CHAPTER 3

Configuring the Content Services Gateway

This chapter describes how to configure the CSG and contains these sections:

Preparing to Configure the CSG

Before you configure the CSG, take the following actions:

- Make sure that the Cisco IOS version for the switch matches that of the module. You must use Cisco IOS Release 12.1(12c)E or later.
- Configure VLANs on the Catalyst 6000 series switch or Cisco 7600 series router *before* you configure VLANs for the CSG. VLAN IDs must be the same for the switch and the module. Refer to the *Catalyst 6000 Series IOS Software Configuration Guide* or the *Cisco 7600 Series Cisco IOS Software Configuration Guide* for details.

  The following example shows how to configure VLANs:

```
Router> enable
Router# vlan database
Router(vlan)# vlan 130
VLAN 130 added:
    Name: VLAN130
Router(vlan)# vlan 150
VLAN 150 added:
    Name: VLAN150
Router(vlan)# exit
```

- Place physical interfaces that connect to the servers and to the clients in the corresponding VLAN.

  The following example shows how to configure a physical interface as a Layer 2 interface and assign it to a VLAN:

```
Router> enable
Router# config
Router(config)# interface 3/1
Router(config-if)# switchport
Router(config-if)# switchport access vlan 150
Router(config-if)# no shutdown
Router(vlan)# exit
```

- If the Multilayer Switch Function Card (MSFC) is used on the next-hop router on either the client-side or the server-side VLAN, then you must configure the corresponding Layer 3 VLAN interface.

Caution If you use the MSFC as the router for both the client and the server side at the same time, you must ensure that packets for billable flows cannot bypass the CSG. Also, if you use static **ip route** statements to switch traffic to the CSGs, packets might loop between the MSFC and the CSG in this configuration. To avoid these problems, use other routing techniques to switch packets to the CSG, such as policy-based routing.

The following example shows how to configure the Layer 3 VLAN interface:

```
Router> enable
Router# config
Router(config)# interface vlan 130
Router(config-if)# ip address 10.10.1.10 255.255.255.0
Router(config-if)# no shutdown
Router(vlan)# exit
```

Using the CLI

The software interface for the CSG is the Cisco IOS command-line interface (CLI). For more information about using the CLI and Cisco IOS command modes, see Chapter 2 in the *Catalyst 6000 Series IOS Software Configuration Guide*, and Chapter 2 in the *Cisco 7600 Series Cisco IOS Software Configuration Guide.*

Accessing Online Help

In any command mode, you can get a list of available commands by entering a question mark (?) as follows:

```
Router> ?
```

or

```
Router(config)# ip csg ?
```

Online help shows the default configuration values and ranges available to commands.

Upgrading to a New CSG Release

This section describes three methods for upgrading the CSG:

During the upgrade, enter all commands on a console connected to the supervisor engine. Enter each configuration command on a separate line.

Note To complete the upgrade, enter the **exit** command to return to the supervisor engine prompt. If you do not terminate the session, and you remove the CSG from the Catalyst 6000 series chassis, you cannot enter configuration commands to the CSG unless you press **Ctrl** + **^**, enter **x**, and enter the **disconnect** command at the prompt.

The CSG can run in hybrid mode, with CatOS on the Supervisor Engine and Cisco IOS on the MSFC. In the CSG/Hybrid, you can only upgrade the CSG from the MSFC. To enter the MSFC console from CatOS, enter **switch console**. After you enter the MSFC console, you can configure the CSG the same as in native mode. To exit from the MSFC console, enter **^C** three times.

In a redundant MSFC configuration, you cannot upgrade older versions of the CSG from the MSFC in slot 2 with the keyword **slot0:**. To work around this problem, you can either upgrade from the MSFC in slot 1, or you can upgrade with IP address 127.0.0.22.

Upgrading from the Supervisor Engine Bootflash

For instructions on loading images into bootflash, see the *Catalyst 6000 Family Flash Card Install Note* or to the *Cisco 7600 Series Cisco IOS Software Configuration Guide*.

To upgrade the CSG from the supervisor engine bootflash, perform these steps:

Step 1 Enable the TFTP server to supply the image from bootflash:

```
Router> enable
Router# configure terminal
Router(config)# tftp-server bootflash:name
```

where *name* is the CSG image name, such as c6csg-apc.31-3.c5.5.

Step 2 Set up a session between the supervisor engine and the CSG:

```
Router# session slot slot-number processor 0
```

where *slot-number* is the slot number for the CSG to be upgraded.

Step 3 Load the image from the supervisor engine to the CSG:

```
CSM# upgrade 127.0.0.zz name
```

where:

- zz is **12** if the supervisor engine is installed in slot 1 or **22** if installed in slot 2.
- *name* is the CSG image name, such as c6csg-apc.31-3.c5.5.

Step 4 Reboot the CSG by turning it off, then back on, or by entering the following command on the supervisor engine console:

```
Router# hw-module module slot-number reset
```

where *slot-number* is the slot number for the CSG that has been upgraded.

Upgrading from a Flash PC Card

To upgrade the CSG from a removable Flash PC card inserted in the supervisor engine, perform these steps:

Step 1 Enable the TFTP server to supply the image from the removable Flash PC card:

```
Router> enable
Router# configure terminal
Router(config)# tftp-server slotx:name
```

where *name* is the CSG image name, such as c6csg-apc.31-3.c5.5.

Step 2 Set up a session between the supervisor engine and the CSG:

```
Router# session slot slot-number processor 0
```

where *slot-number* is the slot number for the CSG to be upgraded.

Step 3 Load the image from the supervisor engine to the CSG:

```
CSM# upgrade 127.0.0.zz name
```

where:

- zz is **12** if the supervisor engine is installed in slot 1 or **22** if installed in slot 2.
- *name* is the CSG image name, such as c6csg-apc.31-3.c5.5.

Step 4 Reboot the CSG by turning it off then back on, or by entering the following commands on the supervisor engine console:

```
Router# configure terminal
Router# hw-module module slot-number reset
```

where:

- *slot-number* is the slot number for the CSG that has been upgraded.

Upgrading from an External TFTP Server

To upgrade the CSG from an external TFTP server, perform these steps:

Step 1 Create a VLAN on the supervisor engine for the TFTP CSG runtime image download.

Note You can use an existing VLAN. However, for a reliable download, we recommend that you create a VLAN specifically for the TFTP connection.

Step 2 Configure the interface that is connected to your TFTP server.

Step 3 Add the interface to the VLAN.

Step 4 Enter the CSG **vlan** command. See the *"Configuring VLANs" section on page 3-35* for more information.

Step 5 Add an IP address to the VLAN for the CSG.

Step 6 (Optional) Add a route to the TFTP server for the CSG, if necessary.

Step 7 Enter the **show csg** *slot* **vlan detail** command to verify your configuration. See the *"Configuring VLANs" section on page 3-35* for more information.

Step 8 Make a Telnet connection into the CSG with the **session** *slot-number* **0** command.

Step 9 Upgrade the image using the **upgrade** *TFTP-server-IP-address* **c6csg-apc.***revision***.bin** command, where *revision* is **31-3.c5.5** if you are using the CSG 3.1(3)C5(5).

Step 10 Reboot the CSG.

For the CSG/Hybrid, you must enable the VLAN for the CSG from the CatOS console. To do so, enter the following command:

set vlan *vlan-list*

To add a VLAN:

set trunk *slot*/**1** *vlan-list*

To reset a VLAN:

clear trunk *slot*/**1** *vlan-list*

Upgrading to the CSG 3.1(3)C5(5)

The CSG 3.1(3)C5(5) requires one of the following supervisor engines running Cisco IOS Release 12.2(18)SXD:

- Supervisor Engine 2 with an MSFC2 (SUP2-MSFC2)
- Supervisor Engine 720 with an MSFC3-BXL (SUP720-MSFC3-BXL)

The CSG 3.1(3)C5(5) does not support Cisco IOS Releases 12.2(17d)SXB and 12.2(17d)SXB1. Therefore, you must upgrade to a supervisor engine running Cisco IOS Release 12.2(18)SXD, either before you upgrade the CSG or at the same time.

Even if you keep your existing configuration and you do not enable any new CSG 3.1(3)C5(5) features, you must be aware of the following differences between the CSG 3.1(3)C5(5) and the CSG 3.1(3)C5(3):

- The CSG now supports full HTTP pipelining and chunked transfer encoding. This support required extensive redesign of the TCP engine and of HTTP parsing, which in turn impacted the way the CSG counts bytes.

  For HTTP billing, the CSG now reports only TCP byte counts. To maintain backward compatibility, the CSG still reports IP byte counts, but the values reported are the same as the TCP byte counts. Packet counts for pipelined HTTP operations are a snapshot of the number of packets detected on the connection since the previous statistics were reported. The packet count might even be zero if two pipelined operations share the same packet.

  For HTTP, IMAP, POP3, SMTP, and TCP billing, the CSG no longer counts the SYN or FIN for the **basis byte tcp** command in CSG service configuration mode. The CSG now limits the TCP byte count to only the TCP payload.

  For HTTP billing, the CSG no longer supports the **basis bytes ip** command in CSG service configuration mode. If your configuration includes this command, the Cisco IOS generates a warning message.

  The CSG no longer supports pipeline tolerance:

  - The CSG ignores pipeline tolerance in existing configurations.
  - The CGS no longer inserts a FIN.
  - The CSG no longer forces HTTP 1.0 behavior to the server and browser.
- All new CSG 3.1(3)C5(5) TLVs are optional. Except as indicated below, they cause no backward compatibility issues with entities that support previous releases of the interface, provided those entities ignore unrecognized TLVs and messages. The following incompatibility issues are the only known issues at this time:
  - For SMTP and POP3 billing, the CSG sends only the SMTP or POP3 CDR. The CSG no longer sends a TCP CDR.
  - For HTTP billing, the CSG no longer sends HTTP CDRs and associated TCP CDRs sequentially. For persistent connections, the CSG might send two or more HTTP CDRs in a row before sending the associated TCP CDRs.

Performing a Hitless Upgrade

A hitless upgrade allows you to upgrade to a new version without any major service disruption due to the downtime for the upgrade. To perform a hitless upgrade, perform these steps:

Step 1 Perform a write memory on standby.

Step 2 Upgrade the standby system with the new release, and then reboot the CSG. The standby CSG boots as standby with the new release.

Step 3 After rebooting, wait for all of the information to propagate to the standby. Be aware that it might take up to an hour for this process to complete.

Step 4 Upgrade the active CSG with the new release, and then reboot the active CSG. When the active CSG reboots, the standby CSG becomes the new active CSG and takes over the service responsibility.

The rebooted CSG comes up as standby.

Saving and Restoring Configurations

For information about saving and restoring configurations, see the *Catalyst 6000 Series IOS Software Configuration Guide* or to the *Cisco 7600 Series Cisco IOS Software Configuration Guide*.

Configuring the CSG

This section identifies the tasks you must perform before you can use the content billing feature on the CSG. It provides information on the following topics:

Other Configuration Tasks

This section provides information on the following topics

Specifying CSG Locations

Before you can enter CSG configuration commands on the switch, you must specify the CSG that you want to configure.

To specify the slot number of a CSG in module CSG configuration mode, perform this task:

	Command	Purpose
Step 1	`Router# config t`	Enters configuration mode.
Step 2	`Router(config)# module csg` *slot-number*	Enters module CSG configuration mode for a specified slot.

