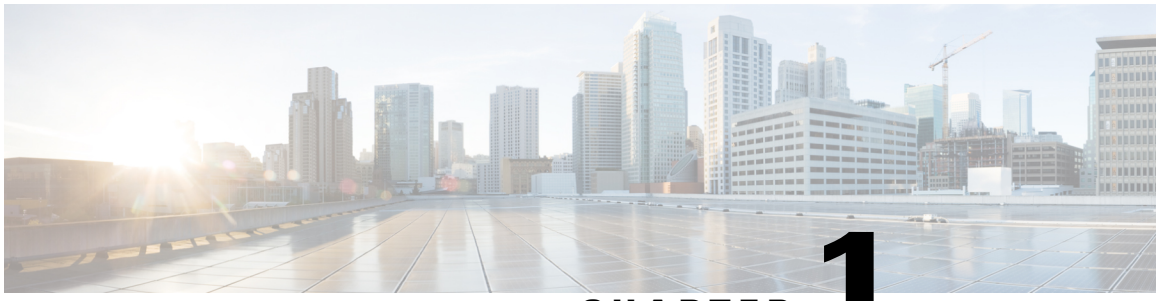
IP Application Services Configuration Guide, Cisco IOS XE Release 3S (Cisco ASR 1000)

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

CONTENTS



CHAPTER 1

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

IP Source Routing

The software examines IP header options on every packet. It supports the IP header options Strict Source Route, Loose Source Route, Record Route, and Time Stamp, which are defined in RFC 791. If the software finds a packet with one of these options enabled, it performs the appropriate action. If it finds a packet with

an invalid option, it sends an Internet Control Message Protocol (ICMP) parameter problem message to the source of the packet and discards the packet.

IP provides a provision known as source routing that allows the source IP host to specify a route through the IP network. Source routing is specified as an option in the IP header. If source routing is specified, the software forwards the packet according to the specified source route. IP source routing is employed when you want to force a packet to take a certain route through the network. The default is to perform source routing. IP source routing is rarely used for legitimate purposes in networks. Some older IP implementations do not process source-routed packets properly, and it may be possible to crash devices running these implementations by sending them datagrams with source routing options. Disable IP source routing whenever possible. Disabling IP source routing will cause a Cisco router to never forward an IP packet that carries a source routing option.

ICMP Overview

Originally created for the TCP/IP suite in RFC 792, the Internet Control Message Protocol (ICMP) was designed to report a small set of error conditions. ICMP can also report a wide variety of error conditions and provide feedback and testing capabilities. Each message uses a common format and is sent and received by using the same protocol rules.

ICMP enables IP to perform addressing, datagram packaging, and routing by allowing encapsulated messages to be sent and received between IP devices. These messages are encapsulated in IP datagrams just like any other IP message. When the message is generated, the original IP header is encapsulated in the ICMP message and these two pieces are encapsulated within a new IP header to be returned as an error report to the sending device.

ICMP messages are sent in several situations: when a datagram cannot reach its destination, when the gateway does not have the buffering capacity to forward a datagram, and when the gateway can direct the host to send traffic on a shorter route. To avoid the infinite regress of messages about messages, no ICMP messages are sent about ICMP messages.

ICMP does not make IP reliable or ensure the delivery of datagrams or the return of a control message. Some datagrams may be dropped without any report of their loss. The higher-level protocols that use IP must implement their own reliability procedures if reliable communication is required.

ICMP Unreachable Error Messages

Type 3 error messages are sent when a message cannot be delivered completely to the application at a destination host. Six codes contained in the ICMP header describe the unreachable condition as follows:

- 0—Network unreachable
- 1—Host unreachable
- 2—Protocol unreachable
- 3—Port unreachable
- 4—Fragmentation needed and the “don’t fragment□? (DF) bit is set
- 5—Source route failed

software can suppress the generation of ICMP unreachable destination error messages, which is called rate-limiting. The default is no unreachable messages more often than once every half second. Separate

intervals can be configured for code 4 and all other unreachable destination error messages. However, there is no method of displaying how many ICMP messages have not been sent.

The ICMP Unreachable Destination Counters feature provides a method to count and display the unsent Type 3 messages. This feature also provides console logging with error messages when there are periods of excessive rate limiting that would indicate a Denial of Service (DoS) attack against the router.

If the software receives a nonbroadcast packet destined for itself that uses an unknown protocol, it sends an ICMP protocol unreachable message back to the source. Similarly, if the software receives a packet that it is unable to deliver to the final destination because it knows of no route to the destination address, it sends an ICMP host unreachable message to the source. This functionality is enabled by default.

Disable ICMP host unreachable messages whenever possible. ICMP supports IP traffic by relaying information about paths, routes, and network conditions. These messages can be used by an attacker to gain network mapping information.

Because the null interface is a packet sink, packets forwarded there will always be discarded and, unless disabled, will generate host unreachable messages. In that case, if the null interface is being used to block a Denial-of-Service attack, these messages flood the local network with these messages. Disabling these messages prevents this situation. In addition, because all blocked packets are forwarded to the null interface, an attacker receiving host unreachable messages could use those messages to determine Access Control List (ACL) configuration. If the “null 0□? interface is configured on your router, disable ICMP host unreachable messages for discarded packets or packets routed to the null interface.

ICMP Mask Reply Messages

Occasionally, network devices must know the subnet mask for a particular subnetwork in the internetwork. To obtain this information, such devices can send ICMP mask request messages. ICMP mask reply messages are sent in reply from devices that have the requested information. The software can respond to ICMP mask request messages if this function is enabled.

These messages can be used by an attacker to gain network mapping information.

ICMP Redirect Messages

Routes are sometimes less than optimal. For example, it is possible for the router to be forced to resend a packet through the same interface on which it was received. If the router resends a packet through the same interface on which it was received, the software sends an ICMP redirect message to the originator of the packet telling the originator that the router is on a subnet directly connected to the receiving device, and that it must forward the packet to another system on the same subnet. The software sends an ICMP redirect message to the originator of the packet because the originating host presumably could have sent that packet to the next hop without involving this device at all. The redirect message instructs the sender to remove the receiving device from the route and substitute a specified device representing a more direct path. This functionality is enabled by default.

In a properly functioning IP network, a router will send redirects only to hosts on its own local subnets, no end node will ever send a redirect, and no redirect will ever be traversed more than one network hop. However, an attacker may violate these rules; some attacks are based on this. Disabling ICMP redirects will cause no operational impact to the network, and it eliminates this possible method of attack.

Denial of Service Attack

Denial of service has become a growing concern, especially when considering the associated costs of such an attack. DoS attacks can decrease the performance of networked devices, disconnect the devices from the network, and cause system crashes. When network services are unavailable, enterprises and service providers suffer the loss of productivity and sales.

