# Cisco ACNS Software Commands

This chapter contains an alphabetical listing of all the commands in the Cisco ACNS 5.x software. The ACNS software CLI is organized into the following command modes:

- EXEC mode—For setting, viewing, and testing system operations. It is divided into two access levels, user and privileged. To use the privileged access level, enter the **enable** command at the user access level prompt and then enter the privileged EXEC password when you see the password prompt.
- Global configuration mode—For setting, viewing, and testing the configuration of ACNS software features for the entire device. To use this mode, enter the **configure** command from privileged EXEC mode.
- Interface configuration mode—For setting, viewing, and testing the configuration of a specific interface. To use this mode, enter the **interface** command from global configuration mode.
- Other configuration modes—A number of configuration modes are available from the global configuration mode for managing specific features. The commands used to access these modes are marked with a footnote in Table 2-1.

See Chapter 1, "Command-Line Interface Command Summary," for a complete discussion of using CLI command modes.

Table 2-1 summarizes the ACNS commands and indicates the command mode for each command. The commands used to access configuration modes are marked with a footnote in Table 2-1. The same command may have different effects when entered in a different command mode, and for this reason, they are listed and documented separately. In Table 2-1, when the first occurrence is entered in EXEC mode, the second occurrence is entered in global configuration mode. When the first occurrence is entered in global configuration mode, the second occurrence is entered in interface configuration mode.

The ACNS software device mode determines whether the ACNS device is functioning as a Content Engine, Content Distribution Manager, Content Router, or IP/TV Program Manager. The commands available from a specific CLI mode are determined by the ACNS device mode in effect. Table 2-1 also indicates the device mode for each command. *All* indicates that the command is available for every device mode.

**Note** When viewing this guide online, click the name of the command in the left column of the table to jump to the command page, which provides the command syntax, examples, and usage guidelines.

