the IPv6-Routing State of an Interface

SUMMARY STEPS

1. **track timer interface** {*seconds* | **msec** *milliseconds*}
2. **track** *object-number* **interface** *type number* **ipv6 routing**
3. **carrier-delay**
4. **delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
5. **end**
6. **show track** *object-number*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**track timer interface** {*seconds* | **msec** *milliseconds*}
	Example:
	`Device(config)# track timer interface 5`
		(Optional) Specifies the interval that a tracking process polls the tracked interface.
		• The default interval that the tracking process polls interface objects is 1 second.

	Command or Action	Purpose
		Note All polling frequencies can be configured down to 500 milliseconds, overriding the minimum 1-second interval configured using the **msec** keyword and *milliseconds* argument.
Step 2	**track** *object-number* **interface** *type number* **ipv6 routing** **Example:** `Device(config)# track 1 interface GigabitEthernet 0/0/1 ipv6 routing`	Tracks the IPv6-routing state of an interface and enters tracking configuration mode. • IPv6-route tracking tracks an IPv6 route in the routing table and the ability of an interface to route IPv6 packets.
Step 3	**carrier-delay** **Example:** `Device(config-track)# carrier-delay`	(Optional) Enables enhanced object tracking to consider the carrier-delay timer when tracking the status of an interface.

Step 4	**delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
	Example:
	`Device(config-track)# delay up 30`
		(Optional) Specifies a period of time (in seconds) to delay communicating state changes of a tracked object.
		Note The **up** keyword specifies the time to delay the notification of an up event. The **down** keyword specifies the time to delay the notification of a down event.

Step 5	**end**
	Example:
	`Device(config-track)# end`
		Returns to privileged EXEC mode.

Step 6	**show track** *object-number* **Example:** `Device# show track 1`	Displays tracking information. • Use this command to verify the configuration.

Tracking the Threshold of IPv6-Route Metrics

SUMMARY STEPS

1. **track timer ipv6 route** {*seconds* | **msec** *milliseconds*}
2. **track resolution ipv6 route** {**bgp** | **eigrp** | **isis** | **ospf** | **static** } *resolution-value*
3. **track** *object-number* **ipv6 route** *ipv6-address/prefix-length* **metric threshold**
4. **delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
5. **ipv6 vrf** *vrf-name*
6. **threshold metric** {**up** *number* [**down** *number*] | **down** *number* [**up** *number*]}
7. **end**
8. **show track** *object-number*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**track timer ipv6 route** {*seconds* | **msec** *milliseconds*}
	Example:
	`Device(config)# track timer ipv6 route 20`
		(Optional) Specifies the interval that a tracking process polls the tracked object.
		• The default interval that the tracking process polls IPv6-route objects is 15 seconds.
		Note All polling frequencies can be configured down to 500 milliseconds, overriding the minimum 1-second interval configured using the **msec** keyword and *milliseconds* argument.

Step 2	**track resolution ipv6 route** {**bgp** | **eigrp** | **isis** | **ospf** | **static** } *resolution-value* **Example:** `Device(config)# track resolution ipv6 route eigrp 300`	(Optional) Specifies resolution parameters for a tracked object. • Use this command to change the default metric resolution values.
Step 3	**track** *object-number* **ipv6 route** *ipv6-address/prefix-length* **metric threshold** **Example:** `Device(config)# track 6 ipv6 route 2001:DB8:0:ABCD::1/10 metric threshold`	Tracks the scaled metric value of an IPv6 route to determine if it is above or below a threshold and enters tracking configuration mode. • The default down value is 255, which equates to an inaccessible route. • The default up value is 254.
Step 4	**delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}	(Optional) Specifies a period of time (in seconds) to delay communicating state changes of a tracked object.

	Command or Action	Purpose
	Example: `Device(config-track)# delay up 30`	**Note** The **up** keyword specifies the time to delay the notification of an up event. The **down** keyword specifies the time to delay the notification of a down event.
Step 5	**ipv6 vrf** *vrf-name* **Example:** `Device(config-track)# ipv6 vrf VRF1`	(Optional) Tracks an IPv6 route in a specific VPN virtual routing and forwarding (VRF) table.

Step 6	**threshold metric** {**up** *number* [**down** *number*] | **down** *number* [**up** *number*]}
	Example:
	`Device(config-track)# threshold metric up 254 down 255`
		(Optional) Sets a metric threshold other than the default value.
		Note The **up** keyword specifies the up threshold. The state is up if the scaled metric for that route is less than or equal to the up threshold. The default up threshold is 254. The **down** keyword specifies the down threshold. The state is down if the scaled metric for that route is greater than or equal to the down threshold. The default down threshold is 255.