The **module csg** command places you in module CSG configuration mode. All further configuration commands that you enter apply to the CSG installed in the slot you have specified.

Note Unless otherwise specified, all the examples in this publication assume that you have already entered this command and entered the configuration mode for the CSG you are configuring.

Configuring User Groups

To configure the CSG to record and generate accounting records, you must specify the user groups you want to generate accounting records for, as well as the user database that the CSG queries for user IDs.

To configure user groups on the CSG; to specify the user database, RADIUS endpoint, and quota servers; and to configure redirect NAT, perform the following steps:

	Command	Purpose
Step 1	`Router(config)# ip csg user-group group-name`	Defines a CSG user group and specifies a user database name.
Step 2	`Router(config-csg-group)# database ip-address port-number`	Specifies the location of the user database, including the IP address and port number of the user database.
Step 3	`Router(config-csg-group)# entries max entries-number`	(Optional) Defines the maximum number of entries in the CSG User Table.
Step 4	`Router(config-csg-group)# quota local-port port-number`	(Optional) Configures the local port on which the CSG receives communications from quota servers.
Step 5	`Router(config-csg-group)# quota server ip-address port-number priority`	(Optional) Configures the quota servers that return billing quota values for users. **Note** The CSG does not support multiple quota servers with the same IP address.
Step 6	`Router(config-csg-group)# quota activate number`	(Optional) Allows load balancing of quota servers, similar to the BMA load balancing feature. Multiple quota servers can be simultaneously active, and the CSG assigns a quota server to each user. All quota transactions for the user are done with the same quota server. When a quota server fails, all users associated with that quota server are distributed among other quota servers. The valid range for the *number* argument is 1 through 10. **Note** Multiple quota servers cannot have the same IP address.
Step 7	`Router(config-csg-group)# radius acct-port port-number`	Specifies the port number for the RADIUS accounting endpoint. You can still use the **radius key** and **radius acct-port** commands in CSG user group configuration mode to configure the CSG as a RADIUS Accounting endpoint, but we recommend that you use the **radius endpoint** command in module CSG configuration mode. The CSG 3.1(3)C5(5) supports both endpoint configuration methods. However, if you plan to use RADIUS PoD with RADIUS endpoint, then you must use the **radius endpoint** command in module CSG configuration mode. We do not recommend using both configuration methods in the same environment.
Step 8	`Router(config-csg-group)# radius handoff [duration]`	(Optional) Configures RADIUS handoff support.
Step 9	`Router(config-csg-group)# radius key secret`	Configures the CSG to be the RADIUS endpoint for accounting records, and provides the key. You can still use the **radius key** and **radius acct-port** commands in CSG user group configuration mode to configure the CSG as a RADIUS Accounting endpoint, but we recommend that you use the **radius endpoint** command in module CSG configuration mode. The CSG 3.1(3)C5(5) supports both endpoint configuration methods. However, if you plan to use RADIUS PoD with RADIUS endpoint, then you must use the **radius endpoint** command in module CSG configuration mode. We do not recommend using both configuration methods in the same environment.

	Command	Purpose
Step 10	`Router(config-csg-group)# radius parse strict`	(Optional) Tightens the parsing rules for RADIUS flows.
Step 11	`Router(config-csg-group)# radius pod attribute` *radius_attribute_number*	(Optional) Specifies the RADIUS attributes to be copied from the RADIUS Start message and sent to the NAS in the Packet of Disconnect (PoD).
Step 12	`Router(config-csg-group)# radius pod nas` [*start-ip end-ip*] *port* [**key** [*encrypt*] *secret-string*]	(Optional) Specifies the NAS port to which the CSG should send the Packet of Disconnect (PoD) message, and the key to use in calculating the Authenticator.
Step 13	`Router(config-csg-group)# radius pod timeout` *timeout* **retransmit** *retransmit*	(Optional) Specifies the number of times to retry the RADIUS Packet of Disconnect (PoD) message if it is not ACKed, and the interval between retransmissions.
Step 14	`Router(config-csg-group)# radius server` *ip-address* [*port-number*]	(Optional) Enables RADIUS proxy.
Step 15	`Router(config-csg-group)# radius userid` {**1** | **31** | **User-Name** | **Calling-Station-Id**}	(Optional) RADIUS attribute used to extract the user IDs from a RADIUS record.
Step 16	`Router(config-csg-group)# radius start restart session-id` {*attr_number* | {**26** | **vsa**} {*vendor_id* | **3gpp**} *sub-attr_number*}	(Optional) Deletes an existing User Table entry for a specific user (when a RADIUS Accounting Start is received), and creates a new entry for that user (similar to when a RADIUS Accounting Stop has been received).
Step 17	`Router(config-csg-group)# radius stop purge` {*attr_number* | {**26** | **vsa**} {*vendor_id* | **3gpp**} *sub-attr_number*}	(Optional) Specifies the attribute (which may be a vendor-specific attribute) that must be included in the RADIUS Accounting Stop request in order for the User Table entry to be deleted.
Step 18	`Router(config-csg-group)# radius monitor` *server_addr server_port* [**key** [*encrypt*] *secret-string*]	Specifies that the CSG should monitor the RADIUS flows to the specified server.
Step 19	`Router(config-csg-group)# redirect nat` *ip-address* [*port-number*]	(Optional) Redirects client NAT flows to an alternate IP address when the client's quota is exhausted.
Step 20	`Router(config-csg-group)# redirect http` *url*	(Optional) Redirects client HTTP flows to an alternate URL when the client's quota is exhausted.
Step 21	`Router(config-csg-group)# redirect wap` *url*	(Optional) Redirects client WAP flows to an alternate URL when the client's quota is exhausted.
Step 22	`Router(config-csg-group)# aoc confirmation`	Configures a token for use in advice of charge (AoC) URL-rewriting.
Step 23	`Router(config-csg-group)# user-profile server` {**quota** | **radius** {**remove** | **pass**}}	(Optional) Specifies which server is used to obtain the user profile or billing plan. **Note** The VSA is removed from the Access-Accept message only if **remove** is specified. We recommend that you use **pass** to reduce processing time on the CSG. You should use **remove** only if the RADIUS client cannot tolerate the Cisco VSA in the message. Additionally, the user ID must be in the message containing the billing plan.

The following example shows how to configure a CSG user group, including a database, a RADIUS endpoint, quota servers, and redirect NAT:

```
ip csg user-group G1
 entries max 100000
 database 10.1.2.3 11111
 quota local-port 6666
 quota server 10.1.4.5 888 1
 quota server 10.1.6.7 999 2
 radius acct-port 7777
 radius key SECRET_PASSWORD
 radius parse strict
 radius server 10.13.14.15
 radius userid User-Name
 redirect nat 10.33.33.3
!
ip csg user-group U1
 radius userid User-Name
 radius monitor 10.2.3.4 1234 key cisco
 radius monitor 10.2.3.9 1234 key cisco2
 radius monitor 10.2.7.4 3901 key cisco
```

Configuring Accounting Policies

To configure the CSG to record and generate accounting records, you must define content-based client accounting as a service. This includes specifying the user groups you want to generate accounting records for, as well as the Billing Mediation Agent to send accounting records to.

To configure the accounting policies on the CSG, perform the following steps:

	Command	Purpose
Step 1	`Router(config)# ip csg accounting` *name*	Defines content-based client accounting as a policy.
Step 2	`Router(config-csg-accounting)# user-group` *name*	Associates a user group with a specific accounting service.
Step 3	`Router(config-csg-accounting)# agent` *ip-address port-number priority*	Defines the primary and backup Billing Mediation Agents (BMAs) to which billing records are to be sent. **Note** The CSG does not support multiple agents with the same IP address.
Step 4	`Router(config-csg-accounting)# agent activate` [*number* [**sticky** *seconds*]]	(Optional) Enables support for multiple active BMAs
Step 5	`Router(config-csg-accounting)# agent local-port` *port-number*	(Optional) Defines the port on which the CSG is to listen for packets from the BMAs.
Step 6	`Router(config-csg-accounting)# keepalive` *number-of-seconds*	(Optional) Defines the keepalive time interval (in seconds) used to test the health of BMAs.
Step 7	`Router(config-csg-accounting)# records batch`	(Optional) Batches billing records into a single message before sending them to the BMA.
Step 8	`Router(config-csg-accounting)# records http-statistics`	(Optional) Sends the HTTP Statistics data record to the BMA.
Step 9	`Router(config-csg-accounting)# records intermediate` {**bytes** *bytes* | **time** *seconds* | **bytes** *bytes* **time** *seconds*}	(Optional) Enables the generation of intermediate billing records.

	Command	Purpose
Step 10	`Router(config-csg-accounting)# records max`	(Optional) Defines the maximum number of billing records that can be stored or queued in the CSG before they are forwarded to the Billing Mediation Agent (BMA).
Step 11	`Router(config-csg-accounting)# records format`	(Optional) Specifies variable, fixed, or variable-single CDR format.
Step 12	`Router(config-csg-accounting)# record-storage` *ip-address* [*port*]	(Optional) Defines a PSD to associate with this accounting group.
Step 13	`Router(config-csg-accounting)# record-storage local-port` *port*	(Optional) Defines the source port to be used by the CSG when communicating with the record store.
Step 14	`Router(config-csg-accounting)# report http header` *header_name*	(Optional) Defines the inclusion of multiple HTTP request headers in the CSG HTTP_Header CDR.
Step 15	`Router(config-csg-accounting)# report radius attribute` *radius_attribute_number*	(Optional) Specifies the RADIUS attributes to be copied from the RADIUS Start message and sent to the BMA in each billing record.
Step 16	`Router(config-csg-accounting)# inservice`	Activates the accounting service on a CSG.

Step 17	`Router# show module csg` *slot* `accounting {agent | database | error | quota-server | radius | users {all | statistics | ip-address` [*ipmask*] `| userid` *userid*`}} [detail] [module` *num*`]`
	
	or
	
	`Router# show ip csg accounting {agent | database | error | quota-server | radius | users {all | statistics | ip-address` [*ipmask*] `| userid` *userid*`}} [detail] [module` *num*`]`
		Displays information for the CSG billing feature.

The following example shows how to define the CSG accounting policy:

```
ip csg accounting A1
 user-group G1
 agent activate 2
 agent local-port 3775
 agent 10.1.2.4 11112 1
 agent 10.1.2.5 11113 2
 keepalive 3
 records batch
 records http-statistics
 records intermediate bytes 100000 time 3600
 records max 250
 record-storage local-port 5002
 record-storage 172.18.12.226
 report http header x-subno
 report http header x-al-session-id
 report radius attribute 3
 report radius attribute 5
 inservice
```

Activating the Accounting Policy on the CSG

To activate the accounting policy on the CSG, enter the following command in module CSG configuration mode:

Command	Purpose
`Router(config-csg-module)# accounting service-name`	Downloads a configured accounting service to a CSG card.

Defining Client/Server Connectivity

To properly configure the CSG, you must create VLANs for both the client side and server side of the switch. You must do this so that the CSG knows where to forward the traffic it receives. The minimal configuration requires one client-side VLAN and one server-side VLAN. Additionally, you must configure IP addresses for the VLANs, and all gateway IP addresses.

To configure server-side VLANs on the CSG, enter the following command in module CSG configuration mode:

Command	Purpose
`Router(config-csg-module)# vlan vlan-id server [vlan-name]`	Configures the server-side VLANs and enters the server VLAN mode. **Note** You cannot use VLAN 1 as a server-side VLAN for the CSG.