**Note** See Appendix A, "Acronyms" for an expansion of all acronyms used in this publication.

*Table 2-1 CLI Commands*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| aaa accounting | Configures authentication, authorization, and accounting (AAA) for the Content Engine. | global configuration | Content Engine |
| access-lists | Configures the access control list entries. | global configuration | Content Engine |
| acquirer (EXEC) | Configures the content acquirer. | privileged-level EXEC | Content Engine |
| acquirer (global configuration) | Enables authentication when the acquirer obtains content through a proxy server. | global configuration | Content Engine |
| acquisition-distribution | Starts and stops the acquisition and distribution database cleanup process and the content acquisition and distribution process. | privileged-level EXEC | Content Engine |
| alarm overload-detect | Configures the detection of alarm overload. | global configuration | All |
| asset | Configures the asset tag name string. | global configuration | All |
| authentication | Configures the authentication parameters. | global configuration | All |
| auto-register | Enables the discovery of a Fast Ethernet or Gigabit Ethernet interface device and its automatic registration with the Content Distribution Manager through DHCP. | global configuration | Content Engine, Content Router |
| autosense | Sets the current interface to autosense. | interface configuration | All |
| bandwidth (global configuration) | Sets the allowable bandwidth usage and its duration for Cisco Streaming Engine, RealProxy, RealServer, and WMT streaming media. | global configuration | Content Engine |
| bandwidth (interface configuration) | Sets the specified interface bandwidth to 10, 100, or 1000 Mbps. | interface configuration | All |
| banner | Configures the EXEC, login, and message-of-the-day (MOTD) banners. | global configuration | All |
| bitrate | Configures the maximum pacing bit rate for large files sent using HTTP and configures WMT bit-rate settings. | global configuration | Content Engine, Content Router |
| bypass | Configures the bypass functions. | global configuration | Content Engine |
| cache | Specifies the cache commands. | privileged-level EXEC | All |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| cd | Changes the directory. | user-level EXEC and privileged-level EXEC | All |
| cdm | Configures the Content Distribution Manager IP address and primary or standby role settings. | global configuration | All |
| cdnfs | Manages the ACNS network file system (cdnfs). | privileged-level EXEC | All |
| cdp (global configuration) | Enables the Cisco Discovery Protocol (CDP) for the ACNS network device. | global configuration | All |
| cdp (interface configuration) | Enables Cisco Discovery Protocol (CDP) on an interface. | interface configuration | All |
| cfs | Partitions the cache file system. | privileged-level EXEC | All |
| channel | Assigns, creates, deletes, adds, modifies, or otherwise configures a channel. | user-level EXEC and privileged-level EXEC | Content Distribution Manager |
| channel-group | Adds the current interface to an EtherChannel group. | interface configuration | All |
| clear | Resets the counters and other specified functions. | privileged-level EXEC | All |
| clock | Manages the system clock. | privileged-level EXEC | All |
| clock (global configuration) | Sets the summer daylight saving time of day and time zone. | global configuration | All |
| cms (EXEC) | Configures the Centralized Management System (CMS) embedded database parameters. | privileged-level EXEC | All |
| cms (global configuration) | Schedules the maintenance and enables the Centralized Management System on a given node. | global configuration | All |
| configure[1] | Enters configuration mode from privileged EXEC mode. | privileged-level EXEC | All |
| contentrouting | Enables dynamic content routing on a Content Router and on a Content Engine that has simple hybrid routing enabled. | global configuration | Content Engine, Content Router |
| content-routing-api | Enables the content routing API. | global configuration | Content Router |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| copy | Copies the configuration or image files to and from the CD-ROM, flash memory, disk, or remote hosts. | privileged-level EXEC | All |
| cpfile | Copies a file. | user-level EXEC and privileged-level EXEC | All |
| debug | Configures the debugging options. | privileged-level EXEC | All |
| delfile | Deletes a file. | user-level EXEC and privileged-level EXEC | All |
| deltree | Deletes a directory and its subdirectories. | user-level EXEC and privileged-level EXEC | All |
| device | Configures the mode of operation on a device. | global configuration | All |
| dir | Displays the files in a long list format. | user-level EXEC and privileged-level EXEC | All |
| disable | Turns off the privileged EXEC commands. | privileged-level EXEC | All |
| disk (EXEC) | Allocates the disks among the cdnfs, cfs, mediafs, and sysfs file systems. | privileged-level EXEC | All |
| disk (global configuration) | Configures how the disk errors should be handled. | global configuration | All |
| distribution | Reschedules and refreshes the content redistribution through multicast for all channels or a specified channel ID or name. | privileged-level EXEC | Content Engine, Content Router |
| dns | Configures the Content Engine's memory-based DNS cache. | global configuration | Content Engine, Content Distribution Manager |
| dnslookup | Resolves a hostname (DNS). | user-level EXEC and privileged-level EXEC | All |
| enable[1] | Accesses the privileged EXEC commands. | EXEC user | All |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| end | Exits configuration and privileged EXEC modes. | global configuration | All |
| error-handling | Customizes how the Content Engine handles errors. | global configuration | Content Engine, Content Router |
| exception | Enables exception debug mode. | global configuration | Content Engine, Content Router |
| exec-timeout | Configures the length of time that an inactive Telnet or Secure Shell (SSH) session remains open. | global configuration | All |
| exit | Exits from interface, global configuration, or privileged EXEC modes. | All | All |
| external-ip | Configures up to a maximum of eight external IP addresses. | global configuration | All |
| find-pattern | Searches for a particular pattern in a file. | privileged-level EXEC | All |
| ftp-native | Configures the FTP native caching services. | global configuration and user-level EXEC | Content Engine |
| ftp-over-http | Configures the FTP-over-HTTP caching services. | global configuration | Content Engine |
| full-duplex | Sets the current interface to the full-duplex mode. | interface configuration | All |
| gui-server | Configures and enables the Content Engine GUI server. | global configuration | Content Engine, Content Distribution Manager |
| half-duplex | Sets the current interface to half-duplex mode. | interface configuration | All |
| help | Provides assistance for the command-line interface. | user-level EXEC, privileged-level EXEC, and global configuration | All |
| hostname | Configures the Content Engine network name. | global configuration | All |
| http | Configures the HTTP-related parameters. | global configuration | Content Engine, Content Router |
| http custom-error-page | Displays the custom HTTP error messages. | user-level EXEC and privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| https (EXEC) | Creates, removes, and imports the certificates and private keys when the Content Engine is used as an HTTPS server. | privileged-level EXEC | Content Engine, Content Router |
| https (global configuration) | Configures the HTTPS-related parameters. | global configuration | Content Engine, Content Router |
| https server[1] | Enables HTTPS caching and allows a Content Engine to act as an origin HTTPS server. Provides access to the HTTPS configuration mode. | global configuration | Content Engine, Content Router |
| icap | Enables the Internet Content Adaptation Protocol for supporting third-party software applications and plug-ins. | global configuration | Content Engine |
| icap service[1] | Configures ICAP service configurations. Provides access to the ICAP service configuration mode. | global configuration | Content Engine |
| icp | Configures Internet Cache Protocol parameters. | global configuration | Content Engine |
| inetd | Enables FTP, RCP, and TFTP services. | global configuration | All |
| install | Installs a new version of the caching application. | privileged-level EXEC | All |
| interface[1] | Configures a Fast Ethernet, Fibre Channel, Gigabit Ethernet, or port-channel interface. Provides access to interface configuration mode. | global configuration | All |
| ip | Configures the Internet Protocol. | global configuration | All |
| ip access-group | Controls the connections on a specific interface by applying a predefined access list. | interface configuration | Content Engine |
| ip access-list[1] | Creates and modifies the access lists for controlling access to interfaces or applications. Provides access to ACL configuration mode. | global configuration | Content Engine |
| ip address | Configures the IP address, subnet mask, or DHCP IP address negotiation on the Content Engine interface. | interface configuration | All |
| iptv program-manager | Configures the IP/TV Program Manager. | EXEC | Program Manager |
| kernel kdb | Enables the kernel debugger configuration mode. | global configuration | All |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| ldap | Configures the LDAP server parameters. | global configuration | Content Engine |
| less | Displays the contents of a file using the LESS application. | user-level EXEC and privileged-level EXEC | All |
| line | Specifies the terminal line settings. | global configuration | All |
| lls | Displays the files in a long list format. | user-level EXEC and privileged-level EXEC | All |
| logging | Configures system logging (syslog). | global configuration | All |
| ls | Lists the files and subdirectories in a directory. | user-level EXEC and privileged-level EXEC | All |
| mediafs-division | Configures the media file system space allocation for the WMT and RealProxy cache. | global configuration | Content Engine |
| mkdir | Makes a directory. | user-level EXEC and privileged-level EXEC | All |
| mkfile | Makes a file (for testing). | user-level EXEC and privileged-level EXEC | All |
| mode | Sets the Fibre Channel interface operation mode. | interface configuration | All |
| mtu | Sets the interface maximum transmission unit packet size. | interface configuration | All |
| multicast | Configures the multicast client license and delay timing options. | global configuration | Content Engine |
| multicast connectivity-test | Generates the multicast packets and tests connectivity through multicast routers. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| network-filesystem client (EXEC) | Configures the NAS share to preempt ownership to this Content Engine in error conditions. | user-level EXEC and privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| network-filesystem client (global configuration) | Extends the Content Engine storage to remote disk drives with the Common Internet File System (CIFS) or Network File System (NFS) protocols. | global configuration | Content Engine |
| network-filesystem server | Enables the use of a Windows file server with the Content Engine. | global configuration | Content Engine |
| no (global configuration) | Negates a global configuration command or sets its defaults. | global configuration | All |
| no (interface configuration) | Negates an interface command or sets its defaults. | interface configuration | All |
| ntlm | Configures the NTLM NT server parameters. | global configuration | Content Engine |
| ntp | Configures the Network Time Protocol server. | global configuration | All |
| ntpdate | Sets the NTP server name. | privileged-level EXEC | All |
| offline-operation | Enables offline operation if external network links are disrupted. | global configuration | Content Engine, Content Router |
| pgmrategen | Starts the pgmrategen application. | user-level EXEC and privileged-level EXEC | Content Engine |
| pgmratemon | Starts the pgmratemon application. | user-level EXEC and privileged-level EXEC | Content Engine |
| ping | Sends the echo packets. | user-level EXEC and privileged-level EXEC | All |
| port-channel | Configures the port-channel load-balancing options. | global configuration | All |
| pre-load | Configures the Content Engine to fetch and preload content. | global configuration | Content Engine |
| pre-load force | Forces a preload operation. | privileged-level EXEC | All |
| primary-interface | Configures a primary interface for the ACNS network to be a Fast Ethernet, Gigabit Ethernet, or port-channel interface. | global configuration | All |
| proxy-auto-config (EXEC) | Downloads the proxy automatic configuration file from an FTP server. | privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| proxy-auto-config (global configuration) | Enables the browser automatic configuration feature. | global configuration | Content Engine |
| proxy-protocols | Configures the proxy protocols-related parameters. | global configuration | Content Engine |
| pwd | Displays the present working directory. | user-level EXEC and privileged-level EXEC | All |
| radius-server | Configures the RADIUS authentication. | global configuration | All |
| reload | Halts a device and performs a cold restart. | privileged-level EXEC | All |
| rename | Renames a file. | user-level EXEC and privileged-level EXEC | All |
| restore | Restores a device to its manufactured default status. | privileged-level EXEC | All |
| rmdir | Removes a directory. | user-level EXEC and privileged-level EXEC | All |
| rtsp (EXEC) | Restores RealProxy or RealSubscriber to its default configuration. | privileged-level EXEC | All |
| rtsp (global configuration) | Configures the Real-Time Streaming Protocol-related parameters. | global configuration | Content Engine, Content Router |
| rule | Sets the rules by which the Content Engine filters HTTP, HTTPS, and RTSP traffic. | global configuration | Content Engine |
| script | Checks the errors in a script or executes a script. | privileged-level EXEC | All |
| setup | Configures the basic configuration settings and a set of commonly used caching services. | privileged-level EXEC | All |
| show | Displays the running system information. | user-level EXEC and privileged-level EXEC | All |
| show aaa accounting | Displays the AAA accounting configuration. | privileged-level EXEC | All |
| show access-lists | Displays the access control list configuration. | user-level EXEC and privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show acquirer | Displays the acquirer channel information and progress for a specified channel number or name. | user-level EXEC and privileged-level EXEC | Content Engine |
| show alarms | Displays information on various types of alarms, their status, and history. | privileged-level EXEC | All |
| show arp | Displays the Address Resolution Protocol entries. | user-level EXEC and privileged-level EXEC | All |
| show authentication | Displays the authentication configuration. | user-level EXEC and privileged-level EXEC | All |
| show auto-register | Displays the automatic registration status of a Content Engine or Content Router. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| show bandwidth | Displays the bandwidth allocated to a particular device. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| show banner | Displays information on various types of banners. | user-level EXEC and privileged-level EXEC | All |
| show bitrate | Displays the Content Engine bit-rate configuration. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| show bypass | Displays the Content Engine bypass configuration. | user-level EXEC and privileged-level EXEC | Content Engine |
| show cdnfs | Displays the ACNS network file system information. | user-level EXEC and privileged-level EXEC | Content Engine, Content Distribution Manager |
| show cdn-statistics | Displays the statistical data about Content Engines and device groups. | user-level EXEC and privileged-level EXEC | Content Distribution Manager |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show cdp | Displays the Cisco Discovery Protocol configuration. | user-level EXEC and privileged-level EXEC | All |
| show cfs | Displays the cache file system status. | user-level EXEC and privileged-level EXEC | Content Engine, Content Distribution Manager |
| show clock | Displays the system clock. | user-level EXEC and privileged-level EXEC | All |
| show cms | Displays the Centralized Management System protocol, embedded database content, maintenance status, and other information. | user-level EXEC and privileged-level EXEC | All |
| show content-routing | Displays the Content Router simplified hybrid routing table. | user-level EXEC and privileged-level EXEC | Content Router |
| show content-routing | Displays the state of each debugging option. | user-level EXEC and privileged-level EXEC | All |
| show device-mode | Displays the configured or current mode of a Content Distribution Manager, Content Engine, or Content Router device. | user-level EXEC and privileged-level EXEC | All |
| show disks | Displays the disk configurations. | user-level EXEC and privileged-level EXEC | All |
| show distribution | Displays the distribution information for a specified channel. | user-level EXEC and privileged-level EXEC | Content Engine |
| show dns | Displays the DNS cache status and the memory allocated to cache use. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| show error-handling | Displays the error-handling configuration. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show flash | Displays the flash memory information. | user-level EXEC and privileged-level EXEC | All |
| show ftp-native | Displays the FTP native proxy configuration. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| show ftp-over-http | Displays the FTP-over-HTTP caching-related configuration. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| show gui-server | Displays the Content Engine graphical user interface (GUI) server configuration. | user-level EXEC and privileged-level EXEC | Content Engine, Content Distribution Manager |
| show hardware | Displays the system hardware information. | user-level EXEC and privileged-level EXEC | All |
| show hosts | Displays the IP domain name, name servers, IP addresses, and host table. | user-level EXEC and privileged-level EXEC | All |
| show http | Displays the HTTP-related caching configuration. | user-level EXEC and privileged-level EXEC | Content Engine |
| show http-authcache | Displays the authentication cache. | user-level EXEC and privileged-level EXEC | Content Engine |
| show https | Displays the HTTPS-related parameters. | user-level EXEC and privileged-level EXEC | Content Engine |
| show icap | Displays the ICAP configurations. | user-level EXEC and privileged-level EXEC | Content Engine |
| show icp | Displays the Internet Cache Protocol information. | user-level EXEC and privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show inetd | Displays the status of TCP/IP services. | user-level EXEC and privileged-level EXEC | All |
| show interface | Displays the hardware interface information. | user-level EXEC and privileged-level EXEC | All |
| show inventory | Displays the system inventory information. | user-level EXEC and privileged-level EXEC | All |
| show ip access-list | Displays the information about access lists that are defined and applied to specific interfaces or applications. | user-level EXEC and privileged-level EXEC | Content Engine |
| show ip routes | Displays the IP routing table. | user-level EXEC and privileged-level EXEC | All |
| show ldap | Displays the LDAP parameters. | user-level EXEC and privileged-level EXEC | Content Engine |
| show logging | Displays the system logging configuration. | user-level EXEC and privileged-level EXEC | All |
| show mediafs | Displays the media file system (mediafs) information. | user-level EXEC and privileged-level EXEC | Content Engine |
| show memory | Displays the memory blocks and statistics. | user-level EXEC and privileged-level EXEC | All |
| show multicast | Displays the multicast configuration and license parameters. | user-level EXEC and privileged-level EXEC | Content Engine |
| show network-filesystem | Displays the status of network-attached storage (NAS) devices or file servers. | user-level EXEC and privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show ntlm | Displays the NTLM parameters. | user-level EXEC and privileged-level EXEC | Content Engine |
| show ntp | Displays the Network Time Protocol configuration status. | user-level EXEC and privileged-level EXEC | All |
| show pac-file-server | Displays the information regarding the dynamic proxy autoconfig file server. | user-level EXEC and privileged-level EXEC | Content Engine |
| show pre-load | Displays the preload configuration. | user-level EXEC and privileged-level EXEC | Content Engine |
| show processes | Displays the process status. | user-level EXEC and privileged-level EXEC | All |
| show programs | Displays the scheduled programs. | user-level EXEC and privileged-level EXEC | Content Engine |
| show proxy-auto-config | Displays the state of the browser automatic configuration feature. | user-level EXEC and privileged-level EXEC | Content Engine |
| show proxy-protocols | Displays the proxy protocols' parameters. | user-level EXEC and privileged-level EXEC | Content Engine |
| show radius-server | Displays the RADIUS server information. | user-level EXEC and privileged-level EXEC | All |
| show rtsp | Displays the RTSP configurations. | user-level EXEC and privileged-level EXEC | Content Engine |
| show rule | Displays the Rules Template configuration information. | user-level EXEC and privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show running-config | Displays the current operating configuration. | user-level EXEC and privileged-level EXEC | All |
| show services | Displays the services-related information. | user-level EXEC and privileged-level EXEC | All |
| show snmp | Displays the SNMP parameters. | user-level EXEC and privileged-level EXEC | All |
| show ssh | Displays the Secure Shell status and configuration. | user-level EXEC and privileged-level EXEC | All |
| show standby | Displays the information related to the standby interface. | user-level EXEC and privileged-level EXEC | All |
| show startup-config | Displays the startup configuration. | user-level EXEC and privileged-level EXEC | All |
| show statistics access-lists 300 | Displays the access control list statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics acquirer | Displays the Content Engine acquirer channel statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics authentication | Displays the authentication statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics bandwidth | Displays the Content Engine bandwidth statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics bypass | Displays the Content Engine bypass statistics. | user-level EXEC and privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show statistics cdnfs | Displays the Content Engine ACNS network file system statistics. | user-level EXEC and privileged-level EXEC | Content Engine, Content Distribution Manager |
| show statistics cfs | Displays the cache file system statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics cifs-server | Displays the CIFS server statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics content-distribution-network | Displays the status of the Content Engines or device groups for an ACNS network. | privileged-level EXEC | Content Distribution Manager |
| show statistics content-routing | Displays the simplified hybrid content routing statistics. | user-level EXEC and privileged-level EXEC | Content Router |
| show statistics distribution | Displays the simplified statistics for content distribution components. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics dns-cache | Displays the DNS caching statistics. | user-level EXEC and privileged-level EXEC | Content Router |
| show statistics ftp-native | Displays the statistics for the FTP native requests. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics ftp-over-http | Displays the statistics for the FTP-over-HTTP requests. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics http | Displays the Hypertext Transfer Protocol statistics. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| show statistics http-authcache | Displays the HTTP cache authentication statistics. | user-level EXEC and privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show statistics https | Displays the HTTPS statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics icap | Display the ICAP-related statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics icmp | Displays the Internet Control Message Protocol statistics. | user-level EXEC and privileged-level EXEC | All |
| show statistics icp | Displays the Internet Cache Protocol statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics ip | Displays the Internet Protocol statistics. | user-level EXEC and privileged-level EXEC | All |
| show statistics ldap | Displays the Lightweight Directory Access Protocol statistics. | user-level EXEC and privileged-level EXEC | Content Engine, Content Distribution Manager |
| show statistics mediafs | Displays the media file system statistics. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| show statistics netstat | Displays the Internet socket connection statistics. | user-level EXEC and privileged-level EXEC | All |
| show statistics ntlm | Displays the Windows NT LAN Manager statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics pac-file-server | Displays the statistics for the dynamic proxy autoconfig file server. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics pre-load | Displays the preloaded URL list statistics. | user-level EXEC and privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show statistics radius | Displays the RADIUS authentication statistics. | user-level EXEC and privileged-level EXEC | All |
| show statistics replication | Displays the channel replication status and related statistical data. | user-level EXEC and privileged-level EXEC | Content Engine, Content Distribution Manager |
| show statistics rtsp | Displays the Real-Time Streaming Protocol statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics rule | Displays the rule statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics services | Displays the services statistics. | user-level EXEC and privileged-level EXEC | All |
| show statistics snmp | Displays the SNMP statistics. | user-level EXEC and privileged-level EXEC | All |
| show statistics tacacs | Displays the TACACS+ authentication and authorization statistics. | user-level EXEC and privileged-level EXEC | All |
| show statistics tcp | Displays the Transmission Control Protocol statistics. | user-level EXEC and privileged-level EXEC | All |
| show statistics transaction-logs | Displays the transaction log export statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics tvout | Displays the Content Engine TV-out statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics udp | Displays the User Datagram Protocol statistics. | user-level EXEC and privileged-level EXEC | All |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show statistics url-filter | Displays the URL filtering statistics for HTTP, RTSP, and WMT. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics wccp | Displays the WCCP statistics for the Content Engine. | user-level EXEC and privileged-level EXEC | Content Engine |
| show statistics wmt | Displays the Windows Media Technologies statistics. | user-level EXEC and privileged-level EXEC | Content Engine |
| show sysfs | Displays the system file system information. | user-level EXEC and privileged-level EXEC | All |
| show tacacs | Displays the TACACS+ configuration. | user-level EXEC and privileged-level EXEC | All |
| show tcp | Displays the TCP configuration. | user-level EXEC and privileged-level EXEC | All |
| show tech-support | Displays the system information for Cisco technical support. | user-level EXEC and privileged-level EXEC | All |
| show telnet | Displays the Telnet services configuration. | user-level EXEC and privileged-level EXEC | All |
| show tftp-server | Displays the Trivial File Transfer Protocol (TFTP) server configuration. | user-level EXEC and privileged-level EXEC | All |
| show transaction-logging | Displays the transaction logging information. | user-level EXEC and privileged-level EXEC | Content Engine |
| show tvout | Displays the TV-out information. | user-level EXEC and privileged-level EXEC | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| show url-filter | Displays the URL filter configurations. | user-level EXEC and privileged-level EXEC | Content Engine |
| show user | Displays the user identification number and username information. | user-level EXEC and privileged-level EXEC | All |
| show users | Displays the specified users. | user-level EXEC and privileged-level EXEC | All |
| show version | Displays the software version. | user-level EXEC and privileged-level EXEC | All |
| show wccp | Displays the WCCP information. | user-level EXEC and privileged-level EXEC | Content Engine |
| show websense-server | Displays the URL filtering statistics for the local Websense server. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| show wmt | Displays the WMT configuration. | user-level EXEC and privileged-level EXEC | Content Engine |
| shutdown (interface configuration) | Shuts down the specified interface. | interface configuration | All |
| shutdown (EXEC) | Shuts down the device (stops all applications and operating system). | privileged-level EXEC | All |
| snmp-server access-list | Configures an access control list to allow access through an SNMP agent. | global configuration | All |
| snmp-server access-list | Enables SNMP; sets the community string and optionally names the group and enables the read-write access with the community string. | global configuration | All |
| snmp-server contact | Specifies the text for the MIB object sysContact. | global configuration | All |
| snmp-server enable traps | Enables the SNMP traps. | global configuration | All |
| snmp-server group | Defines a user security model group. | global configuration | All |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| snmp-server host | Specifies the hosts to receive SNMP traps. | global configuration | All |
| snmp-server location | Specifies the path for MIB object sysLocation. | global configuration | All |
| snmp-server mib | Configures the persistence for the SNMP Event MIB. | global configuration | All |
| snmp-server notify inform | Configures the SNMP inform request. | global configuration | All |
| snmp-server user | Defines a user who can access the SNMP engine. | global configuration | All |
| snmp-server view | Defines a Version 2 SNMP (SNMPv2) MIB view. | global configuration | All |
| speed | Sets the Fibre Channel interface speed. | interface configuration | All |
| sshd | Configures the SSH service parameters. | global configuration | All |
| ssh-key-generate | Generates a Secure Shell (SSH) host key. | global configuration | All |
| standby | Configures an interface to be a backup for another interface. | interface configuration | All |
| tacacs | Enables and configures the TACACS+ authentication parameters. | global configuration | All |
| tcp | Configures the TCP parameters. | global configuration | All |
| tcpdump | Dumps the TCP traffic on the network. | privileged-level EXEC | All |
| telnet | Starts the Telnet client. | user-level EXEC and privileged-level EXEC | All |
| telnet enable | Enables the Telnet services. | global configuration | All |
| terminal | Sets the terminal output commands. | user-level EXEC and privileged-level EXEC | All |
| test-url | Tests the accessibility of a URL using the FTP, HTTP, or HTTPS protocol. | user-level EXEC and privileged-level EXEC | Content Engine, Content Router |
| tftp-server | Sets the Trivial File Transfer Protocol server directory. | global configuration | All |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| traceroute | Traces the route to a remote host. | user-level EXEC and privileged-level EXEC | All |
| transaction-log force | Forces archiving of the working log file to make a transaction log file. | privileged-level EXEC | All |
| transaction-logs | Configures and enables the transaction logging parameters. | global configuration | Content Engine |
| tvout | Enables and configures TV-out service. | global configuration | Content Engine |
| type | Displays a file. | user-level EXEC and privileged-level EXEC | All |
| type-tail | Displays the last several lines of a file. | user-level EXEC and privileged-level EXEC | All |
| undebug | Disables the debugging functions (see also **debug**). | privileged-level EXEC | All |
| url-filter (EXEC) | Reloads the new local good sites or good sites' lists for HTTP, RTSP, or WMT when URL filtering is enabled. | privileged-level EXEC | All |
| url-filter (global configuration) | Configures and enables URL filtering over HTTP, RTSP, or WMT. | global configuration | Content Engine |
| username | Establishes the username authentication. | global configuration | All |
| wccp access-list | Configures the IP access list for inbound WCCP GRE-encapsulated traffic. | global configuration | Content Engine |
| wccp custom-web-cache | Configures the custom web caching service. | global configuration | Content Engine |
| wccp dns | Enables the interception and redirection of DNS packets to a boomerang server. | global configuration | Content Engine, Content Router |
| wccp flow-redirect enable | Enables the WCCP flow redirection. | global configuration | Content Engine |
| wccp ftp-native | Enables or disables the transparent interception of FTP native traffic with WCCP Version 2. | global configuration | Content Engine, Content Router |
| wccp home-router | Specifies a WCCP Version 1 home router IP address. | global configuration | Content Engine |