Step 7	**end**
	Example:
	`Device(config-track)# end`
		Returns to privileged EXEC mode.

Step 8	**show track** *object-number* **Example:** `Device# show track 6`	(Optional) Displays tracking information. • Use this command to verify the configuration.

Tracking IPv6-Route Reachability

Perform this task to track the reachability of an IPv6 route. A tracked object is considered up when a routing table entry exists for the route and the route is accessible.

SUMMARY STEPS

1. **track timer ipv6 route** {*seconds* | **msec** *milliseconds*}
2. **track** *object-number* **ip route** *ip-address/prefix-length* **reachability**
3. **delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
4. **ipv6 vrf** *vrf-name*
5. **end**
6. **show track** *object-number*

DETAILED STEPS

	Command or Action	Purpose

Step 1	**track timer ipv6 route** {*seconds* | **msec** *milliseconds*}
	Example:
	`Device(config)# track timer ipv6 route 20`
		(Optional) Specifies the interval that a tracking process polls the tracked object.
		• The default interval that the tracking process polls IPv6-route objects is 15 seconds.
		Note All polling frequencies can be configured down to 500 milliseconds, overriding the minimum 1-second interval configured using the **msec** keyword and *milliseconds* argument.

Step 2	**track** *object-number* **ip route** *ip-address/prefix-length* **reachability** **Example:** `Device(config)# track 4 ipv6 route 2001:DB8:0:AB82::1/10 reachability`	Tracks the reachability of an IPv6 route and enters tracking configuration mode.

Step 3	**delay** {**up** *seconds* [**down** *seconds*] | [**up** *seconds*] **down** *seconds*}
	Example:
	`Device(config-track)# delay up 30`
		(Optional) Specifies a period of time (in seconds) to delay communicating state changes of a tracked object.
		Note The **up** keyword specifies the time to delay the notification of an up event. The **down** keyword specifies the time to delay the notification of a down event.

Step 4	**ipv6 vrf** *vrf-name* **Example:** `Device(config-track)# ipv6 vrf VRF2`	(Optional) Configures a VPN virtual routing and forwarding (VRF) table.

Step 5	**end**
	Example:
	`Device(config-track)# end`
		Returns to privileged EXEC mode.

Step 6	**show track** *object-number* **Example:** `Device# show track 4`	(Optional) Displays tracking information. • Use this command to verify the configuration.

Configuration Examples for Object Tracking: IPv6 Route Tracking

Example: Tracking the IPv6-Routing State of an Interface

The following example shows how to configure tracking for IPv6 routing on the GigabitEthernet 0/0/1 interface:

```
Device(config)# track timer interface 5
Device(config)# track 1 interface GigabitEthernet 0/0/1 ipv6 routing
Device(config-track)# carrier-delay
Device(config-track)# delay up 30
Device(config-track)# end
```

Example: Tracking the Threshold of IPv6-Route Metrics

The following example shows how to configure tracking for IPv6 metric thresholds:

```
Device(config)# track timer ipv6 route 20
Device(config)# track resolution ipv6 route eigrp 300
Device(config)# track 6 ipv6 route 2001:DB8:0:ABCD::1/10 metric threshold
Device(config-track)# delay up 30
Device(config-track)# ipv6 vrf VRF1
Device(config-track)# threshold metric up 254 down 255
Device(config-track)# end
```

Example: Tracking IPv6-Route Reachability

The following example shows how to configure tracking for IPv6-route reachability:

```
Device(config)# track timer ipv6 route 20
Device(config)# track 4 ipv6 route 2001:DB8:0:AB82::1/10 reachability
Device(config-track)# delay up 30
Device(config-track)# ipv6 vrf VRF2
Device(config-track)# end
```

Additional References for Object Tracking: IPv6 Route Tracking

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Object tracking	*Configuring Enhanced Object Tracking*
IP Application Services commands	*Cisco IOS IP Application Services Command Reference*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