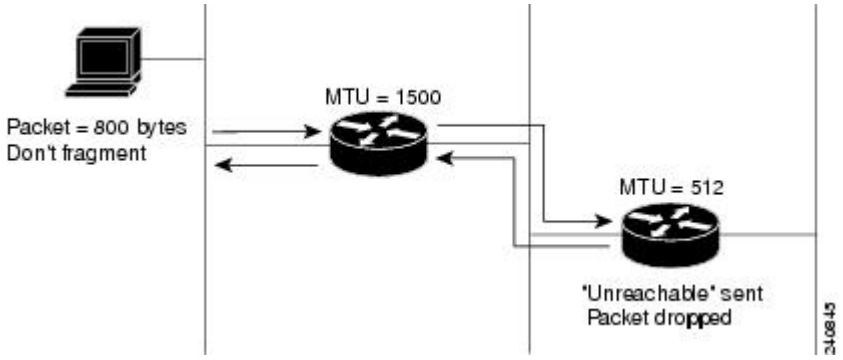
The objective of a DoS attack is to deprive a user or organization access to services or resources. If a Website is compromised by a DoS attack, millions of users could be denied access to the site. DoS attacks do not typically result in intrusion or the illegal theft of information. Instead of providing access to unauthorized users, DoS attacks can cause much aggravation and cost to the target customer by preventing authorized access. Distributed DoS (DDoS) attacks amplify DoS attacks in that a multitude of compromised systems coordinate to flood targets with attack packets, thereby causing denial of service for users of the targeted systems.

A DoS attack occurs when a stream of ICMP echo requests (pings) are broadcast to a destination subnet. The source addresses of these requests are falsified to be the source address of the target. For each request sent by the attacker, many hosts on the subnet will respond flooding the target and wasting bandwidth. The most common DoS attack is called a “smurf” attack, named after an executable program and is in the category of network-level attacks against hosts. DoS attacks can be easily detected when error-message logging of the ICMP Unreachable Destination Counters feature is enabled.

Path MTU Discovery

The software supports the IP Path MTU Discovery mechanism, as defined 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 router is unable to forward a datagram because it requires fragmentation (the packet is larger than the MTU you set for the interface with the **ip mtu** interface configuration command), but the “don’t fragment□? (DF) bit is set. The software sends a message to the sending host, alerting it to the problem. The host will need to fragment packets for the destination so that they fit the smallest packet size of all the links along the path. This technique is shown in the figure below.

Figure 1: IP Path MTU Discovery



IP Path MTU Discovery is useful when a link in a network goes down, forcing the use of another, different MTU-sized link (and different routers). As shown in the figure above, suppose a router is sending IP packets over a network where the MTU in the first router is set to 1500 bytes, but the second router is set to 512 bytes.

If the “don’t fragment□? bit of the datagram is set, the datagram would be dropped because the 512-byte router is unable to forward it. All packets larger than 512 bytes are dropped in this case. The second router returns an ICMP destination unreachable message to the source of the datagram with its Code field indicating “Fragmentation needed and DF set.□? To support IP Path MTU Discovery, it would also include the MTU of the next hop network link in the low-order bits of an unused header field.

IP Path MTU Discovery is also useful when a connection is being established and the sender has no information at all about the intervening links. It is always advisable to use the largest MTU that the links will bear; the larger the MTU, the fewer packets the host must send.

Note IP Path MTU Discovery is a process initiated by end hosts. If an end host does not support IP Path MTU Discovery, the receiving device will have no mechanism to avoid fragmenting datagrams generated by the end host.

If a router that is configured with a small MTU on an outbound interface receives packets from a host that is configured with a large MTU (for example, receiving packets from a Token Ring interface and forwarding them to an outbound Ethernet interface), the router fragments received packets that are larger than the MTU of the outbound interface. Fragmenting packets slows the performance of the router. To keep routers in your network from fragmenting received packets, run IP Path MTU Discovery on all hosts and routers in your network, and always configure the largest possible MTU for each router interface type.

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.

Show and Clear Commands for IOS Sockets

The Show and Clear Commands for IOS Sockets feature introduces the **show udp**, **show sockets**, and **clear sockets**commands. These new commands are useful for monitoring and managing the Cisco IOS Socket library.

In Cisco IOS software, sockets are a per process entity. This means that the maximum number of sockets is per process and all sockets are managed on a per process basis. For example, each Cisco IOS process could have a socket with file descriptor number 1. This is unlike UNIX or other operating systems that have per system file descriptor allocations.

The **show**and **clear** commands operate on a per process basis to be consistent with the current functionality. Thus, any action taken by the commands will be applicable only to a particular process at a time as selected by the process ID entered on the CLI.

Many applications have a need for **show** and **clear**commands, which primarily aid in debugging. The following scenarios provide examples of when these commands might be useful:

- The application H.323 is using sockets for voice calls. According to the current number of calls, there is still space for more sockets. However, no more sockets can be opened. You can now use the the **show sockets** command to find out if the socket space is indeed exhausted or if there are unused sockets available.
- An application is waiting for a particular socket event to happen. A UDP segment was seen, but the application never became active. You can use the **show udp** command to display the list of events being monitored to determine if a UDP socket event is being monitored or if the socket library failed to activate the application.
- An application wants to forcibly close all the sockets for a particular process. You can use the **clear sockets** command to close both the sockets and the underlying TCP or UDP connection or Stream Control Transmission Protocol (SCTP) association.

How to Configure IP Services

Protecting Your Network from DOS Attacks

ICMP supports IP traffic by relaying information about paths, routes, and network conditions. ICMP messages can be used by an attacker to gain network mapping information. IP source routing allows the source IP host to specify a route through the IP network and is rarely used for legitimate purposes in networks. Some older IP implementations do not process source-routed packets properly, and it may be possible to crash devices running these implementations by sending them datagrams with source routing options.

Whenever possible, ICMP messages and IP source routing should be disabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no ip source-route**
4. **interface** *type*/*number*/*slot*
5. **no ip unreachables**
6. **no ip redirects**
7. **no ip mask-reply**

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	**no ip source-route** **Example:** `Device(config)# no ip source-route`	Disables IP source routing.
Step 4	**interface** *type/number/slot* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies the interface to configure and enters interface configuration mode.
Step 5	**no ip unreachables** **Example:** `Device(config-if)# no ip unreachables`	Disables the sending of ICMP protocol unreachable and host unreachable messages. This command is enabled by default. **Note** Disabling the unreachable messages also disables IP Path MTU Discovery because path discovery works by having the software send unreachable messages.
Step 6	**no ip redirects** **Example:** `Device(config-if)# no ip redirects`	Disables the sending of ICMP redirect messages to learn routes. This command is enabled by default.
Step 7	**no ip mask-reply** **Example:** `Device(config-if)# no ip mask-reply`	Disables the sending of ICMP mask reply messages.

Configuring ICMP Unreachable Rate Limiting User Feedback

Perform this task to clear all of the unreachable destination packet statistics and to specify an interval number for unreachable destination messages. This task also configures a packet counter (threshold) and interval to trigger a logging message to a console. This task is beneficial to begin a new log after the thresholds have been set.