*Table 2-1 CLI Commands (continued)*

| Command | Description | CLI Mode | Device Mode |
|---|---|---|---|
| wccp https-cache | Enables the WCCP flow redirection to a Content Engine configured as an HTTPS server. | global configuration | Content Engine, Content Router |
| wccp port-list | Associates the ports with specific WCCP Version 2 dynamic services. | global configuration | Content Engine |
| wccp reverse-proxy | Configures the WCCP Version 2 reverse proxy web caching service. | global configuration | Content Engine |
| wccp router-list | Creates a router list for use in the WCCP Version 2 services. | global configuration | Content Engine |
| wccp rtsp | Configures the WCCP Version 2 RTSP protocol transparent interception. | global configuration | Content Engine |
| wccp service-number | Enables the WCCP Version 2 redirection services. | global configuration | Content Engine |
| wccp shutdown | Sets the maximum time interval after which the Content Engine will perform a clean shutdown. | global configuration | Content Engine |
| wccp slow-start enable | Enables the slow-start capability. | global configuration | Content Engine |
| wccp spoof-client-ip enable | Uses the client IP address while connecting to the origin server. | global configuration | Content Engine |
| wccp version | Specifies the WCCP version number. | global configuration | Content Engine |
| wccp web-cache | Configures the standard web cache service. | global configuration | Content Engine |
| wccp wmt | Configures the web cache service to run with WCCP and Windows Media Technologies (WMT). | global configuration | Content Engine |
| wccp wmt-rtspu | Configures the WMT RTP transparent redirection. | global configuration | Content Engine |
| websense-server | Enables the use of a Websense HTTP URL filtering plug-in in a Content Engine. | global configuration | Content Engine, Content Router |
| whoami | Displays the current user's name. | EXEC both | All |
| wmt (EXEC) | Starts and stops the named WMT multicast stations. | privileged-level EXEC | All |
| wmt (global configuration) | Configures the WMT. | global configuration | Content Engine |
| write | Writes or erases the startup configurations to NVRAM or to a terminal session, or writes the MIB persistence configuration to disk. | privileged-level EXEC | All |

[1] Commands used to access configuration modes.

# aaa accounting

To configure authentication, authorization, and accounting (AAA), use the **aaa accounting** global configuration command. To remove individual options, use the **no** form of this command.

**aaa accounting** {**commands** {**0** | **15**} **default** {**start-stop** | **stop-only** | **wait-start**} **tacacs** | **exec default** {**start-stop** | **stop-only** | **wait-start**} **tacacs** | **system default** {**start-stop** | **stop-only**} **tacacs**}

**no aaa accounting** {**commands** {**0** | **15**} **default** {**start-stop** | **stop-only** | **wait-start**} **tacacs** | **exec default** {**start-stop** | **stop-only** | **wait-start**} **tacacs** | **system default** {**start-stop** | **stop-only**} **tacacs**}

## Syntax Description

| | |
|---|---|
| **commands** | Configures accounting for all commands at the specified privilege level. |
| **0** | Specifies the user privilege level for a normal user. |
| **15** | Specifies the user privilege level for an administrative user. |
| **default** | Sets AAA accounting to use the default accounting list. |
| **start-stop** | Sends a start accounting notice at the beginning of a process and a stop accounting notice at the end of a process. The start accounting record is sent in the background. The requested user process begins regardless of whether the start accounting notice was received by the accounting server. |
| **stop-only** | Sends a stop accounting notice at the end of the process requested by the user. |
| **wait-start** | Sends both a start and a stop accounting notice to the accounting server. However, the requested user service does not begin until the start accounting notice is acknowledged. The user cannot execute a CLI command or login until the user is on record.<br>A stop accounting notice is also sent but does not need acknowledgement. |
| **tacacs** | Enables use of TACACS+ for accounting. |
| **exec** | Enables accounting for user EXEC processes (user shells). When enabled, the EXEC shell accounting reports EXEC terminal session (user shell) events and administrator's login and logout to the EXEC shell. |
| **system** | Enables accounting for all system-level events not associated with users, such as reloads. |

## Defaults

AAA accounting is disabled by default.

## Command Modes

global configuration

## Usage Guidelines

The AAA accounting feature enables you to track the activities of an administrative user, services that users access, and the amount of network resources they consume (for example, connection time or the bytes transferred). You can use the AAA accounting feature to track user activity for billing, auditing, reporting, or security purposes. The ACNS 5.2 software and later releases use TACACS+ to implement

AAA accounting; RADIUS is not currently supported. When AAA accounting is enabled, the Content Engine reports user activity to the TACACS+ security server in the form of accounting records. This data can then be analyzed for network management, client billing, and auditing.

In the ACNS 5.2 software and later releases, you can activate accounting for the following types of events:

- EXEC—EXEC shell accounting is used to report the events of an administrator logging in and out of the EXEC shell through Telnet, FTP, or SSH (SSH Version 1 or Version 2). This type of accounting records information about user EXEC terminal sessions (user shells) on the Content Engine, including username, date, start and stop times for each session, time zone, and IP address of the system used to access the Content Engine. The EXEC shell accounting information can be accessed through the TACACS+ server's accounting log file. This log file uses the following report format for this type of accounting information:

  ```
  WeekDay#Month#Day#Time#Year#CEaddress#username#terminal#RemoteHost#Event#
  EventTime#TaskId#Timezone#Service
  ```

- Command—The Content Engine records information about the CLI commands that were executed on the Content Engine. Each command accounting record includes the executed command syntax, username of the user who executed the command, the user's privilege level, and the date and time that each command was executed. The Content Engine supports two privilege levels, 0 and 15, representing normal users and administrative users, respectively. The command accounting information can be accessed through the TACACS+ server's accounting log file. This log file uses the following report format for this type of accounting information:

  ```
  WeekDay#Month#Day#Time#Year#CEaddress#username#terminal#RemoteHost#Event#
  EventTime#TaskId#Timezone#Service#PrivilegeLevel#CLICommand
  ```

- System—The Content Engine records information about all system-level events (for example, when the system reboots). You can access the system accounting information through the TACACS+ server's accounting log file. This log file uses the following report format for this type of accounting information:

  ```
  WeekDay#Month#Day#Time#Year#CEaddress#username#terminal#RemoteHost#Event#
  EventTime#TaskId#Timezone#SystemService#SystemAccountingEvent#EventReason
  ```

The ACNS 5.2 software and later releases support only the default accounting list.

**Caution** The ACNS software displays the following warning message if the wait-start option is configured:

```
Warning: The device may become non-responsive if it cannot contact a configured TACACS+
server.
```

The administrator is asked to confirm the configuration in an indefinite loop until the administrator enters "yes" to the following prompt:

```
Are you sure you want to proceed? [yes]
```

## Examples

The following example configures TACACS+ on the Content Engine and also specifies that a start accounting notice should be sent at the beginning of the process and a stop accounting notice at the end of the process, and the requested user process should begin regardless of whether the start accounting notice was received by the accounting server:

```
ContentEngine(config)# tacacs key abc
```

```
ContentEngine(config)# tacacs server 172.16.50.1 primary

ContentEngine(config)# aaa accounting system default start-stop tacacs

ContentEngine# show aaa accounting
Accounting Type   Record event(s)  Protocol
-----------------------------------------------------------------
Exec shell         unknown           unknown
Command level  0   unknown           unknown
Command level 15   unknown           unknown
System             start-stop        TACACS+
```

The following example shows that the Content Engine is set to record all user EXEC sessions. The command also specifies that a stop accounting notice should be sent to the TACACS+ server at the end of the session.

```
ContentEngine(config)# aaa accounting exec default stop-only tacacs
```

The following example shows that the Content Engine is set to record all CLI commands executed by a normal user. The command also specifies that a stop accounting notice should be sent to the TACACS+ server at the end of each CLI command executed by a normal user.

```
ContentEngine(config)# aaa accounting commands 0 default stop-only tacacs
```

The following example shows that the Content Engine is set to record all CLI commands executed by an administrative user. The command also specifies that a start accounting notice should be sent to the TACACS+ server at the beginning of the process and a stop accounting notice at the end of the process. The CLI command executed by the administrative user does not proceed until the start accounting notice has been acknowledged.

```
ContentEngine(config)# aaa accounting commands 15 default wait-start tacacs
```

The following example shows the EXEC shell accounting report that is available on the TACACS+ server:

```
Wed Apr 14 11:19:19 2004 172.16.0.0 super10 pts/0 172.31.0.0 start
start_time=1081919558 task_id=3028 timezone=PST service=shell
Wed Apr 14 11:19:23 2004 172.16.0.0 super10 pts/0 172.31.0.0
stop stop_time=1081919562 task_id=3028 timezone=PST service=shell
Wed Apr 14 11:22:13 2004 172.16.0.0 normal20 pts/0 via5.abc.com start
start_time=1081919732 task_id=3048 timezone=PST service=shell
Wed Apr 14 11:22:16 2004 172.16.0.0 normal20 pts/0 via5.abc.com stop
stop_time=1081919735 task_id=3048 timezone=PST service=shell
Wed Apr 14 11:25:29 2004 172.16.0.0 admin ftp via5.abc.com start start_time=1081919928
task_id=3069 timezone=PST service=shell
Wed Apr 14 11:25:33 2004 172.16.0.0 admin ftp via5.abc.com stop stop_time=1081919931
task_id=3069 timezone=PST service=shell
```

The following example shows the system accounting report that is available on the TACACS+ server:

```
Wed Apr 14 08:37:14 2004 172.16.0.0 unknown unknown 0.0.0.0 start start_time=1081909831
task_id=2725 timezone=PST service=system event=sys_acct reason=reload
Wed Apr 14 10:19:18 2004 172.16.0.0 admin ttyS0 0.0.0.0 stop stop_time=1081915955
task_id=5358 timezone=PST service=system event=sys_acct reason=shutdown
```

The following example shows the command accounting report that is available on the TACACS+ server:

```
Wed Apr 14 12:35:38 2004 172.16.0.0 admin ttyS0 0.0.0.0 start start_time=1081924137
task_id=3511 timezone=PST service=shell -lvl=0 cmd=logging console enable
Wed Apr 14 12:35:39 2004 172.16.0.0 admin ttyS0 0.0.0.0 stop stop_time=1081924137
task_id=3511   timezone=PST service=shell priv-lvl=0 cmd=logging console enable
```

In addition to command accounting, the Content Engine records any executed CLI command in the system log (syslog). The message format is as follows:

```
ce_syslog(LOG_INFO, CESM_PARSER, PARSER_ALL, CESM_350232,
          "CLI_LOG %s: %s \n", __FUNCTION__, pd->command_line);
```

## Related Commands

**debug aaa accounting**
**show aaa accounting**

# access-lists

To configure access control list entries, use the **access-lists** global configuration command. To remove access control list entries, use the **no** form of this command.

**access-lists** {**300** {**deny groupname** {**any** [**position** *number*] | *groupname* [**position** *number*]}} |
{**permit groupname** {**any** [**position** *number*] | *groupname* [**position** *number*]}} | **enable**}

**no access-lists** {**300** {**deny groupname** {**any** [**position** *number*] | *groupname* [**position** *number*]}}|
{**permit groupname** {**any** [**position** *number*] | *groupname* [**position** *number*]}} | **enable**}

## Syntax Description

| | |
|---|---|
| **300** | Specifies the group name-based access control list (ACL). |
| **deny** | Specifies the rejection action. |
| **groupname** | Defines which groups are granted or denied access to content that is served by this Content Engine. |
| **any** | Specifies any group name. |
| **position** | (Optional) Specifies the position of the access control list record within the access list. |
| *number* | Position number within the access control list (1–4294967294). |
| *groupname* | Name of the group that is permitted or denied from accessing the Internet using a Content Engine. |
| **permit** | Specifies the permission action. |
| **enable** | Enables the access control list. |

## Defaults

No default behaviors or values

## Command Modes

global configuration

## Usage Guidelines

In the ACNS 5.x software, you can configure group authorization using an access control list (ACL) only after a user has been authenticated against an NTLM or LDAP HTTP-request authentication server. The use of this list configures a group privilege when members of the group are accessing content provided by the Content Engine. You can use the ACL to allow the users who belong to certain groups or to prevent them from viewing specific content. This authorization feature offers more granular access control by specifying that access is only allowed to specific groups.

Use the **access-lists enable** global configuration command to enable the use of the ACL.

Use the **access-lists 300** command to permit or deny a group from accessing the Internet using the Content Engine. For instance, use the **access-lists 300 deny groupname marketing** command to prevent any user from the marketing group from accessing content through the Content Engine.

At least one login authentication method, such as local, TACACS+, or RADIUS, must be enabled.