Then configure an IP address on this VLAN.

To configure client-side VLANs on the CSG, enter the following commands, beginning in module CSG configuration mode:

Command	Purpose
`Router(config-csg-module)# vlan vlan-id client [vlan-name]`	Configures the client-side VLANs and enters the client VLAN mode. **Note** You cannot use VLAN 1 as a client-side VLAN for the CSG.

Then configure an IP address on this VLAN.

The following example shows how to configure client and server VLANs:

```
vlan 10 server
 ip address 10.250.0.1 255.255.0.0
 gateway 10.250.1.1

vlan 251 client
 ip address 10.251.0.1 255.255.0.0
 route 10.200.0.0 255.254.0.0 gateway 10.251.2.11
```

Downloading an Accounting Service

Before you can configure the CSG to perform content billing, you must enable it to reference and download a specific accounting service configuration.

To install the accounting service in a specific CSG, enter the following command in module CSG configuration mode:

Command	Purpose
`Router(config-csg-module)# accounting service-name`	Assigns a specific accounting service to a specific CSG.

Downloading Ruleset Content

A CSG billing ruleset is a list of all content names that are to be downloaded to a specific CSG card.

To download all content defined by a ruleset to a CSG card, enter the following command in module CSG configuration mode:

Command	Purpose
`Router(config-csg-module)# ruleset ruleset-name`	Downloads all content defined by a ruleset to a CSG card.

Configuring Policies and Traffic Types

Policies are access rules that traffic must match for a server farm. Policies allow the CSG to apply filters to certain types of traffic subject to the accounting service.

When the CSG is able to match policies, it selects the policy that appears first in the policy list. Policies are located in the policy list in the sequence in which they were configured in the content. You can reorder the policies in the list by removing policies and reentering them in the correct order.

To configure accounting records policies in module CSG configuration mode, perform this task:

	Command	Purpose
Step 1	`Router(config-csg-module)# ip csg policy policy-name`	Defines a policy for qualifying flows for CSG accounting services, and enters CSG policy configuration mode.
Step 2	`Router(config-csg-policy)# accounting [type {http | ftp | wap {connection-oriented | connectionless} | rtsp | ftp | smtp | pop3 | other}] [customer-string string]`	Defines the accounting type and customer string for all flows that comply with a CSG billing policy.
Step 3	`Router(config-csg-policy)# client-group {std-access-list-number | std-access-list-name}`	References a standard access list that is part of a CSG billing policy.
Step 4	`Router(config-cag-policy)# client-ip http-header x-forwarded-for`	Specifies that the user's IP address is to be obtained from the URL header after the **x-forwarded-for** keyword.
Step 5	`Router(config-cag-policy)# header-map header-map-name`	References a header map that is part of a CSG billing policy.
Step 6	`Router(config-cag-policy)# next-hop ip-address`	Defines a next-hop IP address.
Step 7	`Router(config-cag-policy)# url-map url-map-name`	References a URL map that is part of a CSG billing policy.

The following example shows how to define a policy:

```
ip csg policy MOVIES_COMEDY
 accounting type http customer-string MOVIES_COMEDY
 client-group 44
 client-ip http-header x-forwarded-for
```

```
header-map MOVIES
next-hop 33.0.0.150
url-map MOVIES
```

Configuring a Content Billing Service

A CSG content billing service is a component of a billing plan that is subscribed to by users.

You can configure one or more content billing services for the CSG. Each service represents a group of content that is billed the same way, such as billing per-click (or per-request) or billing per-IP byte, and that shares part of a user's quota. Grouping content into one or more services enables you to separate, for example, a user's prepaid quota for Internet browsing from his quota for e-mails.

For each service, the CSG downloads a separate quota, and deducts from that quota. Quotas are specified in units called *quadrans*. A quadran is a generic unit whose exact "value" is defined by each quota server. A quadran can represent, for example, a click for a per-click service (for example, an HTTP request), and a byte for a per-volume service. The value of a quadran is transparent to the CSG; it simply requests and downloads quadrans as needed from quota servers.

The CSG requests an additional quota grant when a user's per-click quota falls below a specified percentage of the last quota grant, or when a user's per-volume quota falls below a specified percentage of the last quota grant or 32 KB, whichever is greater.

For each service that a user tries to access, the CSG maintains a separate logical accounting session. When a user's quota is divided among multiple services, the CSG requests an additional quota grant for each service individually, based on its usage.

If a user fails authorization for a service, but continues to send new requests for that service, the CSG waits a specified time before sending a reauthorization request for that user to the quota server. This ensures that the quota server is not inundated with reauthorization requests from unauthorized users.

The billing basis specifies how billing is to be charged:

- Per-click (fixed-cost) billing is charged at a fixed cost, which is deducted for each content instance accessed (that is, deducted for each request).
- Volume-based billing can be based on either the number of IP bytes or the number of TCP bytes.
- Duration-based billing can be based on either service duration time or connection duration time.
- The **exclude mms** option specifies that MMS content over WAP is not billed.

To configure a content billing service, perform this task:

	Command	Purpose
Step 1	`Router(config-csg-module)# ip csg service` *service-name*	Defines a content billing service, and enters CSG service configuration mode.
Step 2	`Router(config-csg-service)# content` *content-name* `policy` *policy-name* [`weight` *weight-name*]	Defines content as a member of a CSG billing service, identifies a policy to apply to this content, and optionally assigns a weight to this content.
Step 3	`Router(config-csg-service)# basis {byte {ip | tcp} | {fixed | second [connect] [exclude mms]}`	(Optional) Specifies the billing basis for a CSG content billing service. **Note** When changing the basis for a service, the content must be taken out of service.
Step 4	`Router(config-csg-service)# idle` *duration*	(Optional) Specifies the minimum amount of time that the CSG maintains a service with no user sessions.

The CSG allows you to define a pool of up to 255 services. You can authorize each user for any number of services from that pool, but we recommend that the billing system not authorize each user for more than 10 active services. Exceeding this guideline could lead to the following problems:

- The increase in the number of quota authorizations per user can overload the quota server, as well as the CSG.
- As the number of services for which a user is actively authorized increases, the user's quota becomes fragmented. Although the CSG allows the billing system to recall and redistribute the quota, so that the user is not denied service due to quota fragmentation, the process increases overhead in both the quota server and the CSG.

The following example shows how to define a content billing service:

```
ip csg service MOVIES
 basis fixed
 content MOVIES_COMEDY policy MOVIES_COMEDY
 content MOVIES_ACTION policy MOVIES_ACTION weight DOUBLE
 idle 120
```

Configuring Content

The CSG uses the Cisco command-line interface (CLI), and requires content definitions or virtual server definitions. This section provides information about configuring content.

A CSG content specification contains the following information:

- Layer 3 information that specifies the IP-level details of the content.
- Layer 4 information that specifies transport layer parameters, such as TCP and UDP port numbers.

If the content specification does not match any service listed under a user's billing plan, the CSG considers the service to be either free or postpaid. The CSG does not try to authorize the user with the quota server for the service.

To specify content for a CSG accounting service, perform the following tasks, beginning in module CSG configuration mode:

	Command	Purpose
Step 1	`Router(config-csg-module)# ip csg content` *content-name*	Defines content for CSG accounting services, and enters CSG content configuration mode.
Step 2	`Router(config-csg-content)# policy` *policy-name*	References a CSG billing policy.
Step 3	`Router(config-csg-content)# ip {any | ip-address [netmask]} [protocol [port-number last port-number]]`	Defines the Layer 3/Layer 4 subset of flows that can be processed by the CSG accounting services. You can define *port-number* as a single value or a range of numbers.
Step 4	`Router(config-csg-content)# client [include | exclude] {any | ip-address [netmask]`	(Optional) Defines the client IP address spaces that can use the CSG content server.
Step 5	`Router(config-csg-content)# idle` *duration*	(Optional) Specifies the minimum amount of time the CSG maintains an idle content connection.
Step 6	`Router(config-csg-content)# pending` *timeout*	(Optional) Sets the pending connection timeout.
Step 7	`Router(config-csg-content)# replicate connection tcp`	(Optional) Replicates the connection state for all TCP connections to the CSG content servers on the backup system.

	Command	Purpose
Step 8	`Router(config-csg-content)# vlan vlan-name`	(Optional) Restricts CSG billing content to a single source VLAN.
Step 9	`Router(config-csg-content)# inservice`	Activates the content service on each CSG.
Step 10	`Router # show module csg slot content [name content-name] [detail]`	Displays statistics and counters for CSG content.

The following example shows how to define content for a CSG accounting service:

```
ip csg content MOVIES_COMEDY
 policy POLICY1
 client 10.4.4.0 255.255.255.0
 idle 120
 ip 172.18.45.0/24 tcp 8080
 pending 300
 replicate connection tcp
 vlan MOVIES_COMEDY
 inservice
```

The following example shows how to define a range of port numbers:

```
ip csg content MULTI_PORT
 policy WAP_SRV_POLICY
 ip any udp 30000 30150
 inservice
```

Configuring Fixed or Variable Format CDR Support

The CSG supports both variable and fixed format CDR generation, including a fixed variable format for WAP CDRs. The same set of variables are reported in each CDR regardless of WSP PDU type. CDRs contain zero-length variables when there is no information to report, but the same set of variables are always reported in the same sequence. To configure a specific format, perform the following tasks:

	Command	Purpose
Step 1	`Router(config)# ip csg accounting records format [variable | fixed | variable-single-cdr]`	Specifies variable, fixed, or variable single CDR format.
Step 2	`Router(config)# module csg 3 hostname MYHOST`	Specifies a variable hostname for a CSG module
Step 3	`Router(config)# ip csg billing FOO mode postpaid service X service Y`	Specifies that a billing plan is postpaid or prepaid.
Step 4	`Router(config)# ip csg service FOO owner name ABC_CORP owner id ABC123456`	Specifies the owner responsible for the content associated with a service. The administrator who configures owner identification is responsible for its accuracy. Correct configuration requires that contents for this service, their policies and any associated URL or header maps, identify all data transfers with this owner, and only data transfers with this owner.

	Command	Purpose
Step 5	`Router(config)# ip csg service FOO class 7`	Specifies a service class value.
Step 6	`Router(config)# ip csg transport-type assign 1.2.3.4 6 assign 2.5.3.1 7 assign 6.6.7.5 0`	Classifies data traffic based on its access path using the NAS-IP reported in RADIUS. Use the **assign** command to associate IP addresses with transport-type values. Transport-type information is reported in fixed record format CDRs.

Configuring a Refund Policy on the CSG

The prepaid error reimbursement feature allows the CSG to automatically refund quota for failed transactions, as defined by the CLI. The CSG checks them in the following order: TCP/WAP flags, ApplicationReturnCode. To configure a refund policy on the CSG, perform the following tasks:

	Command	Purpose
Step 1	`Router(config)# ip csg refund`	Specifies a refund policy that can then be applied to the various services, and enters CSG refund configuration mode.
Step 2	`Router(config)# ip csg refund COMPANY-REFUND retcode http 500 509 retcode wap 0x44 0x50 retcode ftp 454`	Specifies the range of application return codes for which the CSG refunds quota.
Step 3	`Router(config)# ip csg refund COMPANY-REFUND retcode http 500 509 retcode wap 0x44 0x50 retcode ftp 454 flags tcp 43 00 flags tcp 63 01 flags tcp 80 80 flags ip 80 80 flags wap 0 8`	Specifies a mask of interesting TCP, IP, or WAP flag bits and values for which the CSG refunds quota.