SUMMARY STEPS

1. **enable**
2. **clear ip icmp rate-limit** [*interface-type interface-number*]
3. **configure terminal**
4. **ip icmp rate-limit unreachable** [**df**] [*ms*] [**log** [*packets*] [*interval-ms*]]
5. **exit**
6. **show ip icmp rate-limit** [*interface-type interface-number*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**clear ip icmp rate-limit** [*interface-type interface-number*] **Example:** `Router# clear ip icmp rate-limit ethernet 2/3`	Clears all current ICMP unreachable statistics for all configured interfaces. The optional *interface-type* and *interface-number* arguments clear the statistics for only one interface.
Step 3	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 4	**ip icmp rate-limit unreachable** [**df**] [*ms*] [**log** [*packets*] [*interval-ms*]] **Example:** `Router(config)# ip icmp rate-limit unreachable df log 1100 12000`	Specifies the rate limitation of ICMP unreachable destination messages and the error message log threshold for generating a message. The default is no unreachable messages are sent more often than once every half second. The arguments and keywords are as follows: • **df** --(Optional) When “don’t fragment” (DF) bit is set in the ICMP header, a datagram cannot be fragmented. If the **df** keyword is not specified, all other types of destination unreachable messages are sent. • *ms* --(Optional) Interval at which unreachable messages are generated. The valid range is from 1 to 4294967295. • **log** --(Optional) List of error messages. The arguments are as follows: • *packets*--(Optional) Number of packets that determine a threshold for generating a log. The default is 1000.

	Command or Action	Purpose
		• *interval-ms*--(Optional) Time limit for an interval for which a logging message is triggered. The default is 60000, which is 1 minute. **Note** Counting begins as soon as this command is configured.
Step 5	**exit** **Example:** `Router# exit`	Exits to privileged EXEC mode.
Step 6	**show ip icmp rate-limit** [*interface-type interface-number*] **Example:** `Router# show ip icmp rate-limit ethernet 2/3`	(Optional) Displays all current ICMP unreachable statistics for all configured interfaces. The optional *interface-type* and *interface-number* arguments display the statistics for only one interface.

Example

The following output using the**show ip icmp rate-limit** command displays the unreachable destinations by interface:

```
Router# show ip icmp rate-limit
                          DF bit unreachables       All other unreachables
Interval (millisecond)    500                       500
Interface                 # DF bit unreachables     # All other unreachables
---------                 ---------------------     ------------------------
Ethernet0/0               0                         0
Ethernet0/2               0                         0
Serial3/0/3               0                         19
The greatest number of unreachables is on serial interface 3/0/3.
```

Setting the MTU Packet Size

All interfaces have a default MTU packet size. You can adjust the IP MTU size so that the software will fragment any IP packet that exceeds the MTU set for an interface.

Changing the MTU value (with the **mtu** interface configuration command) can affect the IP MTU value. If the current IP MTU value is the same as the MTU value and you change the MTU value, the IP MTU value will be modified automatically to match the new MTU. However, the reverse is not true; changing the IP MTU value has no effect on the value for the **mtu** interface configuration command.

All devices on a physical medium must have the same protocol MTU in order to operate.

Perform this task to set the MTU packet size for a specified interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type/number/slot*
4. **ip mtu** *bytes*
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/slot* **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Specifies the interface to configure and enters interface configuration mode.
Step 4	**ip mtu** *bytes* **Example:** `Device(config-if)# ip mtu 300`	Sets the IP MTU packet size for an interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.

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.
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.

	Command or Action	Purpose
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: Protecting Your Network from DOS Attacks

The following example shows how to change some of the ICMP defaults for Gigabit Ethernet interface 0/0/0 to prevent ICMP from relaying information about paths, routes, and network conditions, which can be used by an attacker to gain network mapping information.

Disabling the unreachable messages will have a secondary effect: it will also disable IP Path MTU Discovery, because path discovery works by having the software send Unreachable messages. If you have a network segment with a small number of devices and an absolutely reliable traffic pattern—which could easily happen on a segment with a small number of rarely used user devices—you would be disabling options that your device would be unlikely to use anyway.

```
Device(config)# no ip source-route
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# no ip unreachables
Device(config-if)# no ip redirects
Device(config-if)# no ip mask-reply
```

Example: Configuring ICMP Unreachable Destination Counters

The following example shows how to clear all of the unreachable destination packet statistics and to specify an interval number for unreachable destination messages. This example also shows how to configure a packet counter threshold and interval to trigger a logging message to a console.

```
Router# clear ip icmp rate-limit ethernet 0/0
Router# configure terminal
Router(config)# ip icmp rate-limit unreachable df log 1100 12000
```

Example: Setting the MTU Packet Size

The following example shows how to change the default MTU packet size for Gigabit Ethernet interface 0/0/0:

```
Device(config)# interface GigabitEthernet 0/0/0
Device(config-if)# ip mtu 300
```

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

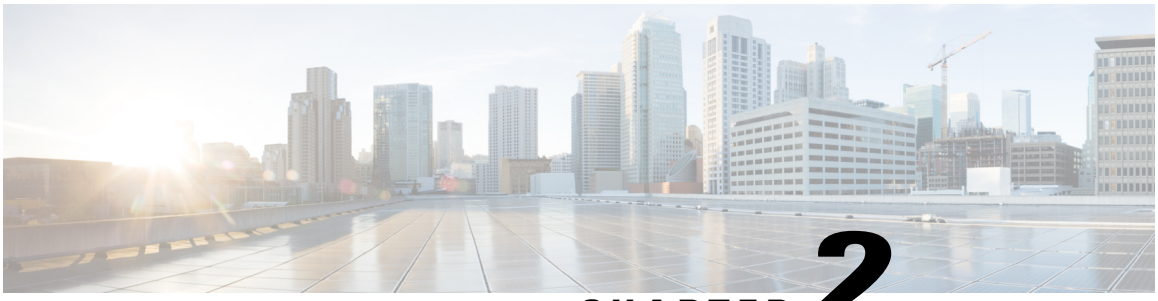
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train 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 1: Feature Information for IP Services

Feature Name	Releases	Feature Information
Clear IP Traffic CLI	12.4(2)T 12.2(31)SB2	The Clear IP Traffic CLI feature introduced the **clear ip traffic**command to clear all IP traffic statistics on a router instead of reloading the router. For added safety, the user will see a confirmation prompt when entering this command. In Cisco IOS Release 12.4(2)T, this feature was introduced. The following command was introduced by this feature: **clear ip traffic**.
ICMP Unreachable Rate Limiting User Feedback	12.4(2)T 12.2(31)SB2	The ICMP Unreachable Rate Limiting User Feedback feature enables you to clear and display packets that have been discarded because of an unreachable destination, and to configure a threshold interval for triggering error messages. When message logging is generated, it displays on your console. In Cisco IOS Release 12.4(2)T, this feature was introduced. The following commands were introduced or modified by this feature: **clear ip icmp rate-limit**, **ip icmp rate-limit unreachable**, **show ip icmp rate-limit**.