**Note** We recommend that you configure the local login authentication method as the primary method.

In the ACNS 5.x software, the access control list contains the following feature enhancements and limitations:

- A user can belong to several groups.
- A user can belong to an unlimited number of groups within group name strings.
- A group name string is a case-sensitive string with mixed-case alphanumeric characteristics.
- Each unique group name string cannot exceed 128 characters.

**Note** If the unique group name string is longer than 128 characters, the group is ignored.

- The group names in a group name string are separated by a comma.
- The total string of individual group names cannot exceed 750 characters.

For Windows-based user groups, you must append the domain name in front of the group name in the form domain\group as follows:

- For Windows NT-based user groups, use the domain NetBIOS name.
- For Windows 2000-based user groups, use the domain DNS name.

### Examples

The following example shows how to display the configuration of the access control list by using the **show access-lists 300** command:

```
ContentEngine# show access-lists 300
Access Control List Configuration
  ---------------------------------
    Access Control List is enabled

    Groupname-based List (300)
    1.  permit  groupname techpubs
    2.  permit  groupname acme1
    3.  permit  groupname engineering
    4.  permit  groupname sales
    5.  permit  groupname marketing
    6.  deny groupname any
```

The following example shows how to display statistical information for the access control list by using the **show statistics access-lists 300** command:

```
ContentEngine# show statistics access-lists 300
    Access Control Lists Statistics
    -----------------------------------------
      Groupname and username-based List (300)
        Number of requests:         1
        Number of deny responses:   0
        Number of permit responses: 1
```

The following example shows how to reset the statistical information for the access control list by using the **clear statistics access-lists 300** command:

```
ContentEngine# clear statistics access-lists 300
ContentEngine(config)# access-lists 300 permit groupname acme1 position 2
```

### Related Commands

**show access-lists 300**
**show statistics access-list 300**

# acquirer (EXEC)

To start or stop content acquisition on a specified acquirer channel, use the **acquirer** EXEC command. You can also use this command to verify and correct the Last-Modified-Time attribute in content acquired using the ACNS software before Release 5.0.3.

**acquirer** {**check-time-for-old-content** [**channel-id** *channel-num* | **channel-name** *channel-name*] [**correct** [**channel-id** *channel-num* | **channel-name** *channel-name*]] | **start-channel** {**channel-id** *channel-num* | **channel-name** *channel-name*} | **stop-channel** {**channel-id** *channel-num* | **channel-name** *channel-name*} | **test-url** *url* [**use-http-proxy** *url* | **use-ntlm-domain** *domain-name* | **use-smb-options** *smb-options*]}

## Syntax Description

| | |
|---|---|
| **check-time-for-old-content** | Checks the content for Last-Modified-Time attributes in the local time format. |
| **channel-id** | (Optional) Sets the channel number identifier. |
| *channel-num* | (Optional) Channel number (0–4294967295). |
| **channel-name** | (Optional) Sets the channel name descriptor. |
| *channel-name* | (Optional) Channel name. |
| **correct** | (Optional) Changes the Last-Modified-Time attributes in the local time format to the Greenwich mean time (GMT) format. |
| **start-channel** | Starts the content acquisition. |
| **stop-channel** | Stops the content acquisition. |
| **test-url** | Tests the accessibility of a URL, using HTTP, HTTPS, FTP, or SMB. |
| *url* | URL to be tested.<br>**Note** For the Server Message Block (SMB) protocol, use the uniform naming convention (UNC) path, for example, //host/share/file. |
| **use-http-proxy** | (Optional) Specifies the HTTP proxy. The connectivity of the URL (content request over HTTP) through the HTTP proxy server (the Content Engine) is tested. Use this option only when the HTTP protocol is used. |
| *url* | HTTP proxy URL. Use one of the following formats to specify the HTTP proxy URL:<br>http://*proxyIpAddress*:*proxyPort*<br>http://*proxyUser*:*proxypasswd*@*proxyIpAddress*:*proxyPort* |
| **use-ntlm-domain** | (Optional) Specifies the domain to be used for NTLM authentication. |
| *domain-name* | NTLM domain name, for example, cisco.com. |
| **use-smb-options** | (Optional) Specifies the username, password, port, and domain for the SMB URL. |
| *smb-options* | Parameters to be specified when an SMB URL is used. Use the following format to specify these parameters:<br>username=xxx,password=xxx,port=xxx,workgroup=xxx<br>**Note** All the comma-separated key=value pairs are optional and need to be specified only if the SMB host requires them. |

## Defaults

If you do not specify the channel, this command applies to all channels assigned to the root Content Engine.

## Command Modes

EXEC

## Usage Guidelines

The acquirer is a software agent that gathers channel content before it is distributed to the receiver Content Engines in an ACNS network. The acquirer maintains a task list, which it updates after receiving a notification of changes in its channel configuration.

In the ACNS software Release 5.0.1 and earlier releases, the acquirer stored the Last-Modified-Time attribute in the local time format. As a result, content acquired using Release 5.0.1 or earlier software has a Last-Modified-Time attribute that is incorrect if used with later versions of the ACNS software, which use GMT format. Content downloaded after you upgrade to Release 5.0.3 and later releases has a Last-Modified-Time attribute in the correct GMT format.

When using Release 5.0.3 and later releases, you must correct the Last-Modified-Time attributes for content acquired with earlier releases by entering the following command from the privileged EXEC prompt:

**acquirer check-time-for-old-content correct** [**channel-id** *channel-num* **channel-name** *channel-name*]

This command changes the Last-Modified-Time attributes for content in all channels assigned to the root Content Engine unless you specify the channel ID or name.

Content Engines running ACNS software, Release 5.0.3 and later releases identify changes in the Last-Modified-Time attribute and download content only when changes have occurred.

Use the **acquirer start-channel** command to immediately start acquisition tasks for the selected channel. Use the **acquirer stop-channel** command to immediately stop all acquisition tasks for the selected channel.

Use the **acquirer test-url** *url* EXEC command to test whether a URL is accessible or not. The actual content is dumped into the path /dev/null.

## Examples

The following example shows how the acquirer starts acquiring content on channel 86:

```
ContentEngine# acquirer start-channel channel-id 86

ContentEngine# acquirer start-channel channel-name corporate
```

The following example shows how the acquirer stops acquiring content on channel 86:

```
ContentEngine# acquirer stop-channel channel-id 86

ContentEngine# acquirer stop-channel channel-name corporate
```

The following example shows how the **acquirer test-url** command is used to test a URL:

```
ContentEngine# acquirer test-url http://172.16.150.26
--05:16:41--  http://10.107.150.26
           => `/dev/null'
Connecting to 10.107.150.26:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1,722 [text/html]
```

```
100%[=====================================>] 1,722          1.64M/s    ETA 00:00

02:45:40 (1.64 MB/s) - `/dev/null' saved [1722/1722]
```

## Related Commands

**show acquirer**

**show statistics acquirer**

# acquirer (global configuration)

To provide authentication when the acquirer obtains content through a proxy server, use the **acquirer** global configuration command. To disable acquirer proxy authentication, use the **no** form of this command.

**acquirer proxy authentication** {**outgoing** {*hostname* | *ip-address*} *port-num* | **transparent**} *username* [**ntlm** *domain* [**basic-auth-disable**] | **password** *password* [**ntlm** *domain* [**basic-auth-disable**]]]

**no acquirer proxy authentication** {**outgoing** {*hostname* | *ip-address*} *port-num* | **transparent**} *username* [**ntlm** *domain* [**basic-auth-disable**] | **password** *password* [**ntlm** *domain* [**basic-auth-disable**]]]

## Syntax Description

| | |
|---|---|
| **proxy** | Configures parameters for outgoing proxy-mode requests for content acquisition. |
| **authentication** | Enables authentication so the acquirer can obtain content through a proxy server. |
| **outgoing** | Enables authentication for a nontransparent proxy server. |
| *hostname* | Hostname of a nontransparent proxy server. |
| *ip-address* | IP address of a nontransparent proxy server. |
| *port-num* | Port number of a nontransparent proxy server (1–65535). |
| **transparent** | Enables authentication for a transparent WCCP proxy server. |
| *username* | Username for authentication using a maximum of 256 characters. |
| **ntlm** | Allows authentication with NTLM. |
| *domain* | NTLM domain name for authentication using a maximum of 256 characters. |
| **basic-auth-disable** | (Optional) Prevents access through basic authentication. |
| **password** | Allows the use of a password for authentication. |
| *password* | Password for authentication using a maximum of 256 characters. |

## Defaults

No default behaviors or values

## Command Modes

global configuration

## Usage Guidelines

Use the **acquirer proxy authentication outgoing** global configuration command to configure authentication when you enable content acquisition through a proxy server. You must first configure the proxy host and the port using the **http proxy outgoing host** global configuration command. The maximum number of outgoing proxies allowed is eight. When you remove an outgoing proxy using the **no http outgoing proxy** command, the authentication information associated with that proxy is automatically removed.

Use the **acquirer proxy authentication transparent** command for transparent caches in the ACNS network that require authentication.

The acquirer supports a proxy with basic or NTLM authentication. Content acquisition through a proxy server is supported only for HTTP and not for HTTPS or FTP. Also, authentication is only supported for a single proxy server in a chain, so if multiple proxy servers in a chain require authentication, the request will fail.

Acquisition through a proxy server can be configured when the root Content Engine cannot directly access the origin server because the origin server is set up to allow access only by a specified proxy server. When a proxy server is configured for root Content Engine content acquisition, the acquirer contacts the proxy server instead of the origin server, and all requests to that origin server go through the proxy server.

**Note** Content acquisition through a proxy server is only supported for HTTP requests. It is not supported for HTTPS, FTP, MMS, or MMS-over-HTTP requests.

If a transparent WCCP proxy is used, you do not need to configure a proxy for content acquisition. In such cases, HTTP requests from the acquirer are redirected to the WCCP proxy by a router. However, when you do not have a router to redirect HTTP requests to the proxy server, you must configure the Content Engine to use the proxy server.

There are three ways to configure the proxy server: through the Content Distribution Manager GUI, through the Content Engine CLI, or through the manifest file. If you need to configure the Content Engine to use the proxy for both caching and pre-positioned content, use the CLI to configure the proxy. The CLI command is a global configuration command that configures the entire Content Engine to use the proxy. If only the acquirer portion of the Content Engine needs to use the proxy for acquiring the pre-positioned content, use the manifest file or specify the outgoing proxy. When you configure the proxy server in the manifest file, you are configuring the acquirer to use the proxy to fetch the content for a particular channel.

**Note** Proxy configurations in the manifest file take precedence over proxy configurations in the CLI. A *noProxy* attribute configuration in the manifest file takes precedence over the other proxy server configurations in the manifest file.

You can also configure a proxy for fetching the manifest file by using the Content Distribution Manager GUI (the Creating New Channel or Modifying Channel window). When you configure a proxy server in the Content Distribution Manager GUI, the proxy configuration is valid only for acquiring the manifest file itself and not for acquiring the channel content. Requests for the manifest file go through the proxy server, and requests for the content go directly to the origin server.

**Tip** Before configuring a proxy server, verify that the root Content Engine is able to ping the proxy server. To check whether the proxy server is accepting incoming HTTP traffic at the configured port, use the **acquirer test-url http://***proxyIP***:***proxyport* global configuration command in the root Content Engine CLI, where the URL in the command is the URL of the proxy server being tested. If the proxy is not servicing the configured port, this message displays “failed: Connection refused.”

## Examples

The following example shows the authentication configuration for a nontransparent proxy server with NTLM authentication:

```
ContentEngine# acquirer proxy authentication outgoing 192.168.1.1 8080 myname password
password ntlm mydomain basic-auth-disable
```

The following example shows the authentication configuration for a transparent proxy server with basic authentication:

```
ContentEngine# acquirer proxy authentication transparent 192.168.1.1 8080 myname
```

## Related Commands

**http proxy outgoing**
**show acquirer**

# acquisition-distribution

To start or stop the content acquisition and distribution process, use the **acquisition-distribution** EXEC command.

**acquisition-distribution** {**database-cleanup** {**start** | **stop**} | **start** | **stop**}

## Syntax Description

| | |
|---|---|
| **database-cleanup** | Cleans up the acquisition and distribution database to maintain consistency with the file system. |
| **start** | Starts the cleanup of the acquisition and distribution database. |
| **stop** | Stops the cleanup of the acquisition and distribution database. |
| **start** | Starts the acquisition and distribution process. |
| **stop** | Stops the acquisition and distribution process. |

## Defaults

No default behaviors or values

## Command Modes

EXEC

## Usage Guidelines

When you use the **acquisition-distribution database-cleanup** command, the acquisition and distribution database is checked to ensure that all pre-positioned content is available in cdnfs. If any pre-positioned content is found to be missing from cdnfs, the content is replicated to all Content Engines in the ACNS network. Root Content Engines assigned to a channel acquire the content directly from the origin server and replicate the content through the channel either by unicast or multicast transmission to other Content Engines in the channel. Receiver Content Engines obtain the content from forwarder Content Engines either by unicast or multicast. In the case of a disk00 failure when the database is stored on disk00 in an internal file system (/state), the recovery of the acquisition and distribution database is done automatically. You should run the acquisition and distribution database cleanup if a failure occurs or if you have to replace a disk drive other than disk00.

## Examples

The following example starts the acquisition and distribution database cleanup process:

```
ContentEngine# acquisition-distribution database-cleanup start
```

The following example starts the acquisition and distribution process:

```
ContentEngine# acquisition-distribution start
```

The following example stops the acquisition and distribution process:

```
ContentEngine# acquisition-distribution stop
```

## Related Commands

**cdnfs cleanup**
**show acquirer**
**show distribution**

# alarm overload-detect

To detect alarm overload situations, use the **alarm overload-detect** global configuration command. To disable alarm overload detection, use the **no** form of this command.