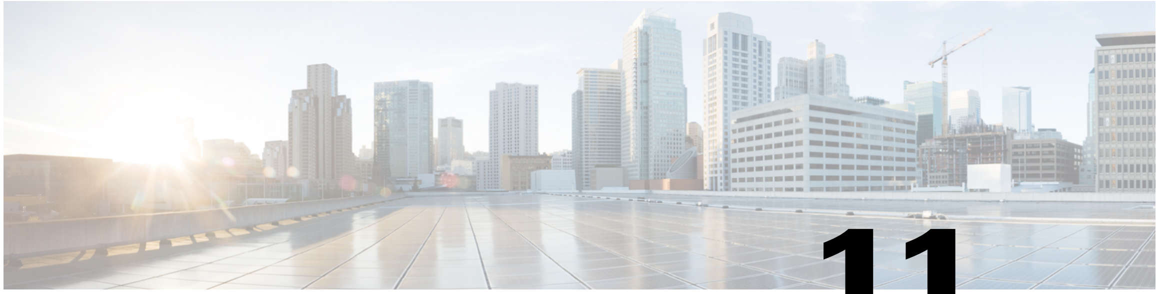
Feature Information for Object Tracking: IPv6 Route Tracking

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 13: Feature Information for Object Tracking: IPv6 Route Tracking

Feature Name	Releases	Feature Information
Object Tracking: IPv6 Route Tracking	Cisco IOS Release 15.2(1)SY Cisco IOS Release 15.2(2)E Cisco IOS Release 15.3(3)M Cisco IOS XE Release 3.6E Cisco IOS XE Release 3.10S	This feature expands Enhanced Object Tracking (EOT) functionality to allow the tracking of IPv6 routes. In Cisco IOS XE Release 3.6E, this feature is supported on the following platforms: • Cisco Catalyst 3650 Series Switches • Cisco Catalyst 3850 Series Switches • Cisco Catalyst 4500E Supervisor Engine 6-E • Cisco Catalyst 4500E Supervisor Engine 6L-E • Cisco Catalyst 4500E Supervisor Engine 7L-E • Cisco Catalyst 4500E Supervisor Engine 8-E • Cisco Catalyst 4900 Series Switches • Cisco 5700 Series Wireless Controllers In Cisco IOS 15.2(2)E, this feature is supported on the following platforms: • Cisco Catalyst 2960 Series Switches • Cisco Catalyst 2960-X Series Switches • Cisco Catalyst 3750 Series Switches In Cisco IOS XE Release 3.10S, this feature was supported on Cisco ASR 1000 Series Aggregation Services Routers. In Cisco IOS Release 15.2(1)SY, this feature was supported on Cisco Catalyst 6500 Series Switches.



CHAPTER 11

IPv6 Static Route Support for Object Tracking

The IPv6 Static Route Support for Object Tracking feature allows an IPv6 static route to be associated with a tracked-object. A static route is only inserted into the routing information base (RIB) when the tracked object is reachable.

This module provides an overview of the feature and explains how to configure it.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Information About IPv6 Static Route Support for Object Tracking

IPv6 Static Route Support for Object Tracking Overview

Object tracking allows you to track specific objects on a device, such as the interface line protocol state, IP routing, and route reachability, and to take action when the tracked object's state changes. Tracking allows software clients to register interest in the behavior of an object, and receive notifications of changes. This object represents the state of the system functionality such as the status of an interface (up or down), the existence of an IP prefix in the Routing Information Base (RIB) and so on.

An IPv6 static route creates a tracked object-context for each tracked object. Tracked object contexts are stored in an AVL list that is maintained by the IPv6 static route and indexed by the object number. A tracked-object context is removed from the AVL list when the object is no longer associated with any IPv6 static routes. All IPv6 static routes associated with a tracked object is linked to the tracked object context by an indirect list. An IPv6 static route becomes a client of the tracked objects, and this allows the IPv6 static route to track the state of a tracked object. The **ipv6 route** command allows an IPv6 static route to be associated with a tracked object.

Routing Table Insertion

An IPv6 static route associated with a tracked-object is inserted into the IPv6 routing table if the state of the tracked-object is up and all other routing-table-insertion criteria are met.

The IPv6 Static Route Object Tracking feature uses the IPv6 static deferred state check mechanism to insert or delete a static route into or from the Routing Information Base (RIB). A change in the state of the tracked object is signaled from tracked objects and this causes IPv6 static to insert all IPv6 static routes associated with the tracked object into the state check queue (unless they are already in it). A separate process removes IPv6 static routes from the state check queue and determines whether these routes should be inserted into the RIB or removed from the RIB using the RIB insertion criteria.

Routing Table Insertion Criteria

The following insertion criteria must be met for an IPv6 static route to be inserted into the IPv6 routing table:

1. Interface is up.
2. Next-hop address is not the device's own address.
3. Next-hop address .
4. Next-hop address is resolved.
5. Bidirectional Forwarding Detection (BFD) session is up, if BFD tracking is configured.