The following example shows how to configure a refund policy on the CSG:

```
ip csg refund COMPANY-REFUND
  retcode http 500 509
  retcode wap 0x44 0x50
  retcode ftp 454
  flags tcp FF 14
  flags wap FF 08
```

To enable and specify the refunding policy for a CSG prepaid service, specify the following command in CSG service configuration mode:

	Command	Purpose
Step 1	`Router(config-csg-service)# refund-policy` *policy-name*	Enables and specifies the refunding policy for a CSG prepaid service.

The following example shows how to configure the **refund-policy** command:

```
ip csg service BILLPERCLICK
  basis fixed
  refund-policy COMPANY-REFUND
```

```
  content ADVERTISEMENTS policy ADVERTISEMENTS weight PAYBACK
  content BOOKS policy BOOKSALES
  content BOOKS policy BOOKFREE weight FREE
  content CORPORATE policy CORPORATE weight FREE
!
ip csg service BILLBYVOLUME
  basis byte tcp
  refund-policy COMPANY-REFUND
  content BILLBYVOLUME policy BILLBYVOLUME
!
ip csg service BILLBYIPVOLUME
  basis byte
  refund-policy COMPANY-REFUND
  content INTERNET policy INTERNET
```

Configuring RADIUS Accounting Attribute Reporting

The CSG allows you to configure a list of RADIUS accounting attributes that are to be reported to the BMA and quota server in every CDR. To configure these attributes using their standard numbers, perform the following task:

	Command	Purpose
Step 1	`Router(config)# ip csg accounting` *name*	Defines content-based client accounting as a service, and enters CSG accounting configuration mode.
Step 2	`Router(config-csg-accounting)# report radius attribute 3`	Defines which attributes you want to report.

You can specify as many attributes as you desire. The attributes are copied from the RADIUS accounting message and sent in each billing message to the BMA.

Note The CSG examines only the standard RADIUS attribute number. The CSG is not aware of any special formatting or subclassing for Vendor-Specific Attributes (VSAs). If a VSA is desired, then the CSG reports all VSAs (attribute 26).

If the list of configured attributes changes, only new RADIUS requests are subject to the new attributes. Attributes already saved for a user continue to be reported.

When a RADIUS start request is received, any attributes received from a previous start request are deleted. If there are multiple instances of an attribute, they are all reported. Attributes are reported in the order they exist in the RADIUS message.

The following example shows how to define multiple RADIUS attributes:

```
Router(config)# ip csg accounting a1
Router(config-csg-accounting)# report radius attribute 3
Router(config-csg-accounting)# report radius attribute 5
Router(config-csg-accounting)# report radius attribute 7
Router(config-csg-accounting)# report radius attribute 44
```

Configuring RADIUS Proxy

The RADIUS Proxy feature lets you specify that the CSG should be a proxy for RADIUS messages. To configure the RADIUS Proxy feature, enter the following command in module CSG configuration mode:

Command	Purpose
`Router(config-csg-module)# radius proxy csg_addr server addr [csg_source_addr] [key [encrypt] secret-string]`	Specifies that the CSG should be a proxy for RADIUS messages.

Note If you specify the **user-profile server radius remove** command, you might also need to configure a key.

Configuring RADIUS Endpoint

To configure the CSG as a RADIUS Accounting endpoint, enter the following command in module CSG configuration mode:

Command	Purpose
`Router(config-csg-module)# radius endpoint csg_addr [key [encrypt] secret-string]`	Identifies the CSG as an endpoint for RADIUS Accounting messages.

Configuring HTTP Header Reporting

The CSG allows you to include multiple HTTP request headers in the CSG HTTP_Header CDR. To define HTTP reporting on the CSG, perform the following task:

	Command	Purpose
Step 1	`Router(config)# ip csg accounting name`	Defines content-based client accounting as a service, and enters CSG accounting configuration mode.
Step 2	`Router(config-csg-accounting)# report http header x_header`	Defines the inclusion of multiple HTTP request headers in the CSG HTTP_Header CDR. You can specify any number of headers up to 256, and header names cannot exceed 256 characters.

The following example shows how to enable HTTP header reporting for virtual server VS1:

```
Router(config)# ip csg accounting a1
  report http header x-subno
  report http header x-al-session-id
```

Configuring a Ruleset

A CSG billing ruleset is a list of all content names that are to be downloaded to a specific CSG card.

To define a ruleset for CSG billing, perform this task:

	Command	Purpose
Step 1	`Router(config-csg-module)# ip csg ruleset` *ruleset-name*	Configures a CSG billing ruleset, and enters CSG ruleset configuration mode.
Step 2	`Router(config-csg-ruleset)# content` *content-name*	Adds a content reference to a CSG ruleset.

If you have defined more than one content name using multiple **ip csg content** commands, you can configure more than one **content** command in CSG ruleset configuration mode.The following example shows how to define a CSG billing ruleset:

```
ip csg ruleset R1
 content MOVIES_COMEDY
 content MOVIES_ACTION
```

Configuring Maps for Pattern-Matching

The CSG maps are used to match URLs or headers against a pattern, to determine whether flows are to be processed by the CSG accounting services.

To define the CSG billing content filters (URL maps and header maps), perform this task:

	Command	Purpose
Step 1	`Router(config-csg-module)# ip csg map` *map-name* {**url** | **header**}	Defines the CSG billing content filters (URL maps and header maps), and enters CSG map configuration mode.
Step 2	`Router(config-csg-map-header)# match` [**protocol** *protocol*] **header** *header-name* [**value** *pattern*]	Specifies a header match pattern for a CSG billing map.
Step 3	`Router(config-csg-map-url)# match` [**protocol** *protocol*] [**method** *method*] **url** *pattern*	Specifies a URL match pattern for a CSG billing map.

You can specify more than one **match** command in CSG header map configuration mode to specify multiple header match expressions for a given header map:

- If the header matches *all* of the header match expressions, then the match is TRUE and the flows can be processed by the CSG accounting services (unless there is another map associated with this policy that is FALSE).
- If the header *does not* match *even one* of the header match expressions, then the match is FALSE and the flows are not processed by the CSG accounting services, even if other maps for this policy match TRUE.

- The header match expressions are case-sensitive. For example, if you define the following header match expression:

  match header host1 value *.2.*.44

  but the actual HTTP header keyword is host1, the header *does not* match the header match expression, the match is FALSE, and the flow is not processed by the CSG accounting services.

The following example shows how to specify header match patterns for map MOVIES. In this example, the header match is TRUE *only* for host host1 and IP address 20.2.23.44. Any other combination of host and IP address matches FALSE:

```
ip csg map MOVIES header
 match header host1 value *.2.*.44
 match header host* value 20.*.*.44
 match header host* value *.2.23.*
```

You can specify more than one **match** command in CSG URL map configuration mode to specify multiple URL match expressions for a given URL map:

- If the URL matches *any* of the URL match expressions, then the match is TRUE and the flows can be processed by the CSG accounting services (unless there is another map associated with this policy that is FALSE).
- If the URL *does not* match any of the URL match expressions, then the match is FALSE and the flows are not processed by the CSG accounting services, even if other maps for this policy match TRUE.
- The URL match expressions are case-sensitive. For example, if you define the following URL match expression:

  match protocol http url http://*url-string*

  but a subscriber enters the following URL in a Web browser:

  HTTP://*url-string*

  the URL *does not* match the URL match expression, the match is FALSE, and the flow is not processed by the CSG accounting services.

  Therefore, consider upper- and lowercase combinations carefully when creating URL match expressions.
- When you configure URL match patterns for RTSP streams, keep in mind that you must account for trailing stream IDs in RTSP stream names. For example, URL match pattern ***.mpeg** does not match **rtsp://1.1.1.254:554/movie.mpeg/streamid=0** because the stream name has a trailing **/streamid=0**. To match such RTSP stream names, use a URL match pattern such as ***.mpeg***.
- The CSG can handle up to 1000 single-wildcard URL match patterns (for example, ***movies** or **movies***, but not ***movies***) or up to 11 double-wildcard URL match patterns (for example, ***movies*** or **http://test.*movies.com/*.mpeg**). Double-wildcard URL match patterns are also known as keyword URL match patterns. If you want to use keyword URL match patterns, keep the following considerations in mind in order to optimize the CSG's performance:
  - Minimize the number of URL match patterns that are applied to a given CSG content definition.
  - Minimize the number of keyword URL match patterns that you use. In general, it is better to use multiple single-wildcard URL match patterns instead of individual keyword URL match pattern.

  - Combine multiple keyword URL match patterns into a single pattern using UNIX string-matching special characters. For example, ***.movies_comedy.com/*.mpeg**, ***.movies_action.com/*.mpeg**, and ***.movies_drama.com/*.mpeg** can be combined into the following single pattern:

    ***.movies_(comedy|action|drama).com/*.mpeg**

    And the following patterns:

    ***.movies_comedy.com/*.mpeg**

    ***.movies_action.com/*.mpeg**

    ***.movies_drama.com/*.mpeg**

    ***.clips_comedy.com/*.mpeg**

    ***.clips_action.com/*.mpeg**

    ***.clips_drama.com/*.mpeg**

    can be combined into the following single pattern:

    ***.(movies|clips)*?*(comedy|action|drama).com/*.mpeg**

    Remember that the entire pattern, including wildcards and UNIX string-matching special characters, cannot exceed 128 characters.

- When adding or changing URL match patterns, check their impact on the CSG's memory:
  1. Enter the **show module csg status** command in privileged EXEC mode to check the status of the configuration change.
  2. When the status changes from PENDING (the change has not yet downloaded) to COMPLETE, SUCCESS (the change has downloaded successfully), enter the **show module csm memory** command in privileged EXEC mode. This command displays the CSG's total memory used versus total memory available.

The following example shows how to specify URL match patterns for map MOVIES. In this example, the URL match is TRUE for ***.movies_comedy.com/*.mpeg**, for ***.movies_action.com/*.mpeg**, and for any other URLs that match the pattern:

```
ip csg map MOVIES url
 match url *.movies_(comedy|action|drama).com/*.mpeg
```


Note The CSG implementation of WAP supports URL maps, but not header maps.

Configuring a Symbolic Weight Name

The same weight can occur in multiple rules, specified in multiple billing services. If a weight changes, and you use numeric constants for weights, each occurrence of the weight must be updated. However, if you define symbolic weight names, you need only update a single definition for each weight. The result is a more readable configuration, and price lists that are easier to manage.

The weight-name is referenced in the **content** command in CSG service configuration mode.

To define a symbolic name for a CSG billing weight, perform this task:

Command	Purpose
`Router(config-csg-module)# ip csg weight weight-name weight-value`	Defines a symbolic name for a CSG billing weight, and enters CSG weight configuration mode.

The following example shows how to define a CSG weight:

```
ip csg weight DOUBLE 2
```

Configuring Advice of Charge, Filtering, and Other Per-Event Authorizations

You can instruct the CSG to get authorization from the quota server for each subscriber request for content.

To configure content authorization, perform this task:

	Command	Purpose
Step 1	`Router(config)# ip csg service service name`	Defines a content billing service, and enters CSG service configuration mode.
Step 2	`Router(config-csg-service)# authorize content`	Instructs the CSG to get authorization from the quota server for each subscriber request for content.