**alarm overload-detect** {**clear** *1-999* [**raise** *10-1000*] | **enable** | **raise** *10-1000* [**clear** *1-999*]}

**no alarm overload-detect** {**clear** *1-999* [**raise** *10-1000*] | **enable** | **raise** *10-1000* [**clear** *1-999*]}

## Syntax Description

| | |
|---|---|
| **clear** | Specifies the threshold below which the alarm overload state on the Content Engine is cleared and the SNMP traps and alarm notifications to the Centralized Management System (CMS) resume.<br>**Note** The **alarm overload-detect clear** value must be less than the **alarm overload-detect raise** value. |
| *1-999* | Number of alarms per second that ends an alarm overload condition. |
| **raise** | (Optional) Specifies the threshold at which the Content Engine enters an alarm overload state and SNMP traps and alarm notifications to CMS are suspended. |
| **enable** | (Optional) Enables the detection of alarm overload situations. |
| *10-1000* | Number of alarms per second that triggers an alarm overload. |

## Defaults

**raise**: 10 alarms per second

**clear**: 1 alarm per second

## Command Modes

global configuration

## Usage Guidelines

When multiple applications running on a Content Engine experience problems at the same time, numerous alarms are set off simultaneously, and the Content Engine may stop responding. In the ACNS 5.2 software and later releases, you can use the **alarm overload-detect** global configuration command to set an overload limit for the incoming alarms from the node health manager. If the number of alarms exceeds the maximum number of alarms allowed, the Content Engine enters an alarm overload state until the number of alarms drops down to the number defined in the **clear** option.

When the Content Engine is in the alarm overload state, the following events occur:

- An alarm overload notification is sent to SNMP and the CMS. The **clear** and **raise** values are also communicated to SNMP and the CMS.
- SNMP traps and CMS notifications for subsequent alarm raise and clear operations are suspended.
- An alarm overload clear notification is sent.
- The Content Engine remains in the alarm overload state until the rate of incoming alarms decreases to the **clear** value.

**Note** In the alarm overload state, applications continue to raise alarms and the alarms are recorded within the Content Engine. The **show alarms** and **show alarms history** EXEC commands will display all the alarms even in the alarm overload state.

## Examples

The following example enables the detection of alarm overload:

```
CONTENTENGINE(config)# alarm overload-detect enable
```

The following example sets the threshold for triggering the alarm overload at 100 alarms per second:

```
CONTENTENGINE(config)# alarm overload-detect raise 100
```

The following example sets the level for clearing the alarm overload at 10 alarms per second:

```
CONTENTENGINE(config)# alarm overload-detect clear 10
```

## Related Commands

**show alarms**
**show alarm status**

# asset

To set the tag name for the asset tag string, use the **asset** global configuration command. To remove the asset tag name, use the **no** form of this command.

**asset tag** *name*

**no asset tag** *name*

## Syntax Description

| | |
|---|---|
| **tag** | Sets the asset tag. |
| *name* | Asset tag name string. |

## Defaults

No default behaviors or values

## Command Modes

global configuration

## Examples

The following example shows how to configure a tag name for the asset tag string:

```
ContentEngine(config)# asset tag entitymib
```

# authentication

To specify authentication and authorization methods, use the **authentication** global configuration command. To selectively disable options, use the **no** form of this command.

**authentication** {**configuration** {**local** | **radius** | **tacacs**} **enable** [**primary** | **secondary** | **tertiary**] | **fail-over server-unreachable** | **login** {**local** | **radius** | **tacacs**} **enable** [**primary** | **secondary** | **tertiary**]}

**no authentication** {**configuration** {**local** | **radius** | **tacacs**} **enable** [**primary** | **secondary** | **tertiary**] | **fail-over server-unreachable** | **login** {**local** | **radius** | **tacacs**} **enable** [**primary** | **secondary** | **tertiary**]}

## Syntax Description

| | |
|---|---|
| **configuration** | Sets configuration authentication (authorization). |
| **local** | Selects the local database for authentication or authorization. |
| **radius** | Selects a RADIUS server for authentication or authorization. |
| **tacacs** | Selects a TACACS+ server for authentication or authorization. |
| **enable** | Enables the source of authorization information. |
| **primary** | (Optional) Sets the first authentication method used. |
| **secondary** | (Optional) Sets the second authentication method used. |
| **tertiary** | (Optional) Sets the third authentication method used. |
| **fail-over** | Sets the condition to use the next authentication scheme, when primary authentication fails. |
| **server-unreachable** | Specifies that a failover to the secondary authentication scheme should occur only if the primary authentication server is unreachable. |
| **login** | Selects the local method for authentication. |

## Defaults

The local authentication method is enabled by default.

## Command Modes

global configuration

## Usage Guidelines

Authentication, also referred to as login, is the act of verifying usernames and passwords. Authorization is the action of determining what a user is allowed to do. It permits or denies privileges for authenticated users in the network. For example, if you log in to a Content Engine with a superuser administrator account (for example, the predefined admin account), you have the highest level of access privileges and can perform any administrative task such as the following:

- Configure the standalone Content Engine.
- Obtain statistical information that the standalone Content Engine has collected.
- Reload the device.

Generally, authentication precedes authorization in a network.

The **authentication** command configures both the authentication and authorization methods that govern login and configuration access to the Content Engine. Login and configuration privileges can be maintained in three different databases in the ACNS 5.x software: the local database, TACACS+ database, and RADIUS database. If all databases are enabled, then all three databases are queried. If the user data cannot be found in the first database queried, then the second and third databases are queried.

When an administrator can log in to the Content Engine through the console or the Content Engine GUI, the Content Engine checks the specified authentication database to verify the user's username and password to process these administrative login requests and to determine the access rights that this particular administrator should be granted during this login session. When the Content Engine receives an administrative login request, the Content Engine can check its local database or a remote third-party database (the TACACS+ database or the RADIUS database) to verify the username with the password and to determine the access privileges of the administrator.

When defining or modifying the authentication configuration method for a Content Engine, follow these guidelines:

- You can use the **authentication** command to choose between using an external access server or the internal (local) Content Engine-based AAA system for user access management.
- You can configure any combination of these authentication and authorization methods to control access and set privileges on a Content Engine:
  - Local authentication and authorization
  - RADIUS authentication and authorization
  - TACACS+ authentication and authorization
- Authentication configuration applies to the following:
  - Console and Telnet connection attempts
  - Secure FTP (SFTP), SSH (SSH Version 1 and Version 2), and Websense server access
- If you configure a RADIUS or TACACS+ key on the Content Engine (the RADIUS and the TACACS+ client), make sure that you configure an identical key on the RADIUS or TACACS+ server.
- If you configure multiple RADIUS or TACACS+ servers, the first server configured is the primary server, and authentication requests are sent to this server first. You can also specify secondary and tertiary servers for authentication and authorization purposes.
- By default, the Content Engine uses the local database to authenticate and authorize administrative login requests. The Content Engine verifies whether all authentication databases are disabled and if so, sets the system to the default state. For information on this default state, see the "Default Administrative Login Authentication and Authorization Configuration" section on page 2-42.

The **authentication login** command determines whether the user has any level of permission to access the Content Engine. The **authentication configuration** command authorizes the user with privileged access (configuration access) to the Content Engine.

The **authentication login local** and the **authentication configuration local** commands use a local database for authentication and authorization.

The **authentication login tacacs** and **authentication configuration tacacs** commands use a remote TACACS+ server to determine the level of user access.

The TACACS+ database validates users before they gain access to a Content Engine. TACACS+ is derived from the United States Department of Defense (RFC 1492) and is used by Cisco Systems as an additional control of nonprivileged and privileged mode access. The ACNS 5.3 software release and later releases support TACACS+ only and not TACACS or Extended TACACS.

To configure TACACS+, use the **authentication** and **tacacs** commands. To enable TACACS+, use the **tacacs enable** command.

For more information on TACACS+ authentication, see the "tacacs" section on page -761.

The **authentication login radius** and **authentication configuration radius** commands use a remote RADIUS server to determine the level of user access.

By default, the local method is enabled, with TACACS+ and RADIUS both disabled for login and configuration. Whenever TACACS+ and RADIUS are disabled, local is automatically enabled. TACACS+, RADIUS, and local methods can be enabled at the same time. The **primary** option specifies the first method to attempt for both login and configuration; the **secondary** option specifies the method to use if the primary method fails. The **tertiary** option specifies the method to use if both primary and secondary methods fail. If all methods of an **authentication login** or **authentication configuration** command are configured as primary, or all as secondary or tertiary, local is attempted first, then TACACS+, and then RADIUS.

**Note** You must use the **tacacs** global configuration command to configure a TACACS+ server for the TACACS+ authentication and authorization method. For information about configuring a TACACS+ server, see the "Specifying TACACS+ Authentication and Authorization Settings" section on page 2-44.

You must use the **radius-server** global configuration command to configure a RADIUS server for the RADIUS authentication and authorization method. For information about configuring a RADIUS server, see the "Specifying RADIUS Authentication and Authorization Settings" section on page 2-43.

### Default Administrative Login Authentication and Authorization Configuration

By default, the Content Engine uses the local database to obtain login authentication and authorization privileges for administrative users.

**Note** Use the **authentication** global configuration command to configure the authentication methods that govern administrative login and configuration access to the Content Engine.

Table 2-2 lists the default configuration for administrative login authentication and authorization.

*Table 2-2 Default Configuration for Administrative Login Authentication and Authorization*

| Feature | Default Value |
|---|---|
| Administrative login authentication | Enabled |
| Administrative configuration authorization | Enabled |
| Authentication server failover because the authentication server is unreachable | Disabled |
| TACACS+ login authentication (console and Telnet) | Disabled |
| TACACS+ authorization (console and Telnet) | Disabled |
| TACACS+ key | None specified |
| TACACS+ server timeout | 5 seconds |
| TACACS+ retransmit attempts | 2 times |
| RADIUS login authentication (console and Telnet) | Disabled |
| RADIUS authorization (console and Telnet) | Disabled |

*Table 2-2 Default Configuration for Administrative Login Authentication and Authorization (continued)*

| Feature | Default Value |
|---|---|
| RADIUS server IP address | None specified |
| RADIUS server UDP authorization port | Port 1645 |
| RADIUS key | None specified |
| RADIUS server timeout | 5 seconds |
| RADIUS retransmit attempts | 2 times |

### Enforcing Authentication with the Primary Method

The **authentication fail-over server-unreachable** global configuration command allows you to specify that failover to the secondary authentication method should occur only if the primary authentication server is unreachable. This feature ensures that users gain access to the Content Engine using the local database only when nonlocal authentication servers (TACACS+ or RADIUS) are unreachable. For example, when a TACACS+ server is enabled for authentication with user authentication failover configured and the user tries to log in to the Content Engine using an account defined in the local database, login fails. Login succeeds only when the TACACS+ server is unreachable.

### Server Redundancy

You can specify authentication servers with the corresponding authentication server (NTLM, LDAP, or RADIUS) **host** command options, or in the case of TACACS+ servers, with the **server** *hostname* command option, to configure additional servers. These additional servers provide authentication redundancy and improved throughput, especially when Content Engine load-balancing schemes distribute the requests evenly between the servers. If the Content Engine cannot connect to any of the authentication servers, no authentication takes place and users who have not been previously authenticated are denied access.

### Login Authentication and Authorization Through the Local Database

Local authentication and authorization use locally configured login and passwords to authenticate administrative login attempts. The login and passwords are local to each Content Engine and are not mapped to individual usernames.

By default, local login authentication is enabled first. You can disable local login authentication only after enabling one or more of the other administrative login authentication methods. However, when local login authentication is disabled, if you disable all other administrative login authentication methods, local login authentication is reenabled automatically.

### Specifying RADIUS Authentication and Authorization Settings

RADIUS authentication clients reside on the Content Engine running the ACNS 5.x software. When enabled, these clients send authentication requests to a central (remote) RADIUS server, which contains login authentication and network service access information.

To configure RADIUS authentication on a Content Engine, you must configure a set of RADIUS authentication server settings on the Content Engine. You can use the Content Engine GUI or the CLI to configure this set of RADIUS authentication server settings for a Content Engine.

Table 2-3 describes the RADIUS authentication settings.

*Table 2-3 RADIUS Authentication Settings*

| Setting | Description |
|---|---|
| RADIUS server | RADIUS servers that the Content Engine is to use for RADIUS authentication. To enable the Content Engine to use a specific RADIUS server, enter the IP address or hostname of the RADIUS server and port information. Up to five different hosts are allowed. Early deployment of RADIUS was done using port number 1645, although the official port number for RADIUS is now 1812. Up to five different ports are allowed. |
| RADIUS key | Key used to encrypt and authenticate all communication between the RADIUS client (the Content Engine) and the RADIUS server. The maximum number of characters in the key is 15. There is no default.<br>**Tip** Make sure that the same RADIUS key is enabled on the RADIUS server. |
| RADIUS timeout interval | Number of seconds that the Content Engine waits for a response from the specified RADIUS authentication server before declaring a timeout. The range is 1 to 20 seconds. The default value is 5 seconds. |
| RADIUS retransmit count | Number of times that the Content Engine is to retransmit its connection to the RADIUS if the RADIUS timeout interval is exceeded. The range is one to three tries. The default value is two tries. |

After configuring these RADIUS authentication settings on the Content Engine, you can enable RADIUS login authentication and authorization on the Content Engine.

#### Specifying TACACS+ Authentication and Authorization Settings

TACACS+ controls access to network devices by exchanging Network Access Server (NAS) information between a network device and a centralized database to determine the identity of a user or an entity. TACACS+ is an enhanced version of TACACS, a UDP-based access-control protocol specified by RFC 1492. TACACS+ uses TCP to ensure reliable delivery and encrypt all traffic between the TACACS+ server and the TACACS+ daemon on a network device.

TACACS+ works with many authentication types, including fixed password, one-time password, and challenge-response authentication.

When a user requests restricted services, TACACS+ encrypts the user password information using the MD5 encryption algorithm and adds a TACACS+ packet header. This header information identifies the packet type being sent (for example, an authentication packet), the packet sequence number, the encryption type used, and the total packet length. The TACACS+ protocol then forwards the packet to the TACACS+ server.

A TACACS+ server can provide authentication, authorization, and accounting functions. These services, while all part of TACACS+, are independent of one another. A TACACS+ configuration can use any or all of the three services.