Note An IPv6 static route can be associated with a tracked object and a BFD session. Both tracked object and BFD session state must be up before the IPv6 static route is inserted in the routing table.

6. Tracked object state is up.

An IPv6 static route in the routing table is removed if any of the insertion criteria becomes false.

How to Configure IPv6 Static Route Support for Object Tracking

Configuring the IPv6 Static Routing Support for Object Tracking

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 route vrf** *table-name-id ipv6-prefix* {*interface-type interface-number* [*next-hop-ipv6-address*] | *next-hop-ipv6-address*} [*admin-distance* [*multicast-vrf-distance*]] [**multicast**] [**nexthop-vrf** *table-name-id*] [**unicast**] [**tag** *tag-value*] [**track** *object-number*] **name***static-route*]}
4. **end**
5. **show track** *object-number*
6. **show ipv6 static vrf** *id*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**ipv6 route vrf** *table-name-id ipv6-prefix* {*interface-type interface-number* [*next-hop-ipv6-address*] | *next-hop-ipv6-address*} [*admin-distance* [*multicast-vrf-distance*]] [**multicast**] [**nexthop-vrf** *table-name-id*] [**unicast**] [**tag** *tag-value*] [**track** *object-number*] **name***static-route*]}
	Example:
	`Device(config)# ipv6 route vrf 3 2001:DB8:1:2::/64 fastEthernet0/0 2001:DB8:3:4::1 track 42`
		Establishes static IPv6 routes for all VPN routing and forwarding (VRF) instance tables or a specific VRF table for IPv4 or IPv6 address.
		• Configure the IPv6 static route object tracking to the static route configuration by using the **track** *object-number* command.

Step 4	**end**
	Example:
	`Device(config)# end`
		Exits global configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 5	**show track** *object-number* **Example:** `Device# show track 42`	Displays information about objects that are tracked by the tracking process.
Step 6	**show ipv6 static vrf** *id* **Example:** `Device(config)# show ipv6 static vrf 3`	Displays static routes that are added to the routing-table, and the reasons if a static route is not added.

The following is sample output from the **show track** command:

```
Device# show track 42

Track 42
  IP route 10.21.12.0 255.255.255.0 reachability
  Reachability is Down (no ip route), delayed Up (1 sec remaining) (connected)
    1 change, last change 00:00:24
  Delay up 20 secs, down 10 secs
  First-hop interface is unknown (was Ethernet1/0)
  Tracked by:
    HSRP Ethernet0/0 3
```

Configuration Examples for IPv6 Static Route Support for Object Tracking

Example: IPv6 Static Route Object Tracking

The following example associates the static route 2001:DB8:1:2::/64 with the state of tracked-object number 42. The static route is inserted in the IPv6 routing table if the state of tracked-object number 42 is up.

```
Device> enable
Device# configure terminal
Device(config)# ipv6 route vrf 3 2001:DB8:1:2::/64 fastEthernet0/0 2001:DB8:3:4::1 track
42
Device(config)# end
```

Additional References for IPv6 Static Route Support for Object Tracking

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
IP Application Services commands	Cisco IOS IP Application Services Command Reference
Object tracking	*Configuring Enhanced Object Tracking*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for IPv6 Static Route Support for Object Tracking

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 14: Feature Information for IPv6 Static Route Support for Object Tracking

Feature Name	Releases	Feature Information
IPv6 Static Route Support for Object Tracking	Cisco IOS Release XE3.10S 15.4(1)T Cisco IOS XE Release 3.6E Cisco IOS Release 15.2(2)E Cisco IOS Release 15.2(1)SY	This feature expands Enhanced Object Tracking (EOT) functionality to allow the object tracking for IPv6 static routes. In Cisco IOS XE Release 3.6E, this feature is supported on the following platforms: • Catalyst 3650 Series Switches • Catalyst 3850 Series Switches • Catalyst 4500E Supervisor Engine 6-E • Catalyst 4500E Supervisor Engine 7L-E • Catalyst 4500- XE Series Switches • Cisco 5700 Series Wireless Controllers In Cisco IOS 15.2(2)E, this feature is supported on the following platforms: • Cisco Catalyst 2960 Series Switches • Cisco Catalyst 2960-X Series Switches • Cisco Catalyst 3750 Series Switches In Cisco IOS XE Release 3.10.0S, this feature is supported on Cisco ASR 1000 Series Aggregation Services Routers. In Cisco IOS Release 15.2(1)SY, this feature is supported on Cisco CATALYST 6500 Series Switches.