The following example shows how to configure content authorization for the CSG:

```
Router(config)# ip csg service service_name
   authorize content
```

To define the token used for the URL-rewriting feature of AoC, perform the following task:

	Command	Purpose
Step 1	`Router(config)# ip csg user-group group name`	Creates a group of end-users for which you want to generate accounting records, and enters CSG user group configuration mode.
Step 2	`Router(config-csg-usergroup)# aoc confirmation token`	Configures a token for use in advice of charge (AoC) URL-rewriting.

The following example shows how to specify a token for AoC URL-rewriting:

```
ip csg user-group A1
    aoc confirmation ?CSG_AOC_OK
```

Configuring Quota Server Load-Sharing

The CSG allows load sharing among quota servers, similar to its BMA load-balancing feature. Multiple quota servers can be simultaneously active, and the CSG assigns a quota server to each user.

To configure quota server load-sharing, perform this task:

	Command	Purpose
Step 1	`Router(config)# ip csg user group` *group name*	Creates a group of end-users for which you want to generate accounting records, and enters CSG user group configuration mode.
Step 2	`Router(config-csg-usergroup)# quota activate` *number*	Assigns a quota server to each user. All quota transactions for the user are done with the same quota server. When a quota server fails, all users associated with that quota server are distributed among other quota servers. The valid range for the *number* argument is 1 through 10.

The following example shows how to define quota server load-sharing:

```
router(config)# ip csg user u1
router(config-csg-usergroup)# quota activate 5
```

Configuring Service-Level CDR Summarization

By default, the CSG generates billing records for each transaction. This has the potential to overwhelm the Charging Gateway or the collector. To prevent this situation, the CSG can summarize CDRs at the service level, instead of at the transaction level.

To configure service-level CDR summarization, perform this task:

	Command	Purpose
Step 1	`Router(config-csg-module)# ip csg service` *service-name*	Defines a content billing service, and enters CSG service configuration mode.
Step 2	`Router(config-csg-service)# records granularity {transaction | service {bytes` *bytes* `| time` *seconds* `| bytes` *bytes* `time` *seconds*`}}`	Specifies the granularity at which billing records (CDRs) should be generated. For service-level CDR summarization, specify the **service** keyword.

Note If you specify both **type http** and any other type (**type other**, **type ftp**, **type imap**, and so on) for a service, and you enable service-level CDR summarization for the service, the CSG's incremental and cumulative byte counts are not valid, because they are a mix of TCP bytes (for the HTTP traffic) and IP bytes (for all other traffic).

Configuring Quota Server Reauthorization

After the CSG receives a grant of zero quadrans in a Service Authorization Response, the CSG waits for an interval of time before it requests quota in a Service Reauthorization Request. To configure the initial minimum interval before the CSG sends a Service Reauthorization Request, perform this task:

	Command	Purpose
Step 1	`Router(config)# module csg slot`	Enters module CSG configuration mode for a specified slot.
Step 2	`Router(config-csg-module)# variable CSG_ZERO_QUOTA_TIMEOUT_INIT timeout`	Sets the maximum timeout for reauthorization after quota grant of zero.

For each consecutive grant of zero quadrans in a Service Authorization Response from the quota server, the CSG doubles the retry timeout. If the quota server grants any value for quota greater than zero in a Service Authorization Response, the CSG uses the initial value for retry interval after the next zero quota grant.

Note Service Authorization messages have a usage of zero for RTSP traffic.

To configure the maximum retry timeout value, perform the following task:

	Command	Purpose
Step 1	`Router(config)# module csg slot`	Enters module CSG configuration mode for a specified slot.
Step 2	`Router(config-csg-module)# variable CSG_ZERO_QUOTA_TIMEOUT_MAX timeout`	Sets the initial timeout for reauthorization after quota grant of zero.

Note If the INIT value is greater than the MAX value, the MAX value is used as the minimum retry interval and the INIT value is ignored.

To configure the maximum values for the threshold of available quota for sending a Service Reauthorization Request, perform the following task:

	Command	Purpose
Step 1	`Router(config)# module csg slot`	Enters module CSG configuration mode for a specified slot.
Step 2	`Router(config-csg-module)# variable CSG_BASIS_BYTE_LOW_QUOTA_MAX max_threshold`	Sets the maximum value for the available quota threshold that triggers reauthorization for basis byte.
Step 3	`Router(config-csg-module)# variable CSG_BASIS_FIXED_LOW_QUOTA_MAX max_threshold`	Sets the maximum value for the available quota threshold that triggers reauthorization for basis fixed.

The formula for calculating the reauthorization thresholds are:

- For volume-basis billing, the threshold is the smallest of the following values:
  - csg_basis_byte_low_quota_max
  - last_quota_grant /4
  - 32 KB
- For fixed-basis billing, the threshold is the smallest of the following values:
  - csg_basis_fixed_low_quota_max
  - last_quota_grant /4

Protocol-Specific Configuration Details

This section provides information about the following tasks:

Configuring WAP/WSP Support

The CSG can intercept Wireless Application Protocol (WAP) traffic and generate reports that include contextual WAP information and counts of the bytes transferred. This feature supports both prepaid and postpaid billing. This section provides the following information:

Counting Bytes and Packets

The CSG reports WAP datagram sizes (including IP and UDP headers), the number of IP packets per transaction, and PDU counts. (The PDU count is not the same as the packet count. Multiple WAP PDUs can share a single packet.) Bytes for retransmitted WAP PDUs and segments are counted and listed

separately from non-retransmitted counts in the billing reports. Byte and PDU counts are further specified by source. Reports include the number of bytes and PDUs uploaded from source to destination, and downloaded from destination to source.

Incomplete WAP Transactions

When the internal session representing a WAP flow for the CSG expires (due to inactivity or because a WAP DISCONNECT packet is received), any outstanding elements in the WAP transaction queue are reported. These are transactions that were not completed for some reason. Examples include a GET request for which a full REPLY was not received, or a segmented POST or PUSH that was incomplete (missing a segment). In such cases, the incomplete flag is set on the Wireless Transaction Protocol (WTP) Info Tag-Length-Value (TLV) in the WAP statistics record. The record reports the Wireless Session Protocol (WSP) PDU type, WTP transaction class, WTP transaction ID, and the number of IP bytes transferred during the attempted transaction.

Multimedia Messaging Service (MMS)

Multimedia Messaging Service (MMS) traffic running over WAP is differentiated from other WAP traffic by inspecting the Wireless Session Protocol (WSP) Content Type. If MMS prepaid charging is disabled, all MMS traffic flows even when non-MMS, WAP traffic is blocked due to insufficient quota. Postpaid reports for MMS are generated as for all WAP traffic.

Typically, several WAP packets are exchanged during a transaction before the WSP Content Type can be identified. In situations where prepaid WAP with free MMS is configured, some packets still flow (even if a user has insufficient quota) in order to make this determination. But the transaction does not complete, and the user does not receive content if he or she has insufficient quota for a non-MMS, WAP request.

It is not always possible to determine the WSP Content Type for incomplete transactions. In these instances, no quota is deducted for prepaid users.

Configuring the CSG to Monitor and Generate WAP Reports

To enable the CSG to monitor and generate reports for WAP traffic, perform the following task:

	Command	Purpose
Step 1	`Router(config-csg-module)# ip csg policy` *POLICY_NAME*	Defines a policy for qualifying flows for the CSG accounting services, and enters CSG policy configuration mode.
Step 2	`Router(config-csg-policy)# accounting type wap {connection-oriented | connectionless} [customer-string` *string value*`]`	Defines the accounting type and customer string for all flows that comply with a CSG billing policy.

The following example shows how to enable the CSG to monitor and generate reports for WAP traffic:

```
ip csg policy WAP_CLT_POLICY
  accounting type wap connection-oriented customer-string to_wap_client
```

Note You cannot mix **type wap** with any other types. If one of the policies is **wap** they all must be **wap**.

WAP is only supported for CSG-style configurations—using content and not virtual servers.

Configuring Connection-Oriented and Connectionless WAP

The accounting types **wap connection-oriented** and **wap connectionless** specify how the WAP traffic for that port should be interpreted. To configure **wap connection-oriented** or **wap connectionless**, perform the following task:

	Command	Purpose
Step 1	`Router(config)# ip csg policy policy-name`	Defines a policy for qualifying flows for the CSG accounting services, and enters CSG policy configuration mode.
Step 2	`Router(config-csg-policy)# accounting type wap {connection-oriented | connectionless} [customer-string string]`	Defines the accounting type and customer string for all flows that comply with a CSG billing policy.

The following example shows how to define both connection-oriented and connectionless WAP accounting:

```
ip csg policy WSP_CON_P
 accounting type wap connection-oriented

ip csg policy WAP_NOCON_P
 accounting type wap connectionless

ip csg content WAP_CON
 ip any udp 9201
 policy WAP_CON_P

ip csg content WAP_CONLESS
 ip any udp 9200
 policy WAP_NOCON_P
```

Prepaid Support

Some upstream WAP browsing traffic occurs because the CSG must inspect the reply before determining that the traffic is an MMS transaction. However, the downstream WAP browsing replies are discarded if quota is depleted.

Control information is charged against quota for non-MMS transactions. WSP PDU types SUSPEND and RESUME are never charged against quota.

Redirect

The CSG can redirect client flows to an alternate IP address or URL when the client's quota is exhausted. Once configured, the CSG redirects client requests to another server that informs the user that the quota has been exceeded, and describes any appropriate actions to take.

To configure the redirect option, perform the following task:

	Command	Purpose
Step 1	`Router(config)# ip csg user-group` *group-name*	Creates a group of end-users for which you want to generate accounting records, and allows you to enter CSG user group configuration mode.
Step 2	`Router(config-csg-usergroup)# redirect nat` *ip-address*	Redirects NAT client flows to an alternate IP address when the client's quota is exhausted.
Step 3	`Router(config-csg-usergroup)# redirect wap` *url*	Redirects WAP client flows to an alternate URL when the client's quota is exhausted.

WAP redirect requires that you configure a policy and service so a client who has exhausted quota can access the server specified in the redirect URL.

The following example shows how to define the redirect option for WAP, and to allow redirected WAP traffic to pass without charge:

```
ip csg user-group A1
 database 10.18.12.214 3311
 radius key secret-key
 quota local-port 7788
 redirect wap http://www.topoff.com
 quota server 10.10.1.203 7777 1
ip csg map TOPOFF url
  match protocol http url http://www.topoff.com*
!
ip csg policy URL_TOPOFF
  accounting type wap connection-oriented customer-string topoff
  url-map TOPOFF
!
ip csg content WAP_WTP_CONTENT
  ip any udp 9201
  policy URL_TOPOFF
  inservice
!
ip csg weight ZERO 0
!
ip csg service FREE
  content WAP_WTP_CONTENT policy URL_TOPOFF weight ZERO
```

Disabling Prepaid MMS Billing

By default the CSG treats MMS traffic like any other WAP traffic and generates prepaid and postpaid WAP statistics reports for it. The content type distinguishes it as MMS traffic. You can disable MMS prepaid billing by performing the following task:

	Command	Purpose
Step 1	`Router(config)# ip csg service service-name`	Defines a content billing service, and enters CSG service configuration mode.
Step 2	`Router(config-csg-service)# basis byte {ip exclude mms | fixed exclude mms}`	Specifies the billing basis for a CSG content billing service. This example illustrates how to exclude prepaid billing of MMS content for volume- or fixed-basis users.
Step 3	`Router(config-csg-service)# content content-name policy policy-name`	Defines content as a member of a CSG billing service, identifies a policy to apply to this content, and optionally assigns a weight to this content.