When the TACACS+ server receives a packet, it does the following:

- Authenticates the user information and notifies the client that the login authentication has either succeeded or failed.
- Notifies the client that authentication will continue and that the client must provide additional information. This challenge-response process can continue through multiple iterations until login authentication either succeeds or fails.

You can configure a TACACS+ key on the client and server. If you configure a key on the Content Engine, it must be the same as the one configured on the TACACS+ servers. The TACACS+ clients and servers use the key to encrypt all TACACS+ packets transmitted. If you do not configure a TACACS+ key, packets are not encrypted.

TACACS+ authentication is disabled by default. You can enable TACACS+ authentication and local authentication at the same time.

In order to configure TACACS+ authentication on Content Engines, you must configure a set of TACACS+ authentication settings on the Content Engine. You can use the Content Engine CLI or GUI to configure this set of TACACS+ authentication settings for a Content Engine.

Table 2-4 describes the TACACS+ authentication settings.

**Note** No TACACS+ authentication will be performed if no TACACS+ servers are configured on the Content Engine.

*Table 2-4 TACACS+ Authentication Settings*

| Setting | Description |
|---|---|
| TACACS+ server | TACACS+ servers that the Content Engine uses for TACACS+ authentication. Explicitly specify the primary TACACS+ server; otherwise, the Content Engine makes its own decision. You can configure one primary TACACS+ server and two backup TACACS+ servers. TACACS+ uses the standard port (port 49) for communication, based on the specified service. |
| TACACS+ key | Secret key that the Content Engine uses to communicate with the TACACS+ server. The maximum number of characters in the TACACS+ key should not exceed 99 printable ASCII characters (except tabs). An empty key string is the default. All leading spaces are ignored; spaces within and at the end of the key string are not ignored. Double quotes are not required even if there are spaces in the key, unless the quotes themselves are part of the key. There is no default.<br>**Tip** Make sure that the same TACACS+ key is specified on the TACACS+ server. |
| TACACS+ timeout interval | Number of seconds that the Content Engine waits for a response from the specified TACACS+ authentication server before declaring a timeout. The range is 1 to 20 seconds. The default value is 5 seconds. |
| TACACS+ retransmit count | Number of times that the Content Engine retransmits its connection to the TACACS+ if the TACACS+ timeout interval is exceeded. The range is one to three tries. The default value is two tries. |
| TACACS+ password authentication method | Mechanism for password authentication. By default, the Password Authentication Protocol (PAP) is the mechanism for password authentication. The other option is to use ASCII clear text as the password authentication mechanism. |

### TACACS+ Enable Password Attribute

The ACNS software CLI EXEC mode is used for setting, viewing, and testing system operations. It is divided into two access levels, user and privileged. To access privileged-level EXEC mode, enter the **enable** EXEC command at the user access level prompt and specify a privileged EXEC password (superuser or admin-equivalent password) when prompted for a password.

In TACACS+, an enable password feature allows an administrator to define a different enable password per administrative-level user. If an administrative-level user logs in to the Content Engine with a normal-level user account (privilege level of 0) instead of an admin or admin-equivalent user account (privilege level of 15), that user must enter the admin password in order to access privileged-level EXEC mode. This requirement applies even if these ACNS users are using TACACS+ for login authentication.

```
ContentEngine> enable
Password:
```

## Examples

The following example enables local and TACACS+ authentication and authorization, setting TACACS+ as the first method used and local as the secondary method to use if TACACS+ fails:

```
ContentEngine(config)# authentication login tacacs enable primary
ContentEngine(config)# authentication login local enable secondary
ContentEngine(config)# authentication configuration local enable secondary
ContentEngine(config)# authentication configuration tacacs enable primary
```

The following example shows the output of the **show authentication user** command:

```
ContentEngine# show authentication user
Login Authentication:         Console/Telnet Session
----------------------------- ----------------------
local                         enabled (secondary)
radius                        disabled
tacacs                        enabled (primary)

Configuration Authentication: Console/Telnet Session
----------------------------- ----------------------
local                         enabled (secondary)
radius                        disabled
tacacs                        enabled (primary)
Configuration Authentication: Console/Telnet Session
----------------------------- ----------------------
local                         enabled (secondary)
radius                        enabled (tertiary)
tacacs                        enabled (primary)
```

The following example shows the output of the **show statistics authentication** command:

```
ContentEngine# show statistics authentication

Authentication Statistics
--------------------------------------
Number of access requests: 37
Number of access deny responses: 14
Number of access allow responses: 23
```

The following example enables local, TACACS+, and RADIUS authentication and authorization, setting TACACS+ as the first method used, local as the secondary method if the TACACS+ method fails, and RADIUS as the tertiary method to use if both local and TACACS+ fail:

```
ContentEngine(config)# authentication login tacacs enable primary
ContentEngine(config)# authentication login local enable secondary
ContentEngine(config)# authentication login radius enable tertiary
ContentEngine(config)# authentication configuration tacacs enable primary
ContentEngine(config)# authentication configuration local enable secondary
ContentEngine(config)# authentication configuration radius enable tertiary
```

## Related Commands

**ntlm**
**radius-server**
**show authentication**
**show statistics authentication**
**tacacs**
**username**

# auto-register

To enable discovery of a Fast Ethernet or Gigabit Ethernet Content Engine or Content Router and its automatic registration with the Content Distribution Manager through Dynamic Host Configuration Protocol (DHCP), use the **auto-register** global configuration command. To disable this function, use the **no** form of this command.

**auto-register enable** [**FastEthernet** *slot/port* | **GigabitEthernet** *slot/port*]

**no auto-register enable** [**FastEthernet** *slot/port* | **GigabitEthernet** *slot/port*]

## Syntax Description

| | |
|---|---|
| **enable** | Enables the automatic registration of devices using DHCP with the Content Distribution Manager. |
| **FastEthernet** | (Optional) Selects a Fast Ethernet interface for automatic registration using DHCP. |
| *slot/port* | Fast Ethernet slot (0–3) and port number. |
| **GigabitEthernet** | (Optional) Selects a Gigabit Ethernet interface for automatic registration using DHCP. |
| *slot/port* | Gigabit Ethernet slot (1–2) and port number. |

## Defaults

Automatic registration using DHCP is enabled by default.

## Command Modes

global configuration

## Usage Guidelines

The **auto-register enable** command allows a Fast Ethernet or Gigabit Ethernet Content Engine or Content Router to discover the hostname of the Content Distribution Manager through DHCP and to automatically register the device with the Content Distribution Manager. Discovery and registration occur at bootup.

To assign a static IP address using the **interface GigabitEthernet** *slot/port* command, the automatic registration of devices through DHCP must be disabled by using the **no auto-register enable** command, because automatic registration through DHCP is enabled by default.

For autoregistration to work, you must have a DHCP server that is configured with the hostname of the Content Distribution Manager and that is capable of handling vendor class option 43.

**Note** The form of DHCP used for autoregistration is *not* the same as the interface-level DHCP that is configurable through the **ip address dhcp** interface configuration command.

The DHCP server needs to send the vendor class option (option 43) information to the ACNS network device in the format for encapsulated vendor-specific options as provided in RFC 2132. The relevant section of RFC 2132, Section 8.4, is reproduced here as follows:

You should encode the encapsulated vendor-specific options field as a sequence of code/length/value fields of syntax identical to that of the DHCP options field with the following exceptions:

1. There should not be a “magic cookie” field in the encapsulated vendor-specific extensions field.
2. Codes other than 0 or 255 may be redefined by the vendor within the encapsulated vendor-specific extensions field but should conform to the tag-length-value syntax defined in section 2.
3. Code 255 (END), if present, signifies the end of the encapsulated vendor extensions, not the end of the vendor extensions field. If no code 255 is present, then the end of the enclosing vendor-specific information field is taken as the end of the encapsulated vendor-specific extensions field.

In accordance with the RFC standard, the DHCP server needs to send the Content Distribution Manager hostname information in code/length/value format (code and length are single octets). The code for the Content Distribution Manager hostname is 0x01. DHCP server management and configuration are not within the scope of the autoregistration feature.

The ACNS network device sends CISCOCDN as the vendor class identifier in option 60 to facilitate device groupings by customers.

Autoregistration DHCP also requires that the following options are present in the DHCP server’s offer to be considered valid:

- Subnet-mask (option 1)
- Routers (option 3)
- Domain-name (option 15)
- Domain-name-servers (option 6)
- Host-name (option 12)

Interface-level DHCP requires only subnet-mask (option 1) and routers (option 3) for an offer to be considered valid; domain-name (option 15), domain-name-servers (option 6), and host-name (option 12) are optional. All of the above options, with the exception of domain-name-servers (option 6), replace the existing configuration on the system. The domain-name-servers option is added to the existing list of name servers with the restriction of a maximum of eight name servers.

Autoregistration is enabled by default on the first interface of the device. The first interface depends on the Content Engine model as follows:

- For the CE-507, CE-507AV, CE-560, CE-560AV, CE-590, and CR-4430: FastEthernet 0/0
- For the CE-510, CE-511, CE-565, CE-566, CE-7305, CE-7320, CE-7325, and CE-7326: GigabitEthernet 1/0

If you do not have a DHCP server, the device is unable to complete autoregistration and eventually times out. You can disable autoregistration at any time after the device has booted and proceed with manual setup and registration.

## Examples

The following example enables autoregistration on GigabitEthernet port 2/0:

```
ContentEngine(config)# auto-register enable GigabitEthernet 2/0
```

The following example enables autoregistration on FastEthernet port 0/1:

```
ContentEngine(config)# auto-register enable FastEthernet 0/1
```

The following example disables autoregistration on all configured interfaces:

```
ContentEngine(config)# no auto-register enable
```

## Related Commands

**show auto-registration**
**show running-config**
**show startup-config**

# autosense

To enable autosense on an interface, use the **autosense** interface configuration command. To disable this function, use the **no** form of this command.

**autosense**

**no autosense**

## Syntax Description

This command has no arguments or keywords.

## Defaults

Autosense is enabled by default.

## Command Modes

interface configuration

## Usage Guidelines

Cisco router Ethernet interfaces do not negotiate duplex settings. If the Content Engine is connected to a router directly with a crossover cable, the Content Engine interface must be manually set to match the router interface settings. Disable **autosense** before configuring an Ethernet interface. When **autosense** is on, manual configurations are overridden. You must reboot the Content Engine to start autosensing.

Configuring an interface for autosensing causes the full-duplex or half-duplex operation to be disabled. Conversely, configuring an interface for a full-duplex or half-duplex operation causes autosensing to be disabled.

When you set the Cache Engine Ethernet interface speed or duplex function using the **half-duplex**, **full-duplex**, or **bandwidth** commands, you should turn off the corresponding Ethernet switch port autosense function and manually set the duplex function and speed. If you turn off the Ethernet switch port autosense function, you have to manually set the Content Engine Ethernet interface duplex function and speed to match the Ethernet switch port settings. The Content Engine Ethernet interface **autosense** command will only erase manually set configurations.

## Examples

The following example enables autosense on FastEthernet port 0/1:

```
CONTENTENGINE(config)# interface FastEthernet 0/1
CONTENTENGINE(config-if)# autosense
```

The following example disables autosense on FastEthernet port 0/1:

```
CONTENTENGINE(config)# interface FastEthernet 0/1
CONTENTENGINE(config-if)# no autosense
```

## Related Commands

**interface**
**show interface**
**show running-config**
**show startup-config**

# bandwidth (global configuration)

To set an allowable bandwidth usage limit and its duration for Cisco Streaming Engine, RealProxy, RealServer, and WMT streaming media, use the **bandwidth** global configuration command. To remove individual options, use the **no** form of this command.

**bandwidth** {**advanced config-file** *filename-path* | {{**cisco-streaming-engine** *kbits* | **http** *kbits* | **real-proxy** {**incoming** | **outgoing**} *kbits* | **real-server** *kbits* | **wmt** {**incoming** | **outgoing**} *kbits*} {**default** | **max-bandwidth** | **start-time** *weekday time* **end-time** *weekday time*}}}

**no bandwidth** {**advanced config-file** | {{**cisco-streaming-engine** *kbits* | **http** *kbits* | **real-proxy** {**incoming** | **outgoing**} *kbits* | **real-server** *kbits* | **wmt** {**incoming** | **outgoing**} *kbits*} {**default** | **max-bandwidth** | **start-time** *weekday time* **end-time** *weekday time*}}}

## Syntax Description

| | |
|---|---|
| **advanced** | Configures the subnet-based WMT outgoing bandwidth. For more information, see the "Configuring Subnet-Based Outgoing Bandwidth" section on page 2-54. |
| **config-file** | Specifies the path of the advanced bandwidth configuration file. |
| *filename-path* | Name of the advanced bandwidth configuration file saved in the /local1 sysfs directory on the Content Engine. |
| **cisco-streaming-engine** | Configures the duration of allowable bandwidth settings for the Cisco Streaming Engine. |
| *kbits* | Bandwidth size for the Cisco Streaming Engine in kilobits per second (kbps) (0–2147483647). |
| **http** | Configures the pace and rate for pre-positioned HTTP traffic. |
| *kbits* | Bandwidth size for HTTP in kilobits per second (kbps) (0–2000000). |
| **real-proxy** | Configures the duration of allowable bandwidth settings for RealProxy. |
| **incoming** | Configures the duration of allowable incoming bandwidth settings for RealProxy. |
| **outgoing** | Configures the duration of allowable outgoing bandwidth settings for RealProxy. |
| *kbits* | Bandwidth size for RealProxy in kilobits per second (kbps) (0–2147483647). |
| **real-server** | Configures the duration of allowable bandwidth settings for RealServer. |
| *kbits* | Bandwidth size for RealServer in kilobits per second (kbps) (0–2147483647). |
| **wmt** | Configures the duration of allowable bandwidth settings for WMT. For more information, see the "Configuring Incoming and Outgoing WMT Bandwidth" section on page 2-54. |
| **incoming** | Configures the duration of allowable incoming bandwidth settings for WMT. |
| **outgoing** | Configures the duration of allowable outgoing bandwidth settings for WMT. |
| *kbits* | Bandwidth size for WMT in kilobits per second (kbps) (0–2147483647). |
| **default** | Sets the default value for the bandwidth if this value is not configured. |
| **max-bandwidth** | Sets the value for the maximum bandwidth configured. |

| | |
|---|---|
| **start-time** | Sets the starting day of the week and time (hh:mm) for the permitted bandwidth usage. |
| *weekday* | Day of the week to start:<br>**Friday**<br>**Monday**<br>**Saturday**<br>**Sunday**<br>**Thursday**<br>**Tuesday**<br>**Wednesday** |
| *time* | Time of the day to start, in hours and minutes (hh:mm). |
| **end-time** | Sets the ending day of the week and time for the permitted bandwidth usage. |
| *weekday* | Day of the week to end. |

## Defaults

No default behavior or values

## Command Modes

global configuration

## Usage Guidelines

With the various types of traffic originating from a device, every type of traffic, such as streaming media, HTTP, and metadata, consumes network resources. Use the **bandwidth** command to limit the amount of network bandwidth used by the Cisco Streaming Engine, RealNetworks, and WMT streaming media.