Feature Name	Releases	Feature Information
IP Precedence Accounting	12.2(21) 12.1(27b)E1 12.1(5)T15 12.2(25)S 12.2(33)SRA 12.2(18)SXF13 12.2(33)SXH1 15.0(1)S	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 CEF, dCEF, flow, and optimum switching. The following command was introduced by this feature: **show interface precedence**, **ip accounting precedence**.
Show and Clear Commands for IOS Sockets	12.4(11)T	The Show and Clear Commands for IOS Sockets feature introduces the **show udp**, **show sockets**, and **clear sockets**commands. These new commands are useful for monitoring and managing the Cisco IOS Socket library. The following commands were introduced or modified by this feature: **clear sockets**, **show sockets**,**show udp**. The following command was replaced by this feature:**show ip sockets**.



CHAPTER 2

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.

Restrictions for TCP

The TCP Keepalive timer parameters can be configured only on vty and TTY applications.

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 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 Window Scaling

The TCP Window Scaling feature adds support for the Window Scaling option in RFC 1323,*TCP Extensions for High Performance*. A larger window size is recommended to improve TCP performance in network paths with large bandwidth-delay product characteristics that are called Long Fat Networks (LFNs). The TCP Window Scaling enhancement provides LFN support.

The window scaling extension expands the definition of the TCP window to 32 bits and then uses a scale factor to carry this 32-bit value in the 16-bit window field of the TCP header. The window size can increase to a scale factor of 14. Typical applications use a scale factor of 3 when deployed in LFNs.

The TCP Window Scaling feature complies with RFC 1323. The maximum window size was increased to 1,073,741,823 bytes. The larger scalable window size will allow TCP to perform better over LFNs. Use the **ip tcp window-size** command in global configuration mode to configure the TCP window size.

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 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 Congestion Avoidance

The TCP Congestion Avoidance feature enables the monitoring of acknowledgment packets to the TCP sender when multiple packets are lost in a single window of data. Previous to introduction of this feature, the sender would exit Fast-Recovery mode, wait for three or more duplicate acknowledgment packets before retransmitting

the next unacknowledged packet, or wait for the retransmission timer to start slowly. This delay could lead to performance issues.

Implementation of RFC 2581 and RFC 3782 addresses the modifications to the Fast-Recovery algorithm that incorporates a response to partial acknowledgments received during Fast Recovery, improving performance in situations where multiple packets are lost in a single window of data.

This feature is an enhancement to the existing Fast Recovery algorithm. No commands are used to enable or disable this feature.

The output of the **debug ip tcp transactions** command has been enhanced to monitor acknowledgment packets by showing the following conditions:

- TCP entering Fast Recovery mode.
- Duplicate acknowledgments being received during Fast Recovery mode.
- Partial acknowledgments being received.

TCP Explicit Congestion Notification

The TCP Explicit Congestion Notification (ECN) feature allows an intermediate router to notify end hosts of impending network congestion. It also provides enhanced support for TCP sessions associated with applications, such as Telnet, web browsing, and transfer of audio and video data that are sensitive to delay or packet loss. The benefit of this feature is the reduction of delay and packet loss in data transmissions. Use the **ip tcp ecn** command in global configuration mode to enable TCP ECN.

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.

TCP Applications Flags Enhancement

The TCP Applications Flags Enhancement feature enables the user to display additional flags with reference to TCP applications. There are two types of flags: status and option. The status flags indicate the status of TCP connections such as passive open, active open, retransmission timeout, and app closed for listening. The additional flags indicate the state of set options such as whether a VPN routing and forwarding instance (VRF) is set, whether a user is idle, and whether a keepalive timer is running. Use the **show tcp** command to display TCP application flags.

TCP Show Extension

The TCP Show Extension feature introduces the capability to display addresses in IP format instead of the hostname format and to display the VRF table associated with the connection. To display the status for all endpoints with addresses in IP format, use the **show tcp brief numeric** command.

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

TCP Keepalive Timer

The TCP Keepalive Timer feature provides a mechanism to identify dead connections.

When a TCP connection on a routing device is idle for too long, the device sends a TCP keepalive packet to the peer with only the Acknowledgment (ACK) flag turned on. If a response packet (a TCP ACK packet) is not received after the device sends a specific number of probes, the connection is considered dead and the device initiating the probes frees resources used by the TCP connection.

The following parameters are used to configure TCP keepalive:

- TCP Keepalive idle time—The value of this parameter indicates the time for which a TCP connection can be idle before the connection initiates keepalive probes.
- TCP Keepalive retries—The value of this parameter is the number of unacknowledged probes that a device can send before declaring the connection as dead and tearing it down.
- TCP Keepalive interval—The time between subsequent probe retries.

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.

	Command or Action	Purpose
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. **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** **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	**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.

	Command or Action	Purpose
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.

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
```

Configuring Keepalive Parameters

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip tcp keepalive interval** *seconds*
4. **ip tcp keepalive retries** *number-of-retries*
5. **end**
6. **show 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	**configure terminal** **Example:** `Device# configure terminal`	Enables global configuration mode.
Step 3	**ip tcp keepalive interval** *seconds* **Example:** `Device(config)# ip tcp keepalive interval 23`	Configures the keepalive interval.
Step 4	**ip tcp keepalive retries** *number-of-retries* **Example:** `Device(config)# ip tcp keepalive retries 5`	Configures the number of unacknowledged probes that can be sent before declaring the connection as dead.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	(Optional) Displays the running configuration.

Configuration Examples for TCP

Example: Verifying the Configuration of TCP ECN

The following example shows how to verify whether TCP ECN is configured:

```
Device# show running-config

Building configuration...
.
.
.
ip tcp ecn ! ECN is configured.
.
.
.
```

The following example shows how to verify whether TCP is ECN-enabled on a specific connection (local host):

```
Device# show tcp tcb 123456A

!Local host
!
Connection state is ESTAB, I/O status: 1, unread input bytes: 0
Connection is ECN Enabled
Local host: 10.1.25.31, Local port: 11002
Foreign host: 10.1.25.34, Foreign port: 23
```

The following example shows how to display concise information about one address:

```
Device# show tcp brief

!
TCB          Local address            Foreign Address        (state)
609789C      Router.example.com.23      cider.example.com.3733    ESTAB
```

The following example shows how to enable IP TCP ECN debugging:

```
Device# debug ip tcp ecn
!
TCP ECN debugging is on
!
Device# telnet 10.1.25.31

Trying 10.1.25.31 ...
!
01:43:19: 10.1.25.35:11000 <---> 10.1.25.31:23   out ECN-setup SYN
01:43:21: 10.1.25.35:11000 <---> 10.1.25.31:23   congestion window changes
01:43:21: cwnd from 1460 to 1460, ssthresh from 65535 to 2920
01:43:21: 10.1.25.35:11000 <---> 10.1.25.31:23   in non-ECN-setup SYN-ACK
```

Before a TCP connection can use ECN, a host sends an ECN-setup SYN (synchronization) packet to a remote end that contains an Echo Congestion Experience (ECE) and Congestion window reduced (CWR) bit set in the header. Setting the ECE and CWR bits indicates to the remote end that the sending TCP is ECN capable, rather than an indication of congestion. The remote end sends an ECN-setup SYN-ACK (acknowledgment) packet to the sending host.

In this example the “out ECN-setup SYN” text means that a SYN packet with the ECE and CWR bit set was sent to the remote end. The “in non-ECN-setup SYN-ACK” text means that the remote end did not favorably acknowledge the ECN request and, therefore, the session is not ECN capable.

The following output shows that ECN capabilities are enabled at both ends. In response to the ECN-setup SYN, the other end favorably replied with an ECN-setup SYN-ACK message. This connection is now ECN capable for the rest of the session.