The following example shows how to disable MMS traffic from prepaid volume billing:

```
ip csg service SERVIN_WAP
 basis byte ip exclude mms
 content WAP_CLIENT policy WAP_CLT_POLICY
 content WAP_WSP_SRV policy WAP_SRV_POLICY
 content WAP_WTP_SRV policy WAP_SRV_POLICY
```

Note You can also use **basis fixed exclude mms** to disable prepaid billing for fixed-basis billing.

Configuring the CSG SMTP and POP3 Data Mining

The CSG can report SMTP and POP3 data records. To configure SMTP or POP3 data mining on the CSG, perform the following tasks:

	Command	Purpose
Step 1	`Router(config)# ip csg policy policy-name`	Defines a policy for qualifying flows for the CSG accounting services, and enters CSG policy configuration mode.
Step 2	`Router(config-csg-policy)# accounting type [smtp | pop3] [customer-string string]`	Defines the accounting type and customer string for all flows that comply with a CSG billing policy.

The following example shows how to enable the reporting of SMTP and POP3 data records on the CSG:

```
ip csg policy SMTP
  accounting type smtp

ip csg policy POP3
  accounting type pop3

ip csg content SMTP
  ip any tcp 25
  policy SMTP
  inservice
```

```
ip csg content POP3
  ip any tcp 110
  policy POP3
  inservice
```

Configuring RTSP Billing

RTSP billing correlates various streams associated with an RTSP session, and reports application-level information (for example, filename) to the billing system.

To configure RTSP billing on the CSG, enter the following command in CSG policy configuration mode:

Command	Purpose
`Router(config-csg-policy)#` **`accounting type rtsp`** [**`customer-string`** *`string`*]	Defines the accounting type as RTSP, and optionally the customer string for all flows that comply with a CSG billing policy. Prepaid service matches are based on the IP address and port number of the control connection to the RTSP server IP.

The following example shows how to configure RTSP billing:

```
ip csg policy RTSP
  accounting type rtsp

ip csg content RTSP
  ip any tcp 554
  policy RTSP
  inservice
```

When configuring RTSP billing, keep the following considerations in mind:

- The CSG supports only port 554 for RTSP billing.
- RealPlayer clients ignore the explicit definition of port 554 in the URL and attempt to connect to ports 554, 7070, 80, and 8080. Many other streaming media servers also listen on ports 7070, 80, and 8080. For HTTP transport, if the media streams from any port other than port 554 (such as port 7070, 80, or 8080), the CSG does not bill the stream as RTSP. Therefore, for RTSP billing, you must block TCP and HTTP connections to the server network on ports 80, 8080 and 7070. For more information about blocking ports, see the "Blocking Ports" section on page 3-33.
- HTTP should be your last choice for RTSP transport.
- When using HTTP as the transport for RTSP, the control connection might time out, causing the stream to hang.
- This occurs because, when handling RTSP over HTTP, the client opens two TCP connections, one for the main content and one for control. The client uses the control connection sparingly, which can result in the connection timing out. To prevent this problem, ensure that the idle content timer has a duration of at least 60 seconds (the default setting is 3600 seconds). For more information on setting the idle content timer, see the description of the **idle** command in CSG content configuration mode.

  This is not an issue when using UDP or TCP as the transport.

Blocking Ports

To block a port, specify a content definition that matches the connection to the server network and a policy that sends transactions to a false next-hop IP address, as shown in the following example:

```
ip csg policy RTSP
  accounting type rtsp
!
ip csg policy RTSP-BLOCK
  next-hop 10.10.10.1
!
ip csg content BLOCK7070
  ip 1.1.1.0 255.255.255.0 tcp 7070
  policy RTSP-BLOCK
  inservice
!
ip csg content BLOCK80
  ip 1.1.1.0 255.255.255.0 tcp 80
  policy RTSP-BLOCK
  inservice
!
ip csg content BLOCK8080
  ip 1.1.1.0 255.255.255.0 tcp 8080
  policy RTSP-BLOCK
  inservice
!
ip csg content RTSPCONTSERVER
  ip 1.1.1.0 255.255.255.0 tcp 554
  idle 50
  replicate
  policy RTSP
  inservice
```

Configuring Connection Duration Billing

Connection Duration Billing enables the CSG to deduct quota based on the time that a user is logged on to the IP network.

To configure the Connection Duration Billing feature on the CSG, specify the following commands in CSG service configuration mode:

	Command	Purpose
Step 1	`Router(config-csg-service)# basis second connect [exclude mms]`	Specifies Connection Duration Billing for a CSG content billing service. **Note** When changing the basis for a service, the content must be taken out of service.
Step 1	`Router(config-csg-service)# activation [automatic | user-profile]`	Specifies the activation mode for a Connection Duration service.

The following commands are used to configure Connection Duration Billing for the **OFF_NET** service, with **automatic** activation:

```
ip csg service OFF_NET
  basis second connect
  activation automatic
```

Enabling Passthrough Mode for a Service

To enable passthrough mode for a service, specify the following command in CSG service configuration mode:

	Command	Purpose
Step 1	`Router(config-csg-service)# passthrough` *quota-grant*	Enables passthrough mode for a service.

The following example specifies that the CSG grants 65,535 quadrans of quota to the service NAME each time the service runs low on quota:

```
ip csg service NAME
  passthrough 65535
```

Configuring SNMP Timers

The CSG enables you to configure SNMP timers for lost CSG records.

To configure an SNMP timer, and to enter CSG SNMP timer configuration mode, specify the following command in global configuration mode:

Command	Purpose
`Router(config)# ip csg snmp timer` {**agent** | **quota-server**} [*interval*]	Defines SNMP timers for lost CSG records, and enter s CSG SNMP timer configuration mode.

The following example defines a 300-second CSG SNMP agent timer and enters CSG SNMP timer configuration mode:

```
ip csg snmp timer agent 300
```

Configuring the Idle Content Timer for UDP and WAP 1.x

To configure an idle content timer, specify the following command in CSG content configuration mode:

Command	Purpose
`Router(config-csg-content)# idle` *duration*	Specifies the minimum amount of time that the CSG maintains an idle content connection.

The following example shows how to configure a 120-second idle timer for the CSG content MOVIES_COMEDY:

```
ip csg content MOVIES_COMEDY
 idle 120
```

The CSG tracks usage on a per-session basis. UDP protocols do not have an end-of-session indicator and simply idle out. For that reason, for UDP and WAP 1.x, setting the content idle timer to a low value (for example, 30 seconds) allows the CSG to quickly recognize that a session has ended and generate billing records accordingly. Other service-level features of the CSG that count sessions (such as passthrough mode and service-level CDRs) are similarly affected by the content idle timer setting.

Other Configuration Tasks

The following sections provide additional information to help you configure the CSG. The sections include:

Configuring the CSG and PSD

The configuration tasks required to establish communication between the CSG and the PSD involve several steps that go beyond the scope of this chapter. For specific information on how to configure the CSG and the PSD, see Appendix A, "PSD Configuration for the CSG."

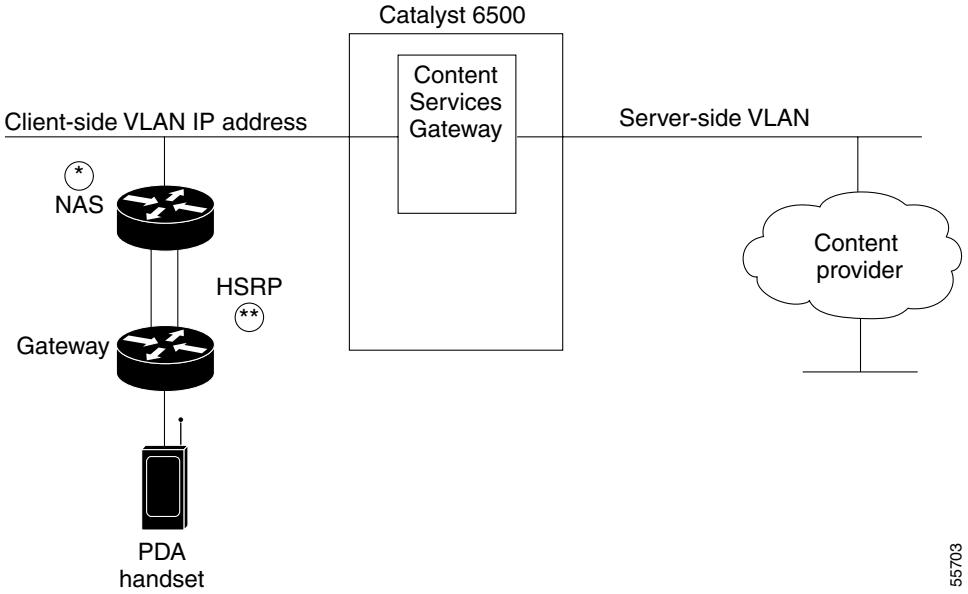
Configuring VLANs

Clients and servers communicate through the CSG using Layer 2 and Layer 3 technology in a specific VLAN configuration. Clients connect to the client-side VLAN, and servers connect to the server-side VLAN. Servers and clients exist on different subnets. Servers can also be located one or more Layer 3 hops away and connect to the server-side VLAN through routers. This section describes how to configure VLANs for the CSG.

A client sends a request to one of the module's server addresses. The CSG extracts the URL—if applicable—and records the statistics. When properly configured, the CSG records statistics for flows in both directions. When a connection ends, the CSG builds an accounting record and sends it to the BMA.

When you install the CSG in a Catalyst 6500 series switch, you must configure client-side and server-side VLANs. (See Figure 3-1.)

Figure 3-1 Configuring VLANs



*Any router configured as a client-side gateway, or a next-hop router for servers more than one hop away, must have ICMP redirects disabled. The CSG does not perform a Layer 3 lookup to forward traffic; the CSG cannot act upon ICMP redirects.

** You can configure up to seven gateways per VLAN for up to 256 VLANs and up to 224 gateways for the entire system. If an HSRP/VRRP gateway is configured, the CSG uses three gateway entries out of the 224 gateway entries because traffic can come from the virtual and physical MAC addresses of the HSRP group. (See the "HSRP Configuration Overview" section on page 4-9.)

Note You must configure VLANs on the Catalyst 6000 series switch or Cisco 7600 series router *before* you configure VLANs for the CSG. VLAN IDs must be the same for the switch and the module.

You must create both a client-side and server-side VLAN:

- Configuring Client-Side VLANs, page 3-37
- Configuring Server-Side VLANs, page 3-37

Configuring Client-Side VLANs

To configure client-side VLANs, perform this task:

	Command	Purpose
Step 1	`Router(config-csg-module)# vlan vlan-id client [vlan-name]`	Configures the client-side VLANs and enters the client VLAN mode. **Note** Do not use VLAN 1 as a client-side VLAN for the CSG.
Step 2	`Router(config-csg-vlan-client)# ip address ip-address netmask`	Configures an IP address to the CSG used by probes and Address Resolution Protocol (ARP) requests on this particular VLAN.
Step 3	`Router(config-csg-vlan-client)# gateway ip-address`	Configures the gateway IP address.

Note You cannot use VLAN 1 as a client-side or server-side VLAN for the CSG.