The content services bandwidth includes the bandwidth allocation for WMT, RealProxy, RealServer, and Cisco Streaming Engine services. WMT bandwidth settings apply to WMT streaming of live, cached, and pre-positioned content. RealServer bandwidth settings apply to RealMedia streaming of pre-positioned and live content that has been specified in the manifest file for a channel. RealProxy bandwidth settings apply to RealMedia streaming of cached and live content that has not been specified in the manifest file for a channel. Cisco Streaming Engine bandwidth settings apply to the standard RTSP server streaming of pre-positioned content only.

For each type of bandwidth, you can specify the amount of bandwidth to be used for a particular time period. This type is called *scheduled bandwidth*. The *default bandwidth* is the amount of bandwidth associated with each content service type when there is no scheduled bandwidth. In centrally managed deployments (the Content Engines are registered with a Content Distribution Manager), if a Content Engine is assigned to a device group and no default bandwidth has been configured for the Content Engine itself, the device group default bandwidth settings are applied. However, if the default bandwidth has been configured for the Content Engine, then that setting overrides the device group settings. If the Content Engine is a member of multiple device groups, the most recently updated default bandwidth settings are applied.

The *maximum bandwidth* specifies the upper limit for the allowable bandwidth. The total bandwidth configured for all content services must not exceed the bandwidth limits specified for any Content Engine platform model in the ACNS network. In addition, the license keys configured for WMT further restrict the maximum bandwidth available for each Content Engine model.

### Configuring Incoming and Outgoing WMT Bandwidth

The bandwidth between the WMT proxy server (the Content Engine) and the WMT client is called the WMT outgoing bandwidth.

The bandwidth between the WMT proxy and the origin streaming server is called the incoming bandwidth. Because the bandwidth from the edge to the outside IP WAN is limited, you must specify a per session limit (the maximum bit rate per request) for each service that is running on the Content Engine and that consumes the incoming bandwidth (for example, the WMT streaming service), and an aggregate limit (the maximum incoming bandwidth.) You need to control the outgoing bandwidth based on the WMT license that is configured on the Content Engine.

The **bandwidth wmt outgoing** and **bandwidth incoming** global configuration commands enable you to specify a WMT incoming and an outgoing bandwidth as follows:

- Use the **bandwidth wmt outgoing** *kbits* global configuration command to specify the outgoing WMT bandwidth in kbps. This command sets the maximum bandwidth for the WMT content that can delivered to a client that is requesting WMT content. The range of values is between 0–2,147,483,647 kilobits per second (kbps).

  If the specified outgoing bandwidth is above the limit specified by the WMT license, then a warning message displays. However, the specified outgoing bandwidth setting is applied to the Content Engine because the outgoing bandwidth may be configured before the WMT licenses are enabled or an enabled WMT license could be changed to a higher value at a later time.

- Use the **bandwidth wmt incoming** *kbits* global configuration command to specify the incoming WMT bandwidth in kbps. This command sets the maximum bandwidth for the WMT content that can delivered to a Content Engine from the origin streaming server or another Content Engine in the case of a cache miss. The specified bit rate is the maximum incoming WMT per session bit rate. The range of values is between 0–2,147,483,647 kbps. Incoming bandwidth applies to broadcast stations, multicast station, and VoD content from the origin server for a cache miss.

  The incoming bandwidth applies to the following:

  - VoD content from the origin server for a cache miss.
  - Broadcast stations in which the source for the broadcast station and multicast stations is a unicast or a multicast. If the source is a multicast, the specified incoming bandwidth is not applied.
  - Multicast stations in which the source of the multicast station is a unicast (RTSP) or a multicast.

With the ACNS 5.x software, you can also configure a maximum bandwidth for the preloading process using the **pre-load max-bandwidth** global configuration command.

### Configuring Subnet-Based Outgoing Bandwidth

In the ACNS 5.3 software release, the ability to configure IP subnet-based bandwidth control for WMT requests was added. This feature allows you to specify the maximum bandwidth consumption for specific client IP subnets (the aggregate bandwidth for the subnet). This bandwidth control feature is supported for WMT streaming through the following protocols: Windows Media 9 RTSP and HTTP.

You specify the rules for limiting the subnet-based outgoing bandwidth in an XML configuration file. This configuration file is called the advanced bandwidth configuration file. For example, if you have three subnets (Subnet A that is the parent subnet, and Subnet B and C that are within Subnet A) and you have specified three subnet-based bandwidth rules as follows:

- Rule A. Subnet A, 10.1.1.0/24, has been configured with an allowable bandwidth of 10000 kbps
- Rule B. Subnet B, 10.1.1.0/25, has been configured with an allowable bandwidth of 7000 kbps
- Rule C. Subnet C, 10.1.1.128/25, has been configured with an allowable bandwidth of 5000 kbps

Then, even though the total allowed bandwidth of Subnet B and C is 12000 kbps (as defined by Rule B and C in the configuration file), the total bandwidth does not exceed 10000 kbps because of Rule A.

The following is an example of the format of the advanced bandwidth configuration file. It also shows the required order of the lines in the advanced bandwidth configuration file.

```
XML Configuration File Format:
<?xml version="1.0"?>
<BandwidthSpec>
  <BandwidthRule>
    <ClientNetwork>10.77.140.133/32</ClientNetwork>
    <description>(Apply to PC jdoe-w2k)</description>
    <Allow limit="3000" service="wmt"/>
  </BandwidthRule>
  <BandwidthRule>
    <description>Comment (Apply to PCs in subnet 10.77.140.x)</description>
    <Allow limit="50000" service="wmt"/>
    <ClientNetwork>10.77.140.0/24</ClientNetwork>
  </BandwidthRule>
  <BandwidthRule>
    <Allow limit="1400" service="wmt"/>
    <ClientNetwork>10.1.1.1/32</ClientNetwork>
  </BandwidthRule>
  <BandwidthRule>
    <ClientNetwork>10.0.18.0/24</ClientNetwork>
    <description>(Apply to my PC)</description>
    <Allow limit="700" service="wmt" />
  </BandwidthRule>
  <BandwidthRule>
    <ClientNetwork>10.0.19.0/24</ClientNetwork>
    <description>(Apply to my PC)</description>
    <Allow limit="700" service="wmt" />
  </BandwidthRule>
  <BandwidthRule>
    <ClientNetwork>10.0.20.0/24</ClientNetwork>
    <description>(Apply to my PC)</description>
    <Allow limit="700" service="wmt" />
  </BandwidthRule>
  <BandwidthRule>
    <ClientNetwork>10.0.21.0/24</ClientNetwork>
    <description>(Apply to my PC)</description>
    <Allow limit="700" service="wmt" />
  </BandwidthRule>
  <Default limit="3000" service="wmt" />
</BandwidthSpec>
```

where

- The <description> tag is optional.
- The <ClientNetwork> - IPAddress/Netmask entry is a required field.
- If the <Allow limit> field is specified as –1, the bandwidth allowed is unlimited.

- The Service tag currently has only one supported option (the wmt option).
- The <Default> tag is optional. This tag is used to configure the default bandwidth. If none of the subnet bandwidth rules match, the default rule is applied if it is configured.

**Note** The **bandwidth wmt outgoing** global configuration command is used to configure the total outgoing WMT bandwidth, which controls the total outgoing bandwidth used for WMT streaming; irrespective of any subnet-based bandwidth configuration that is specified through the advanced bandwidth configuration file.

You use the **bandwidth advanced config-file** *filename-path* global configuration command to specify the path of the advanced bandwidth configuration file. Use the complete pathname when you specify the path of the bandwidth configuration file. You use FTP to download this configuration file to the Content Engine so that the file is available in the local sysfs partition on the Content Engine.

## Examples

The following example limits the RealProxy bandwidth to 1000 kbps from 8:00 a.m. to 6:00 p.m on Mondays through Fridays:

```
ContentEngine(config)# bandwidth allow 1000 real-proxy start-time monday 8:00 end-time
friday 18:00
```

**Tip** For a schedule spanning two days (for example, from 8:00 p.m. to 8:00 a.m.), you must configure two schedules in order to span the two days; one from 8:00 p.m. to 11:59 p.m. (2000 to 2359) and another from 12:00 a.m. to 8:00 a.m. (0000 to 0800).

The following example specifies the path of the advanced bandwidth configuration file new_file.xml that resides in the /local1 directory on the Content Engine:

```
ContentEngine(config)# bandwidth advanced config-file /local1/new_file.xml
```

The following example configures the default bandwidth for pre-positioned HTTP traffic as 2000 kbps:

```
CONTENTENGINE(config)# bandwidth http 2000 default
```

The following example configures the maximum bandwidth for the Cisco Streaming Engine as 56000 kbps:

```
CONTENTENGINE(config)# bandwidth cisco-streaming-engine 56000 max-bandwidth
```

## Related Commands

**bandwidth** (interface configuration)
**interface**
**show bandwidth**
**show interface**
**show running-config**
**show startup-config**
**show statistics bandwidth**

# bandwidth (interface configuration)

To configure an interface bandwidth, use the **bandwidth** interface configuration command. To restore default values, use the **no** form of this command.

**bandwidth** {**10** | **100** | **1000**}

**no bandwidth** {**10** | **100** | **1000**}

## Syntax Description

| | |
|---|---|
| **10** | Sets the bandwidth to 10 megabits per second (Mbps). |
| **100** | Sets the bandwidth to 100 Mbps. |
| **1000** | Sets the bandwidth to 1000 Mbps. This option is not available on all ports and is the same as enabling autosense on the interface. |

## Defaults

No default behaviors or values

## Command Modes

interface configuration

## Usage Guidelines

To configure an interface bandwidth on a Content Engine, use the **bandwidth** interface configuration command. The bandwidth is specified in megabits per second (Mbps). The **1000** Mbps option is not available on all ports and is the same as enabling autosense on the interface. On a Content Engine CE-7320 model that has an optical Gigabit Ethernet interface, you cannot change the speed of this interface. Therefore, Gigabit Ethernet interfaces only run at 1000 Mbps for a CE-7320. For newer models of the Content Engine (for example, the CE-510, CE-565, CE-7305, CE-7325, and CE-7326) that have a Gigabit Ethernet interface over copper, this restriction does not apply; you can configure these Gigabit Ethernet interfaces to run at 10, 100, or 1000 Mbps. On these newer Content Engine models, the 1000-Mbps setting implies autosense (for example, you cannot configure the Gigabit Ethernet interface to run at 1000 Mbps and half duplex). The ACNS 5.x software automatically enables autosense if the speed is set to 1000 Mbps.

In the ACNS 5.3 software and later releases, you can configure the Gigabit Ethernet interface settings (autosense, bandwidth, and duplex settings) if the Gigabit-over-copper-interface is up or down. If the interface is up, it will apply the specific interface settings. If the interface is down, the specified settings are stored and then applied when the interface is brought up. For example, you can specify any of the following commands for a Gigabit-over-copper-interface, which is currently down, and have these settings automatically applied when the interface is brought up:

```
ContentEngine(config-if)# bandwidth 10
ContentEngine(config-if)# bandwidth 100
ContentEngine(config-if)# bandwidth 1000
ContentEngine(config-if)# autosense
ContentEngine(config-if)# half-duplex
ContentEngine(config-if)# full-duplex
```

**Note** In the ACNS 5.2.x software and earlier releases, you could only configure the Gigabit Ethernet interface settings if the interface is up.

You cannot configure the Gigabit Ethernet interface settings on an optical Gigabit Ethernet interface (for example, if the Content Engine is a CE-7320 model).

## Examples

The following example shows how to set an interface bandwidth to 10 Mbps:

```
ContentEngine(config-if)# bandwidth 10
```

The following example shows how to restore default bandwidth values on an interface:

```
ContentEngine(config-if)# no bandwidth
```

## Related Commands

**interface**

# banner

To configure the EXEC, login, and message-of-the-day (MOTD) banners, use the **banner** global configuration command. To disable the banner feature, use the **no** form of this command.