```
Device# telnet 10.10.10.10

Trying 10.10.10.10 ... Open
Password required, but none set
!
1d20h: 10.1.25.34:11003 <---> 10.1.25.35:23   out ECN-setup SYN
1d20h: 10.1.25.34:11003 <---> 10.1.25.35:23   in ECN-setup SYN-ACK
```

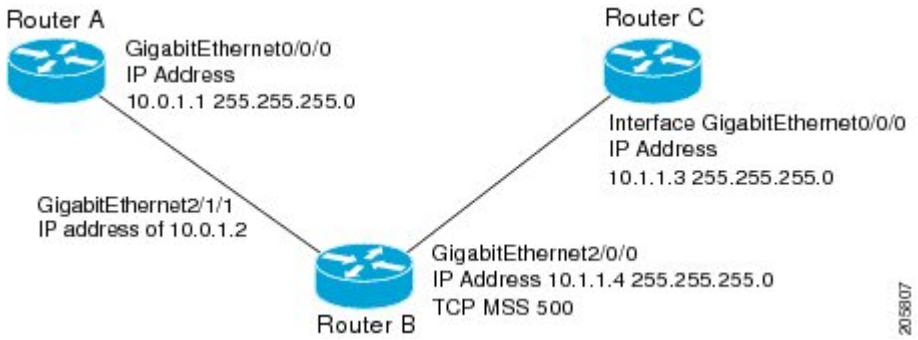
The following example shows how to verify that the hosts are connected:

```
Device# show debugging
!
TCP:
  TCP Packet debugging is on
  TCP ECN debugging is on
!
Device# telnet 10.1.25.234
!
Trying 10.1.25.234 ...
!
00:02:48: 10.1.25.31:11001 <---> 10.1.25.234:23   out ECN-setup SYN
00:02:48: tcp0: O CLOSED 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 ECE CWR SYN  WIN 4128
00:02:50: 10.1.25.31:11001 <---> 10.1.25.234:23   congestion window changes
00:02:50: cwnd from 1460 to 1460, ssthresh from 65535 to 2920
00:02:50: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 ECE CWR SYN  WIN 4128
00:02:54: 10.1.25.31:11001 <---> 10.1.25.234:23   congestion window changes
00:02:54: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:02:54: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 ECE CWR SYN  WIN 4128
00:03:02: 10.1.25.31:11001 <---> 10.1.25.234:23   congestion window changes
00:03:02: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:03:02: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 ECE CWR SYN  WIN 4128
00:03:18: 10.1.25.31:11001 <---> 10.1.25.234:23   SYN with ECN disabled
00:03:18: 10.1.25.31:11001 <---> 10.1.25.234:23   congestion window changes
00:03:18: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:03:18: tcp0: O SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 SYN  WIN 4128
00:03:20: 10.1.25.31:11001 <---> 10.1.25.234:23   congestion window changes
00:03:20: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:03:20: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 SYN  WIN 4128
00:03:24: 10.1.25.31:11001 <---> 10.1.25.234:23   congestion window changes
00:03:24: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:03:24: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 SYN  WIN 4128
00:03:32: 10.1.25.31:11001 <---> 10.1.25.234:23   congestion window changes
00:03:32: cwnd from 1460 to 1460, ssthresh from 2920 to 2920
00:03:32: tcp0: R SYNSENT 10.1.25.234:11001 10.1.25.31:23 seq 1922220018
        OPTS 4 SYN  WIN 4128
!Connection timed out; remote host not responding
```

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 2: 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 192.168.100.0.0.0.255 any
```

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
```

Example: Configuring Keepalive Parameters

The following example shows how to configure TCP keepalive parameters.

```
Device# configure terminal
Device(config)# ip tcp keepalive interval 2
Device(config)# ip tcp keepalive retries 5
```

The following is a sample output of the **show running-config** command:

```
Device# show running-config
```

```
ip tcp keepalive retries 5
ip tcp keepalive interval 2
```

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 2: Feature Information for TCP

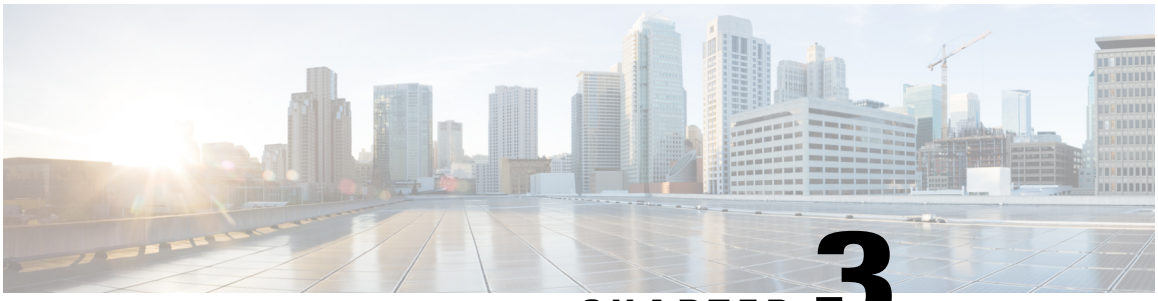
Feature Name	Releases	Feature Information
TCP Application Flags Enhancement	12.2(31)SB2 12.4(2)T	The TCP Applications Flags Enhancement feature enables the user to display additional flags with reference to TCP applications. There are two types of flags: status and option. The status flags indicate the status of TCP connections such as retransmission timeouts, application closed, and synchronized (SYNC) handshakes for listening. The additional flags indicate the state of set options such as whether a VPN routing and forwarding instance (VRF) is set, whether a user is idle, and whether a keepalive timer is running. The following command was modified by this feature: **show tcp**.

Feature Name	Releases	Feature Information
TCP Congestion Avoidance	12.3(7)T	The TCP Congestion Avoidance feature enables the monitoring of acknowledgment packets to the TCP sender when multiple packets are lost in a single window of data. Before this feature was introduced, the sender would exit Fast-Recovery mode, wait for three or more duplicate acknowledgment packets before retransmitting the next unacknowledged packet, or wait for the retransmission timer to start slowly. This delay could lead to performance issues. Implementation of RFC 2581 and RFC 3782 addresses the modifications to the Fast-Recovery algorithm that incorporates a response to partial acknowledgments received during Fast Recovery, improving performance in situations where multiple packets are lost in a single window of data. This feature is an enhancement to the existing Fast Recovery algorithm. No commands are used to enable or disable this feature. The output of the **debug ip tcp transactions** command monitors acknowledgment packets by displaying the following conditions: • TCP entering Fast Recovery mode. • Duplicate acknowledgments being received during Fast Recovery mode. • Partial acknowledgments being received. The following command was modified by this feature: **debug ip tcp transactions**.

Feature Name	Releases	Feature Information
TCP Explicit Congestion Notification	12.3(7)T	The TCP Explicit Congestion Notification (ECN) feature allows an intermediate router to notify end hosts of impending network congestion. It also provides enhanced support for TCP sessions associated with applications such as Telnet, web browsing, and transfer of audio and video data, that are sensitive to delay or packet loss. The benefit of this is the reduction of delay and packet loss in data transmissions. The following commands were introduced or modified by this feature: **debug ip tcp ecn**, **ip tcp ecn**, **show debugging**, **show tcp**.
TCP MIB for RFC4022 Support	12.2(33)XN	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	12.2(4)T 12.2(8)T 12.2(18)ZU2 12.2(28)SB 12.2(33)SRA 12.2(33)SXH 15.0(1)S	The TCP MSS Adjust feature enables the configuration of the maximum segment size (MSS) for transient packets that traverse a device, specifically TCP segments in the SYN bit set. In 12.2(4)T, this feature was introduced. In 12.2(8)T, the command that was introduced by this feature was changed from **ip adjust-mss** to **ip tcp adjust-mss**. In 12.2(28)SB and 12.2(33)SRA, this feature was enhanced to be configurable on subinterfaces. The following command was introduced by this feature: **ip tcp adjust-mss**.
TCP Show Extension	12.2(31)SB2 12.4(2)T	The TCP Show Extension feature introduces the capability to display addresses in IP format instead of hostname format and to display the VRF table associated with the connection. The following command was modified by this feature: **show tcp brief**.
TCP Window Scaling	12.2(8)T 12.2(31)SB2	The TCP Window Scaling feature adds support for the Window Scaling option in RFC 1323. A larger window size is recommended to improve TCP performance in network paths with large bandwidth, long-delay characteristics that are called Long Fat Networks (LFNs). This TCP Window Scaling enhancement provides that support. The following command was introduced or modified by this feature: **ip tcp window-size**.

Feature Name	Releases	Feature Information
TCP Keepalive Timer	15.2(4)M	The TCP Keepalive Timer feature introduces the capability to identify dead connections between multiple routing devices. The following command was introduced or modified by this feature: **ip tcp keepalive**.



CHAPTER 3

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 WCCPv1 and WCCPv2:

- WCCP works only with IPv4 networks.