The following example shows how to configure the CSG for client-side VLANs:

```
Router(config-module-csg)# vlan 130 client
Router(config-csg-vlan-client)# ip address 123.44.50.6 255.255.255.0
Router(config-csg-vlan-client)# gateway 123.44.50.1
Router(config-csg-vlan-client)# exit
```

Configuring Server-Side VLANs

To configure server-side VLANs, perform this task:

	Command	Purpose
Step 1	`Router(config-csg-module)# vlan vlan-id server [vlan-name]`	Configures the server-side VLANs and enters the server VLAN mode. **Note** Do not use VLAN 1 as a server-side VLAN for the CSG.
Step 2	`Router(config-csg-vlan-server)# ip address ip-address netmask`	Configures an IP address for the server VLAN.
Step 3	`Router(config-csg-vlan-server)# alias ip-address netmask`	(Optional) Configures multiple IP addresses to the CSG as alternate gateways for the real server. The alias is required in the redundant configuration.
Step 4	`Router(config-csg-vlan-server)# route ip-address netmask gateway gw-ip-address`	Configures a static route to reach the real servers if they are more than one Layer 3 hop away from the CSG. **Note** If you are adding a new route to an existing gateway, the new route might not take effect until you remove the gateway and reconfigure it to clear the gateway cached entries.

The following example shows how to configure the CSG for server-side VLANs:

```
Router(config-module-csg)# vlan 150 server
Router(config-csg-vlan-server)# ip address 123.46.50.6 255.255.255.0
Router(config-csg-vlan-server)# alias 123.60.7.6 255.255.255.0
Router(config-csg-vlan-server)# route 123.50.0.0 255.255.0.0 gateway 123.44.50.1
Router(config-csg-vlan-server)# exit
```

Preventing Pipelined Requests

Note This procedure is no longer necessary in the CSG 3.1(3)C5(5) and later. It is made obsolete by the CSG's full HTTP pipelining support.

Some customers have handsets that attempt to make pipelined HTTP requests. Because this is not supported prior to CSG 3.1(3)C5(5), the CSG enables you to prevent HTTP pipelined requests by disabling HTTP persistence. This is done by applying TCP FIN to the final response packet to force establishing a new session for each request. The final response packet is identified using the Content-Length: field in the HTTP header, and support was not added to detect the final packet when Content-Length: is not present (as when using Transfer-Encoding: chunked). So, the CSG prevents chunked encoded responses by overwriting the HTTP version in the request to HTTP/1.0. Because chunked encoding is not supported in HTTP/1.0, an HTTP/1.1 server is not allowed to respond with chunked data.

To disable persistence, enter the following commands in module CSG configuration mode:

	Command	Purpose
Step 1	`Router(config-csg-module)# variable CSG_HTTP_PERSISTENCE_DISABLE 0`	Configures setting the FIN bit at end of responses. To disable this variable and prevent HTTP pipelined requests, set this variable to 0.
Step 2	`Router(config-csg-module)# variable CSG_HTTP_1_0_OPERATION 0`	Overwrites the HTTP version to 1.0 on GETs and responses. To disable this variable and prevent HTTP pipelined requests, set this variable to 0.

Configuring Layer 2-Adjacent Devices

Note If a CSG receives a packet with a Layer 2 address it does not recognize, from a device that has a layer 3 address that is not on the same IP subnet as the CSG, it drops the packet. If the CSG already has an Address Resolution Protocol (ARP) cache entry for the Layer 2 source address, it processes the packet normally. This behavior can be a problem if there are Layer 2-adjacent devices that are performing redundancy (for example, HSRP or Virtual Router Redundancy Protocol [VRRP] firewalls).

In a typical network environment, all traffic flows between clients and servers and uses the primary device/firewall. When traffic is coming *from* the device/firewall to the CSG, the source MAC can be that of the physical interface on that device rather than the MAC associated with the virtual IP address that

is shared between the two devices/firewalls. If there is a failover of the second device/firewall, traffic is routed through the backup device/firewall. If the CSG does not have an ARP entry in its ARP cache for the MAC address of the now-active device/firewall, it drops packets received from that device/firewall.

To avoid this behavior, configure static routes on the CSG that point to the IP addresses on the interfaces of the adjacent devices/firewalls. For example, if the CSG is Layer 2-adjacent to two firewalls, and the IP addresses on those firewalls are 1.1.1.5 and 1.1.1.6, configure the following on the CSG:

route *IP address not-in-use on the network* 255.255.255.255 gateway 1.1.1.5
route *IP address not-in-use on the network* 255.255.255.255 gateway 1.1.1.6

This causes the CSG to spawn an ARP for 1.1.1.5 and 1.1.1.6 so that it has an ARP entry in its ARP cache for both firewalls. In the event of a failover, the packets received from the now-active firewall have a source MAC that is in the ARP cache of the CSG.

Configuration Examples

This section includes the following examples:

Sample CSG Billing Rules

Table 3-1 shows sample CSG billing rules.

Table 3-1 Sample CSG Billing Rules

Content Specification	Service/Billing Basis	Quadrans per Unit
IP/Netmask = 1.2.3.4/24 Protocol/Port Number = TCP/80 HostName = *.books-co-inc.com URL = *.jpg	Service = BillByVolume Basis = TCP Volume	1
IP/Netmask = 1.2.3.4/24 Protocol/Port Number = TCP/80 HostName = *.books-co-inc.com URL = *freecontent*	Service = BillPerClick Basis = Constant	0

Table 3-1 Sample CSG Billing Rules (continued)

Content Specification	Service/Billing Basis	Quadrans per Unit
IP/Netmask = 1.2.3.4/24 Protocol/Port Number = TCP/80 HostName = *.advt-co.com URL = *	Service = Advertisements Basis = Constant	-1
IP/Netmask = 198.133.219.0/24 Protocol/Port Number = TCP/80 HostName = *bigcorp* URL = *	Service = Corporate Basis = Constant	0
IP/Netmask = 0.0.0.0/0 Protocol/Port Number = TCP/80 HostName = * URL = *	Service = Internet Basis = IP Volume	1

The following example shows how to configure these CSG billing rules:

```
ip csg user-group U1
 entries max 10000
 radius key cisco
 radius acct-port 23385
 radius userid User-Name
 quota local-port 4095
 quota server 20.20.50.13 3386 5
 quota server 20.20.50.130 3386 6
 quota server 20.20.52.13 3386 7
!
ip csg accounting CSGBILL
 user-group U1
 records max 2000
 agent activate 2 sticky 30
 records intermediate bytes 50000
 agent 9.15.72.5 3386 2
 agent 10.76.86.2 3386 5
!
 agent 20.20.50.131 3386 8
 inservice
!
ip csg map ADVERTISEMENTS header
 match header Host header-value *.advt-co.com
!
ip csg map ALLHOSTS header
 match header Host header-value *
!
ip csg map BOOKS header
 match header Host header-value *.books-co-inc.com
!
ip csg map CORPORATE header
 match header Host header-value *bigcorp*
!
ip csg map ALLURLS url
 match url *
!
```

```
ip csg map FREE url
 match url *freecontent*
!
ip csg map JPGS url
 match url *.jpg
!
ip csg map GIF url
 match url *.gif
!
ip csg policy ADVERTISEMENTS
 accounting type http
 url-map ALLURLS
 header-map ADVERTISEMENTS
!
ip csg policy BOOKFREE
 accounting type http
 url-map BOOKFREE
 header-map BOOKS
!
ip csg policy BOOKSALES
 accounting type http
 url-map JPGS
 header-map BOOKS
!
ip csg policy CORPORATE
 accounting type http
 url-map ALLURLS
 header-map CORPORATE
!
ip csg policy INTERNET
 accounting type http
 url-map ALLURLS
 header-map ALLHOSTS
!
ip csg content ADVERTISEMENTS
 ip 1.2.5.0 255.255.255.0 tcp 80
 policy ADVERTISEMENTS
 inservice
!
ip csg content BOOKS
 ip 1.2.3.0 255.255.255.0 tcp 80
 policy BOOKSALES
 policy BOOKFREE
 inservice
!

ip csg content CORPORATE
 ip 198.133.219.0 255.255.255.0 tcp 80
 policy CORPORATE
 inservice
!
ip csg content INTERNET
 ip any tcp 80
 policy INTERNET
 inservice
!
ip csg ruleset R1
 content ADVERTISEMENTS
 content BOOKS
 content CORPORATE
 content INTERNET
!
ip csg weight FREE 0
ip csg weight PAYBACK -1
```

```
!
ip csg service BILLPERCLICK
 basis fixed
 content ADVERTISEMENTS policy ADVERTISEMENTS weight PAYBACK
 content BOOKS policy BOOKSALES
 content BOOKS policy BOOKFREE weight FREE
 content CORPORATE policy CORPORATE weight FREE
!
ip csg service BILLBYVOLUME
 basis byte tcp
 content BILLBYVOLUME policy BILLBYVOLUME
!
ip csg service BILLBYIPVOLUME
 basis byte
 content INTERNET policy INTERNET
!
ip csg billing PLAN1
 service BILLPERCLICK
 service BILLBYVOLUME
 service BILLBYIPVOLUME
!

module ContentServicesGateway 5
 vlan 30 client AUCTION_HOUSE
  ip address 123.44.50.6 255.255.255.0
  gateway 123.44.50.1
!
 vlan 40 server
  ip address 123.46.50.6 255.255.255.0
!
 ruleset R1
 accounting CSGBILL
```

Simple Postpaid Billing Configuration Example

The following example shows a simple postpaid billing CSG configuration:

```
ip csg policy POLICY1
 accounting type http
!
ip csg content MOVIES_COMEDY
 ip 172.18.45.0/24 tcp 8080
 policy POLICY1
 inservice
!
ip csg content AUCTION_HOUSE
 ip 216.32.120.0/24 tcp 8080
 policy POLICY1
 vlan AUCTION_HOUSE
 inservice
!
ip csg content WAKETECH
 ip 48.33.0.0/16 tcp 80
 policy POLICY1
 inservice
!
ip csg ruleset R1
 content MOVIES_COMEDY
 content AUCTION_HOUSE
 content WAKETECH
!
ip csg user-group G1
```

```
 entries max 100000
 database 10.1.2.3 11111
 radius key secretpassword
!
ip csg accounting A1
 user-group G1
 agent localport 3775
 agent 10.1.2.4 11112 1
 agent 10.1.2.5 11113 2
 agent activate 2
 records max 250
 inservice
!
mod csg 4
 vlan 30 client AUCTION_HOUSE
  ip address 123.44.50.6 255.255.255.0
  gateway 123.44.50.1
 vlan 40 server
  ip address 123.46.50.6 255.255.255.0
  alias 123.60.7.6 255.255.255.0
  route 123.50.0.0 255.255.0.0 gateway 123.44.50.1
 ruleset R1
 accounting A1
```

Basic WAP Configuration Example

The following example illustrates a basic CSG WAP configuration that provides the following functions:

- Charges a fixed rate for all WAP and MMS transactions for which a URL is used.
- Allows requests that are not content-based (control flows) to go through for free.
- Uses a single service for all traffic.

```
ip csg map DEFAULT_URL url
 match protocol http url http://*
!
ip csg policy WAP_URL
 accounting type wap connection-oriented
 url-map DEFAULT_URL
!
ip csg policy WAP_CONTROL
 accounting type wap connection-oriented customer-string control_flow
!
ip csg content WAP_WTP_CONTENT
 ip any udp 9201
 idle 30
 policy WAP_URL
 policy WAP_CONTROL
 inservice
!
ip csg weight ZERO 0
!
ip csg service WAP
 basis fixed
 content WAP_WTP_CONTENT policy WAP_URL
 content WAP_WTP_CONTENT policy WAP_CONTROL weight ZERO
```

Redirect to Top-Off Server Configuration Example

The following example illustrates a WAP configuration with additions to support redirect to a top-off server. This configuration provides the following functions:

- Allows redirect requests to the top-off server to go through for free.
- Defines a second service to be used only for free transactions.