- WCCP bypasses Network Address Translation (NAT) when Cisco Express Forwarding is enabled.

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:

- 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.

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.

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.

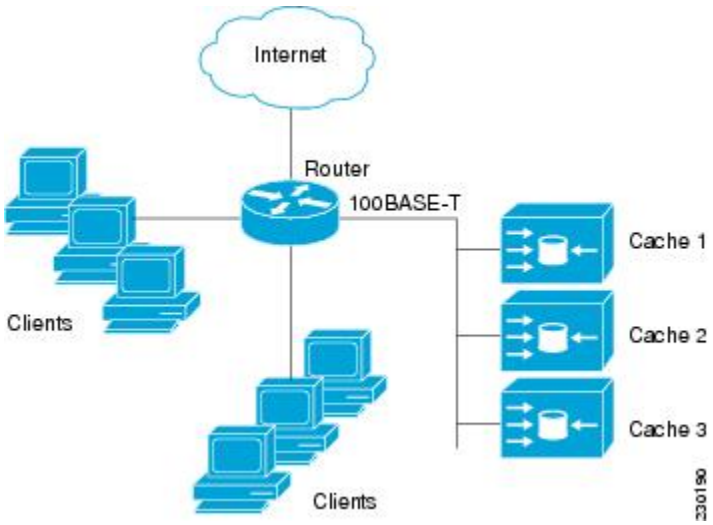
The following guidelines apply to WCCP Layer 2 PFC redirection:

- 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 3: 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.

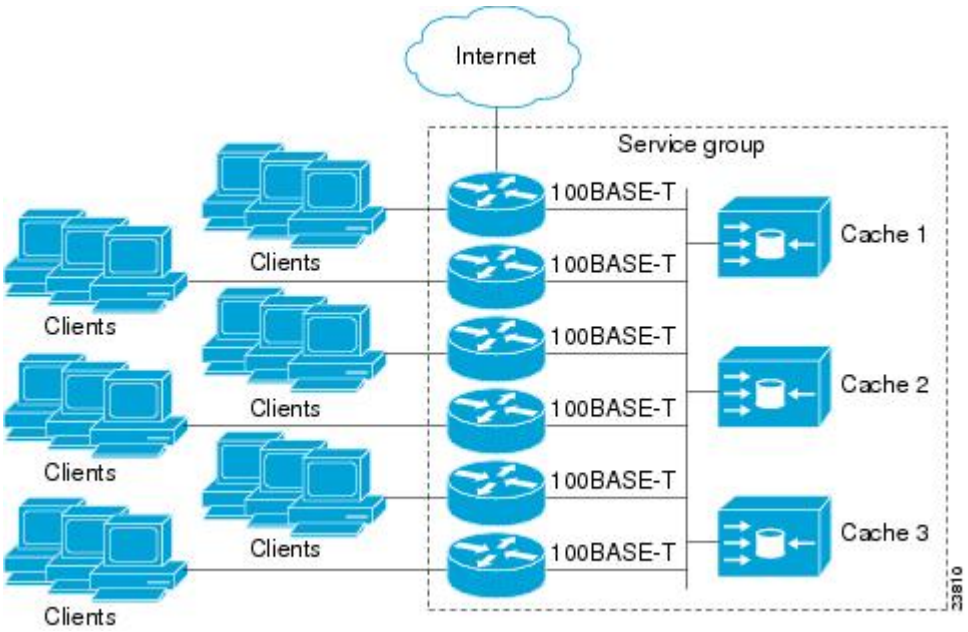
The following sequence of events details how WCCPv1 configuration works:

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 4: 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. WCCPv1 supported the redirection of HTTP (TCP port 80) traffic only. 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.

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 only for closed-mode services. Use the **service-list** keyword and *service-access-list* argument to register an application protocol type or port number.

When there is a mismatch between the service list ACL and the definition received from a cache engine, the service is not allowed to start.

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.

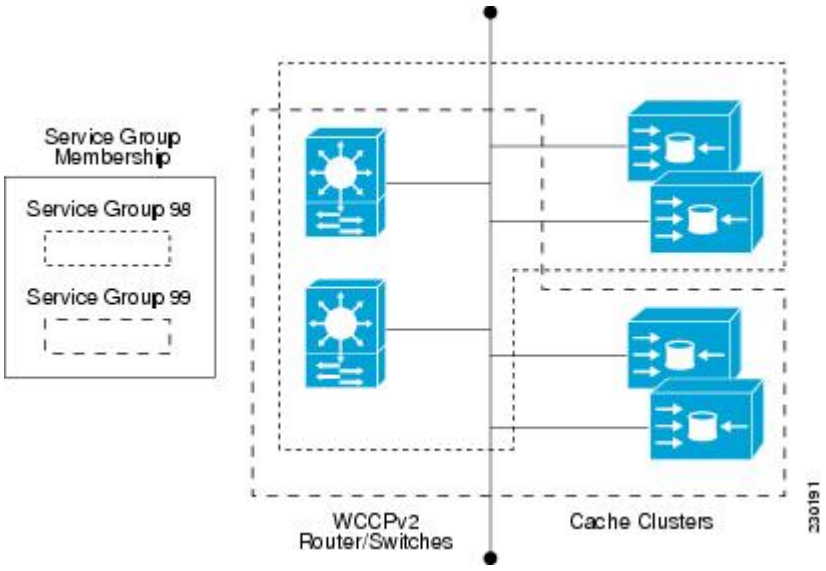
The nature of the selected traffic for redirection is defined by service groups (see figure below) specified on content engines and communicated to routers by using WCCP. The maximum number of service groups allowed across all VRFs is 256.

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 5: 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.

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

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	**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:** `Device(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 4	**interface** *type number* **Example:** `Device(config)# interface ethernet0/0`	Targets an interface number for which the web cache service will run, and enters interface configuration mode.
Step 5	**ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} **redirect** {**in** | **out**} **Example:** `Device(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 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 7	**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.

	Command or Action	Purpose
Step 8	**ip wccp redirect exclude in** **Example:** `Device(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:	Configures a dynamic WCCP service as closed or open.

	Command or Action	Purpose

	• **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`
		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** **Example:** `Device(config)# ip 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 **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.

	Command or Action	Purpose
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 **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.
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**]	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* 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	**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 the WCCP 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. **ip wccp** [**vrf** *vrf-name*] {**web-cache** | *service-number*} [**group-address** *multicast-address*] [**redirect-list** *access-list*] [**group-list** *access-list*] [**password** *password*]
4. **ip 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	**ip 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)# ip wccp web-cache`	Enables the support for a Cisco content engine service group or any content engine service group and configures a redirect ACL list or group ACL. **Note** The **web-cache** keyword is for WCCP version 1 and version 2 and the *service-number* argument is for WCCP version 2 only.
Step 4	**ip wccp check acl outbound** **Example:** `Device(config)# ip 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 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).

	Command or Action	Purpose
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.
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).

	Command or Action	Purpose
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:** `Device> 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:** `Device# 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.

	Command or Action	Purpose
Step 3	**show ip interface** **Example:** `Device# 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:** `Device# 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