Note This configuration is required to allow redirect to work properly.

Users must also be authorized to use this service by the quota server.

No quota needs to be given out for this service, but a cause code of 0x04 (user authorized) must be returned for the transaction to be allowed through.

```
ip csg map TOPOFF url
 match protocol http url http://www.topoff.com*
!
ip csg map DEFAULT_URL url
 match protocol http url http://*
!
ip csg policy URL_TOPOFF
 accounting type wap connection-oriented customer-string topoff
 url-map TOPOFF
!
ip csg policy WAP_URL
 accounting type wap connection-oriented customer-string
 url-map DEFAULT_URL
!
ip csg policy WAP_CONTROL
 accounting type wap connection-oriented customer-string control_flow
!
ip csg content WAP_WTP_CONTENT
 ip any udp 9201
 idle 30
 policy URL_TOPOFF
 policy WAP_URL
 policy WAP_CONTROL
 inservice
!
ip csg weight ZERO 0
!
ip csg service WAP
 basis fixed
 content WAP_WTP_CONTENT policy WAP_URL
!
ip csg service FREE
 content WAP_WTP_CONTENT policy URL_TOPOFF weight ZERO
 content WAP_WTP_CONTENT policy WAP_CONTROL weight ZERO
```

Free MMS Transactions Configuration Example

Specific MMS Transactions

The following example illustrates a WAP 1 or MMS/WAP1.x configuration in which MMS transactions to servers mms1 and mms2 are free, while third-party MMS transactions are charged.

```
ip csg map TOPOFF url
```

```
 match protocol http url http://www.topoff.com*
!
ip csg map OUR_MMS url
 match protocol http url http://www.mms1*
 match protocol http url http://www.mms2*
!
ip csg map DEFAULT_URL url
 match protocol http url http://*
!
ip csg policy URL_TOPOFF
 accounting type wap connection-oriented customer-string topoff
 url-map TOPOFF
!
ip csg policy FREE_MMS
 accounting type wap connection-oriented customer-string free_mms
 url-map OUR_MMS
!
ip csg policy WAP_URL
 accounting type wap connection-oriented customer-string
 url-map DEFAULT_URL
!
ip csg policy WAP_CONTROL
 accounting type wap connection-oriented customer-string control_flow
!
ip csg content WAP_WTP_CONTENT
 ip any udp 9201
 idle 30
 policy URL_TOPOFF
 policy FREE_MMS
 policy WAP_URL
 policy WAP_CONTROL
 inservice
!
ip csg weight ZERO 0
!
ip csg service WAP
 basis fixed
 content WAP_WTP_CONTENT policy WAP_URL
!
ip csg service FREE
 content WAP_WTP_CONTENT policy URL_TOPOFF weight ZERO
 content WAP_WTP_CONTENT policy FREE_MMS weight ZERO
 content WAP_WTP_CONTENT policy WAP_CONTROL weight ZERO
```

All MMS Transactions

The following example illustrates a WAP 1 or MMS/WAP1.x configuration in which all MMS transactions are free. In this example, MMS content is free for service WAP (the user must be authorized for this service).

```
ip csg map TOPOFF url
 match protocol http url http://www.topoff.com*
!
ip csg map DEFAULT_URL url
 match protocol http url http://*
!
ip csg policy URL_TOPOFF
 accounting type wap connection-oriented customer-string topoff
 url-map TOPOFF
!
ip csg policy WAP_URL
 accounting type wap connection-oriented customer-string
 url-map DEFAULT_URL
!
```

```
ip csg policy WAP_CONTROL
 accounting type wap connection-oriented customer-string control_flow
!
ip csg content WAP_WTP_CONTENT
 ip any udp 9201
 idle 30
 policy URL_TOPOFF
 policy WAP_URL
 policy WAP_CONTROL
 inservice
!
ip csg weight ZERO 0
!
ip csg service WAP
  basis fixed exclude mms
  content WAP_WTP_CONTENT policy WAP_URL
!
ip csg service FREE
  content WAP_WTP_CONTENT policy URL_TOPOFF weight ZERO
  content WAP_WTP_CONTENT policy WAP_CONTROL weight ZERO
```

Differentiating MMS Over WAP 2 Example

The following example assumes that the quota server and the accounting agent are already configured for the system. It also assumes that the WAP Proxy can be found on port 9401 on a host addressable using a server VLAN configured to access the subnet 10.10.2.0/24. This example illustrates the differences in a working configuration necessary to differentiate billing WAP2/HTTP and MMS/WAP2/HTTP.

```
ip csg map WAP2MMS_GET_MAP url
 match protocol http method GET url /wap/mms*

ip csg map WAP2MMS_POSTMAP url
 match protocol http method POST url /wap/mms*

ip csg map WAP2MMSPOSTMAPH header
 match protocol http header Content-Type header-value application/vnd.wap.mms-message

ip csg policy WAP2_MMS_GET
! match all wap2/http gets of mms
  accounting type http customer-string wap2mms-get
  url-map WAP2MMS_GET_MAP

ip csg policy WAP2_MMS_POST
! match all wap2/http posts that are mms related
! This catches handset-initiated MMS sends and acknowledgements of
! network-initiated MMS pushes.
  accounting type http customer-string wap2mms-post
  header-map WAP2MMSPOSTMAPH  ! recommended
! or
! url-map WAP2MMS_POSTMAP  ! optional
! The header-map catches MMS even when it goes to an unknown URL,
! so it is recommended over the url-map.

ip csg policy WAP2
! You might choose to differentiate non-MMS wap2 get/posts and URLs/headers
! here, if relevant. In this case, we just label all remaining traffic as
! wap2.
  accounting type http customer-string wap2

ip csg content WAP2
! 10.10.2.0 255.255.255.0 represents the network where WAP 2 Proxies are
```

```
! located. Port 9401 is the port the WAP 2 Proxies are configured to use.
 ip 10.10.2.0 255.255.255.0 tcp 9401
 policy WAP2_MMS_GET
 policy WAP2_MMS_POST
 policy WAP2
 inservice

! Adjust these to change the pre-paid weight associated with each flow:
ip csg weight WEIGHT_WAP2 3
ip csg weight WEIGHT_WAP2GET 1
ip csg weight WEIGHT_WAP2POST 2

ip csg service WAP2MMSGET
 basis fixed
 idle 10000
 content WAP2 policy WAP2_MMS_GET weight WEIGHT_WAP2GET

ip csg service WAP2MMSPOST
 basis fixed
 idle 10000
 content WAP2 policy WAP2_MMS_POST weight WEIGHT_WAP2POST

ip csg service WAP2
 basis fixed
 idle 10000
 content WAP2 policy WAP2 weight WEIGHT_WAP2

ip csg ruleset R
! other contents
 content WAP2

ip csg billing BILL1
 service WAP2MMSGET
 service WAP2MMSPOST
 service WAP2
```

Pricing by Quota Server Configuration Example

The following example shows a CSG configuration in which all pricing is done by a quota server. In this example:

- Assume that User X has $10.00 in his account.
- There are two types of content:
  - C1—This is billed per object (for example, URL GET), where each object costs $0.01.
  - C2—This is billed per byte, where each KB costs $0.01.
- The quota server controls each object transaction for content C1.
- The quota server controls all the pricing.

```
ip csg content C1
 policy P1
 inservice
!
ip csg content C2
 policy P2
 inservice
!
ip csg service PERCLICK
```

```
 basis fixed
 content C1 policy P1
!
ip csg service PERBYTE
 basis byte ip exclude mms
 content C2 policy P2
!


ip csg billing REGULAR
 service PERCLICK
 service PERBYTE
```

When User X, with a subscription to billing plan REGULAR, tries to access content that matches C1, the CSG tries to download quota for User X for service PERCLICK.

The quota server borrows money from User X's $10.00, and returns some quadrans to the CSG. Each quadran is good for one object download, or one click. If the quota server wants the CSG to query for each click, it can choose to send just one quadran at a time, so that the CSG queries the quota server each time. On the other hand, if the quota server wants to grant $2.00 worth to the CSG in one shot, it can send 200 quadrans to the CSG, which the CSG keeps using for User X's access to C1.

When User X tries to access content that matches C2, the CSG makes another request to the quota server to get User X's quota for C2. C2 is billed per IP byte. The quota server borrows another $5.00 from User X's account, and sends 500000 quadrans to the CSG. As User X continues to access C2, his traffic is metered for volume, and for each byte the CSG deducts one quadran.

Differentiating Prices Configuration Example

The following example extends the previous example by adding an additional content type that is priced differently. In this example:

- Assume that User X has $10.00 in his account.
- There are three types of content:
  - C1—This is billed per *.jpg file, where each JPG file costs $0.01.
  - C2—This is billed per byte, where each KB costs $0.01.
  - C3—This is billed per *.mp3 file, where each MP3 file costs $0.05.
- The quota server controls each object transaction for content C1.
- The quota server controls all the pricing.

This configuration requires an additional service type, MP3, which allows the quota server to price clicks differently for MP3 files.

```
ip csg content C1
 policy P1
 inservice
!
ip csg content C2
 policy P2
 inservice
!
ip csg content MP3
 policy P1
 inservice
!
ip csg service PERCLICK
 basis fixed
```

```
 content C1 policy P1
!
ip csg service PERBYTE
 basis byte ip
 content C2 policy P2
!
ip csg service MP3
 basis fixed
 content C1 policy P1
!
ip csg billing REGULAR
 service PERCLICK
 service PERBYTE
 service MP3
```

When User X tries to download an MP3 file (that is, a file that matches content MP3), the CSG requests the MP3 quota for User X. Each download of an MP3 file costs $0.05, so the quota server borrows $1.00 from User X's account, and returns 20 quadrans to the CSG for service MP3. The CSG can use the quadrans for 20 downloads of MP3 files.

Alternatively, the quota server could send just one quadran, which is good for only one transaction. This would force the CSG to ask for quota before each download of an MP3 file.

Reducing the Number of Services Configuration Example

The "Differentiating Prices Configuration Example" section on page 3-48 showed that you can create a new service for a content and differentiate its billing from other types of content.

However, with each new service, the user's quota fragments further, and traffic between the CSG and the quota server increases.

You can improve this situation by specifying a symbolic weight on the CSG. In this example, each MP3 download ($0.05) costs five times as much as each JPG download ($0.01). By assigning a weight of 5 to MP3 downloads, you can keep both content C1 and content MP3 under service PERCLICK, reducing the overall number of services and reducing the traffic between the CSG and the quota server.

```
ip csg content C1
 policy P1
 inservice
!
ip csg content C2
 policy P2
 inservice
!
ip csg content MP3
 policy P1
 inservice
!
ip csg weight MP3 5
!
ip csg service PERBYTE
 basis byte ip
 content C2 policy P2
!
ip csg service PERCLICK
 basis fixed
 content C1 policy P1
 content MP3 policy P1 weight MP3
!
ip csg billing REGULAR
 service PERCLICK
```

```
service PERBYTE
```

When the quota server borrows $1.00 from User X's account, and sends 100 quadrans for service PERCLICK, the CSG can use the quadrans for 100 JPG files, or for 20 MP3 files, or for a mix of the two.