```
Device# configure terminal
Device(config)# ip wccp web-cache group-address 224.1.1.100 password password1
```

```
Device(config)# ip wccp source-interface GigabitEthernet 0/1/0
Device(config)# ip wccp check services all
 Configures a check of all WCCP services.
Device(config)# interface GigabitEthernet 0/1/0
Device(config-if)# ip wccp web-cache redirect in
Device(config-if)# exit
Device(config)# interface GigabitEthernet 0/2/0
Device(config-if)# ip wccp redirect exclude in
Device(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: WCCP Outbound ACL Check Configuration

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)# ip wccp web-cache
Device(config)# ip 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)# ip 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: 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*.

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
```

Additional References

Related Documents

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

Related Topic	Document Title
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

Table 3: Feature Information for WCCP

Feature Name	Releases	Feature Information
WCCP Bypass Counters	12.3(7)T 12.2(25)S	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.

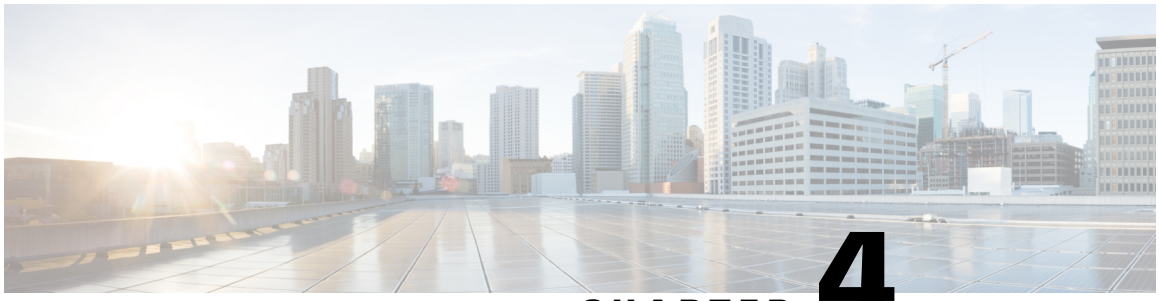
Feature Name	Releases	Feature Information
WCCP Closed Services	12.4(11)T	The WCCP Closed Services feature permits WCCP services to be configured so that WCCP always intercepts traffic for such services but, if no WCCP client (such as a content engine) has registered to receive this traffic, packets are 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	12.3(14)T 12.2(33)SRA 12.2(33)SXH	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**, **ip wccp outbound-acl-check**, **show ip wccp**.

Feature Name	Releases	Feature Information
WCCP Layer 2 Redirection/Forwarding	12.4(20)T	The WCCP Layer 2 Redirection/Forwarding feature allows directly connected Cisco content engines to use Layer 2 redirection, which is more efficient than Layer 3 redirection via GRE encapsulation. You can configure a directly connected Cache Engine to negotiate use of the WCCP Layer 2 Redirection/Forwarding feature. The WCCP Layer 2 Redirection/Forwarding feature requires no configuration on the router or switch. There are no new or modified commands associated with this feature.
WCCP L2 Return	12.4(20)T	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	12.4(20)T	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.

Feature Name	Releases	Feature Information
WCCP Outbound ACL Check	12.3(7)T 12.2(25)S	The WCCP Outbound ACL Check feature enables you to ensure that traffic redirected by WCCP at an input interface is subjected to the outbound ACL checks that may be configured on the output interface prior to redirection. This feature is supported by Web Cache Communication Protocol (WCCP) Version 1 and Version 2. The following commands were introduced or modified by this feature: **ip wccp**, **ip wccp check acl outbound**.
WCCP Redirection on Inbound Interfaces	12.1(3)T 15.0(1)S	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	12.0(3)T 15.0(1)S	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, ip wccp redirect, ip wccp redirect exclude in, ip wccp version, show ip wccp**.

Feature Name	Releases	Feature Information
WCCP VRF Support	15.0(1)M 12.2(33)SRE	The WCCP VRF Support feature provides enhancements to the existing WCCPv2 protocol which support VRF awareness. In Cisco IOS Release 12.2(33)SRE, this feature is supported only on Cisco 7200 NPE-G2 and Cisco 7304-NPE-G100 routers. 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 4

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

- 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.

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 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.

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 supports 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.

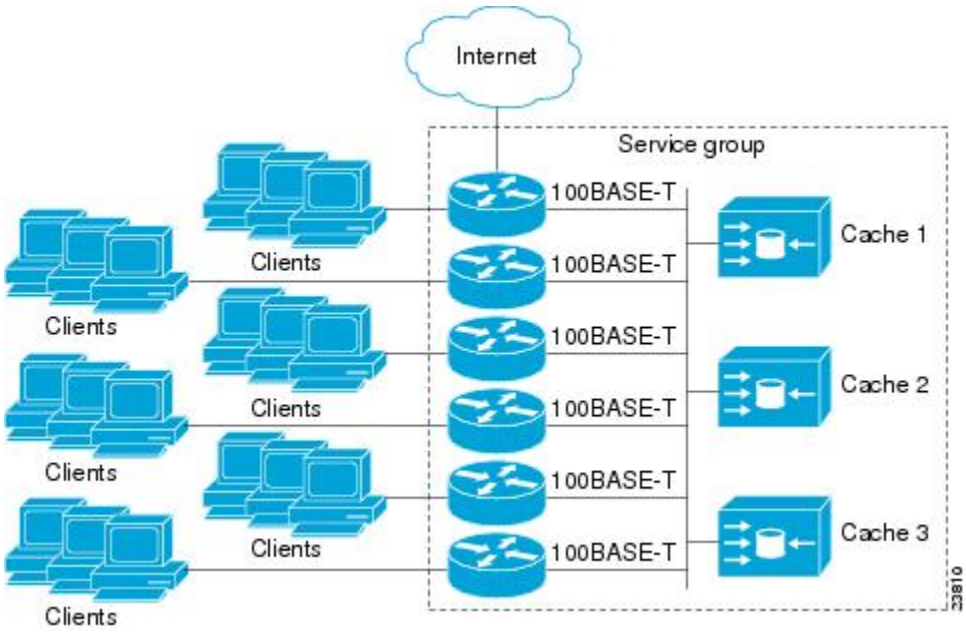
For information about Cisco ACNS commands used to configure Cisco Content Engines, see the Cisco ACNS Software Command Reference.

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 6: 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. WCCPv1 supported the redirection of HTTP (TCP port 80) traffic only. 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 VRF Tunnel Interface

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 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
  0000000000000000120010DB800010000
  0000000000000000110000883E00000000
  Provider: TUNNEL
IPV6     Tunnel1                   2001:DB8:1::12(2)
  Encap length 48
  6000000000002FFF20010DB801000000
  0000000000000000120010DB800010000
  0000000000000000120000883E00000000
  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
                                   0000000000000000120010DB800010000
                                   0000000000000000110000883E00000000
                                   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
                                   0000000000000000120010DB800010000
                                   0000000000000000110000883E00000000
                                   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 only for closed-mode services. Use the **service-list** keyword and *service-access-list* argument to register an application protocol type or port number.

When there is a mismatch between the service list ACL and the definition received from a cache engine, the service is not allowed to start.

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.

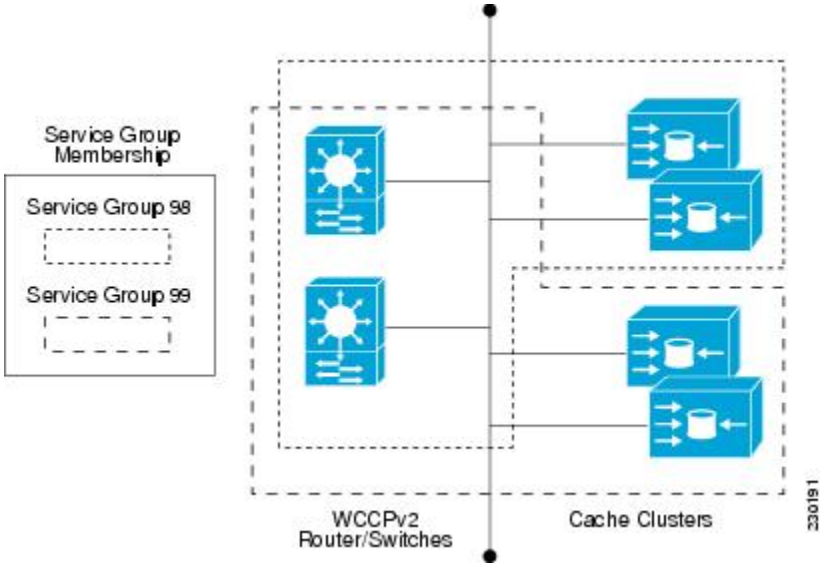
The nature of the selected traffic for redirection is defined by service groups (see figure below) specified on content engines and communicated to routers by using WCCP. The maximum number of service groups allowed across all VRFs is 256.

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 7: 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—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**	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	**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.

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.
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—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.

Example: WCCPv2—IPv6—Cisco ASR 1000 Platform Specific Configuration

The following example shows how to display platform-specific configuration and IPv6 counters information related to WCCP on a Cisco ASR 1000 Series Aggregation Services Router:

```
Device# show platform software wccp service-number ipv6 counters
Service Group (1, 61, 0) counters
        Unassigned count = 0
        Dropped due to closed service count = 0
        Bypass count = 0
        Bypass failed count = 0
        Denied count = 0
        Redirect count = 4
  CE = 2001:1:100::105, obj_id = 213, Redirect Packets = 4
```

The following example shows how to display platform-specific configuration and route processor slot information related to WCCP on a Cisco ASR 1000 Series Aggregation Services Router:

```
Device# show platform software wccp rp active service-number ipv6
IPV6 Dynamic service 61
Priority: 34, Number of clients: 1
Assign Method: Hash, Fwd Method: GRE, Ret Method: GRE
L4 proto: 6, Use Source Port: No
Is closed: No
```

The following example shows how to display platform-specific configuration and embedded service processor slot information related to WCCP on a Cisco ASR 1000 Series Aggregation Services Router:

```
Device# show platform software wccp fp active service-number ipv6
IPV6 Dynamic service 61
Priority: 34, Number of clients: 1
Assign Method: Hash, Fwd Method: GRE, Ret Method: GRE
Is closed: No
Current ACE: 0, Pending ACE: 0
New ACE: 0, New ACE completed: No
ACL id: 0
  AOM id: 0x18a, status: created
```

The following example shows how to display the WCCP service group information in the active Cisco Quantum Flow Processor (QFP) on a Cisco ASR 1000 Series Aggregation Services Router:

```
Device# show platform hardware qfp active feature wccp service id service-id ipv6
Service ID: 61
Service Type: 1
Service Priority: 34
Assign Method: 1
Hash key: 0x51
Hash buckets ppe address: 0x8bceb600
Mode: Open
State: Active
Number of Caches in this service: 1
  ce index: 0
  cache_id : 11
  Cache ip addr : 0x20010001
  Cache cfg ppe addr : 0x8bcab200
  Cache oce ppe addr : 0x891a7670
  Cache state ppe addr : 0x8bcfd288
Number of interfaces using this service: 1
  Interface: GigabitEthernet0/0/0.1
  cpp-if-h: 12
  Dir: 0
  pal-if-h: 15
  uidb sb ppe addr: 0x8bd308e0
```

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 4: Feature Information for WCCPv2 —IPv6 Support

Feature Name	Releases	Feature Information
WCCPv2—IPv6 Support	15.2(3)T 15.1(1)SY1 Cisco IOS XE Release 3.10	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** .