**banner** {**enable** | **exec** {**message** *line* | *message_text*} | **login** {**message** *line* | *message_text*} | **motd** {**message** *line* | *message_text*}}

**no banner** {**enable** | **exec** [**message**] | **login** [**message**] | **motd** [**message**]}

## Syntax Description

| | |
|---|---|
| **enable** | Enables banner support on the Content Engine. |
| **exec** | Configures an EXEC banner. |
| **message** | Specifies a message to be displayed when an EXEC process is created. |
| *line* | EXEC message text on a single line. The Content Engine translates the \n portion of the message to a new line when the EXEC banner is displayed to the user. |
| *message_text* | EXEC message text on one or more lines. Press the **Return** key or enter delimiting characters (\n) to specify an EXEC message to appear on a new line. Supports up to a maximum of 980 characters, including new-line characters (\n). Enter a period (.) at the beginning of a new line to save the message and return to the prompt for the global configuration mode.<br>**Note** The EXEC banner content is obtained from the command line input that the user enters after being prompted for the input. |
| **login** | Configures a login banner. |
| **message** | Specifies a message to be displayed before the username and password login prompts. |
| *line* | Login message text on a single line. The Content Engine translates the \n portion of the message to a new line when the login banner is displayed to the user. |
| *message_text* | Login message text on one or more lines. Press the **Return** key or enter delimiting characters (\n) to specify a login message to appear on a new line. Supports up to a maximum of 980 characters, including new-line characters (\n). Enter a period (.) at the beginning of a new-line to save the message and return to the prompt for the global configuration mode.<br>**Note** The login banner content is obtained from the command line input that the user enters after being prompted for the input. |
| **motd** | Configures an MOTD banner. |
| **message** | Specifies an MOTD message. |

| | |
|---|---|
| *line* | MOTD message text on a single line. The Content Engine translates the \n portion of the message to a new line when the MOTD banner is displayed to the user. |
| *message_text* | MOTD message text on one or more lines. Press the **Return** key or enter delimiting characters (\n) to specify an MOTD message to appear on a new line. Supports up to a maximum of 980 characters, including new-line characters (\n). Enter a period (.) at the beginning of a new line to save the message and return to the prompt for the global configuration mode.<br>**Note** The MOTD banner content is obtained from the command line input that the user enters after being prompted for the input. |

## Defaults

Banner support is disabled by default

## Command Modes

global configuration

## Usage Guidelines

In the ACNS 5.3 software and later releases, you can configure the following three types of banners in any ACNS software device mode:

- The MOTD banner sets the message of the day. This message is the first message that is displayed when a login is attempted.
- The login banner is displayed after the MOTD banner but before the actual login prompt appears.
- The EXEC banner is displayed after the EXEC CLI shell has started.

**Note** All of these banners are effective on a console, Telnet, or a Secure Shell (SSH) version 2 session.

After you configure the banners, enter the **banner enable** global configuration command to enable banner support on the Content Engine. Enter the **show banner** EXEC command to display information about the configured banners.

**Note** When you run an SSH version 1 client and log in to the Content Engine, the MOTD and login banners are not displayed. You need to use SSH version 2 to display the banners when you log in to the Content Engine.

## Examples

The following example shows how to enable banner support on the Content Engine:

```
ContentEngine (config)# banner enable
```

The following example shows how to use the **banner motd message** global configuration command to configure the MOTD banner. In this example, the MOTD message consists of a single line of text.

```
ContentEngine(config)# banner motd message This is an ACNS 5.3 device
```

The following example shows how to use the **banner motd message** global command to configure a MOTD message that is longer than a single line. In this case, the Content Engine translates the \n portion of the message to a new line when the MOTD message is displayed to the user.

```
ContentEngine (config)# banner motd message "This is the motd message.
\nThis is an ACNS 5.3 device\n"
```

The following example shows how to use the **banner login message** global configuration command to configure a MOTD message that is longer than a single line. In this case, Content Engine A translates the \n portion of the message to a new line in the login message that is displayed to the user.

```
ContentEngine(config)# banner login message "This is login banner.
\nUse your password to login\n"
```

The following example shows how to use the **banner exec** global configuration command to configure an interactive banner. The **banner exec** command is similar to the **banner motd message** commands except that for the **banner exec** command, the banner content is obtained from the command line input that the user enters after being prompted for the input.

```
CONTENTENGINE(config)# banner exec
Please type your MOTD messages below and end it with '.' at beginning of line:
(plain text only, no longer than 980 bytes including newline)
This is the EXEC banner.\nUse your ACNS username and password to log in to this Content
Engine.\n
.
Message has 99 characters.
CONTENTENGINE(config)#
```

Assume that a Content Engine has been configured with the MOTD, login, and EXEC banners as shown in the previous examples. When a user uses an SSH session to log in to the Content Engine, the user will see a login session that includes a MOTD banner and a login banner that asks the user to enter a login password as follows:

```
This is the motd banner.
This is an ACNS 5.3 device
This is login banner.
Use your password to login.

Cisco Content Engine

admin@ce's password:
```

After the user enters a valid login password, the EXEC banner is displayed, and the user is asked to enter the ACNS username and password as follows:

```
Last login: Fri Oct  1 14:54:03 2004 from client
System Initialization Finished.
This is the EXEC banner.
Use your ACNS username and password to log in to this Content Engine.
```

After the user enters a valid ACNS username and password, the Content Engine CLI is displayed. The CLI prompt varies depending on the privilege level of the login account. In the following example, because the user entered a username and password that had administrative privileges (privilege level of 15), the EXEC mode CLI prompt is displayed:

```
ContentEngine#
```

## Related Commands

**show banner**

# bitrate

To configure the maximum pacing bit rate for large files sent using the HTTP protocol and to separately configure WMT bit-rate settings, use the **bitrate** global configuration command. To remove the bit-rate settings, use the **no** form of this command.

**bitrate** {**http default** *bitrate* | **wmt** {**incoming** *bitrate* | **outgoing** *bitrate*}}

**no bitrate** {**http default** *bitrate* | **wmt** {**incoming** | **outgoing**}}

## Syntax Description

| | |
|---|---|
| **http** | Configures the maximum pacing bit rate in kilobits per second (kbps) for large files sent using the HTTP protocol. |
| **default** | Sets the default bit rate in kbps for large files. |
| *bitrate* | Bit rate in kbps (0–2000000). |
| **wmt** | Configures the bit rate, in kbps, for large files sent using the WMT protocol. |
| **incoming** | Sets the incoming bit-rate settings. |
| *bitrate* | Incoming bit rate in kbps (0–2147483647). |
| **outgoing** | Sets the outgoing bit-rate settings. |
| *bitrate* | Outgoing bit rate in kbps (0–2147483647). |

## Defaults

**http** *bitrate*: 1500 kbps

**wmt incoming** *bitrate*: 0 (no limit)

**wmt outgoing** *bitrate*: 0 (no limit)

## Command Modes

global configuration

## Usage Guidelines

The ACNS 5.x software includes the Windows Media Technologies (WMT) proxy, which has the ability to cache on-demand media files when the user requests these files for the first time. All subsequent requests for the same file are served by the WMT proxy using the RTSP protocol. The WMT proxy can also live-split a broadcast, which causes only a single unicast stream to be requested from the origin server in response to multiple client requests for the stream.

The bit rate between the proxy and the origin server is called the incoming bit rate. Use the **bitrate** command to limit the maximum bit rate per session for large files delivered using either the HTTP or the RTSP protocol. The **bitrate wmt incoming** and **bitrate wmt outgoing** global configuration commands enable you to specify a WMT incoming and outgoing WMT per session bit rate as follows:

- Use the **bitrate wmt incoming** *bitrate* global configuration command to specify the maximum incoming streaming bit rate per session that can be delivered to the WMT proxy server (a Content Engine) from the origin streaming server or another Content Engine in the case of a cache miss. The specified bit rate is the maximum incoming WMT per session bit rate. The range of values is between 0–2,147,483,647 kbps. The default value is 0 (no bit-rate limit).

- Use the **bitrate wmt outgoing** *bitrate* global configuration command to set the maximum outgoing streaming bit rate per session that can delivered to a client that is requesting WMT content. The specified bit rate is the maximum outgoing WMT per session bit rate). The range of values is between 0–2,147,483,647 kbps. The default value is 0 (no bit-rate limit).

  The outgoing bandwidth applies to the following:

  - VoD content from the WMT proxy server on the Content Engine in the case of a cache miss.
  - Broadcast stations and multicast stations that are configured on the Content Engine. The source for the broadcast station can be a unicast or a multicast.

**Note** The aggregate bandwidth used by all concurrent users is still limited by the default device bandwidth or by the limit configured using the **bandwidth** global configuration command.

### Variable WMT Bit Rates

A content provider can create streaming media files at different bit rates to ensure that different clients who have different connections—for example, modem, DSL, or LAN—can choose a particular bit rate. The WMT caching proxy can cache multiple bit-rate files or variable bit-rate (VBR) files, and based on the bit rate specified by the client, it serves the appropriate stream. Another advantage of creating variable bit-rate files is that you only need to specify a single URL for the delivery of streaming media.

**Note** In the case of multiple bit-rate files, the Content Engine that is acting as the WMT proxy server only retrieves the bit rate that the client has requested.

## Examples

The following example shows how to configure an incoming bit rate for a file sent over HTTP:

```
ContentEngine(config)# bitrate http default 100
```

The following example shows how to configure an incoming bit rate for a file sent using WMT. Use the **show wmt** command to verify that the incoming bit rate has been modified.

```
ContentEngine(config)# bitrate wmt incoming 300000
ContentEngine(config)# exit
ContentEngine# show wmt
--------- WMT Server Configurations -----------------
WMT golden license key installed
WMT outgoing bandwidth limit enforced: 250000 Kbits/sec
WMT end user license agreement accepted
WMT is enabled
WMT disallowed client protocols: none
WMT outgoing bandwidth configured is 250000 Kbits/sec
WMT incoming bandwidth configured is 250000 Kbits/sec
WMT max sessions configured: 3568
WMT max sessions platform limit: 3568
WMT max sessions enforced: 3568 sessions
WMT max outgoing bit rate allowed per stream has no limit
WMT max incoming bit rate allowed per stream has no limit
WMT cache is enabled
WMT cache max-obj-size: 1024 MB
WMT debug level: 0
WMT L4 switch is not enabled
WMT debug client ip not set
WMT debug server ip not set
WMT/REAL cache space partition: wmt 70%, real 30%
WMT Stripping ? from Live URL is not enabled
WMT accelerate live-split is enabled
WMT accelerate proxy-cache is enabled
WMT accelerate VOD is enabled
WMT fast-start is enabled
WMT fast-start max. bandwidth per player is 3500 (Kbps)
WMT fast-cache is enabled
WMT fast-cache acceleration factor is 5
WMT maximum data packet MTU (TCP) enforced is 1472 bytes
WMT maximum data packet MTU (UDP) is 1500 bytes
WMT client idle timeout is 60 seconds
WMT forward logs is enabled
WMT server inactivity-timeout is 65535
WMT Transaction Log format is Windows Media Services 9.0 logging and CE specific
 information
RTSP Gateway incoming port 554
RTSP Gateway L4-switch not enabled
RTSP Gateway Transparent Interception (WCCP):
             Configured on port: 554

--------- WMT HTTP Configurations -------------------
WMT http extensions allowed:
asf none nsc wma wmv nsclog

--------- WMT Proxy Configurations ------------------
Outgoing Proxy-Mode:
--------------------
MMS-over-HTTP Proxy-Mode:
  is not configured.
RTSP Proxy-Mode:
  is not configured.
ContentEngine#
```

## Related Commands

**show http all**
**show wmt**

# bypass

To enable transparent error handling and dynamic authentication bypass, and to configure static bypass lists, use the **bypass** global configuration command. To disable the bypass feature, use the **no** form of this command.

**bypass** {**auth-traffic enable** | **gateway** *ipaddress* | **load** {**enable** | **in-interval** *seconds* | **out-interval** *seconds* | **time-interval** *minutes*} | **static** {*clientip* | **any-client**} {*serverip* | **any-server**} | **timer** *minutes*}

**no bypass** {**auth-traffic enable** | **gateway** *ipaddress* | **load** {**enable** | **in-interval** *seconds* | **out-interval** *seconds* | **time-interval** *minutes*} | **static** {*clientip* {*serverip* | **any-server**} | **any-client** *serverip*} | **timer**}

## Syntax Description

| | |
|---|---|
| **auth-traffic** | Sets the authenticated traffic bypass configuration. |
| **gateway** | Configures a router to which bypassed packets are redirected when the Content Engine receives requests redirected by a Layer 4 switch. |
| *ipaddress* | IP address of the router acting as the bypass gateway. |
| **enable** | Enables the authenticated traffic bypass. |
| **load** | Sets the bypass load configuration. |
| **enable** | Enables the bypass load. |
| **in-interval** | Sets the time interval between the buckets coming back. |
| *seconds* | Time in seconds (2–600). |
| **out-interval** | Sets the time interval between the bypassing buckets. |
| *seconds* | Time in seconds (4–600). |
| **time-interval** | Sets the time interval between one bucket being bypassed and the next. |
| *minutes* | Time in minutes (1–1440). |
| **static** | Adds a static entry to the bypass list. |
| *clientip* | IP address from which requests will bypass the Content Engine. |
| **any-client** | Bypasses the HTTP traffic from any client destined to a particular server. |
| *serverip* | IP address to which requests will bypass the Content Engine. |
| **any-server** | Requests from a specified client to any server to bypass the Content Engine. |
| **timer** | Sets the authentication bypass timer in minutes. The bypass entry is removed from the dynamic list when the timer expires. |
| *minutes* | Time in minutes (1–1440). |

## Defaults

**bypass timer**: 20 minutes

**in-interval**: 60 seconds

**out-interval**: 4 seconds

**time-interval**: 10 minutes

## Command Modes

global configuration

## Usage Guidelines

Bypass refers to a method that the Content Engine can use to handle various error responses (including authentication failure) from an origin server. When the Content Engine receives an error response from an origin server, it adds an entry for the server to its bypass list. When it receives subsequent requests for content residing on the bypassed server, it redirects packets to the bypass gateway. If no bypass gateway is configured, then the packets are returned to the redirecting Layer 4 switch.

If both WCCP Version 2 and a Layer 4 switch are configured, then requests redirected to the Content Engine by WCCP are bypassed to the redirecting WCCP Version 2-enabled router. Requests redirected to the Content Engine by the Layer 4 switch are redirected to the bypass gateway. The Content Engine can differentiate between requests arriving as a result of WCCP and as a result of the Layer 4 switch.

Bypass features can be used with a WCCP Version 2-enabled router or with a Layer 4 switch, such as the Cisco Content Switching Module or Cisco Content Services switch. The Content Engine cannot set up a bypass for proxy-style requests.

### Using a Bypass Gateway

To enable bypass of HTTP requests with a Layer 4 switch, use the **http l4-switch enable** command. To identify the router to which the Content Engine will direct responses when errors are received from the origin server, use the **bypass gateway** command. Replace *ipaddress* with the IP address of a router that is a Layer 2 neighbor of the Content Engine.

With RealMedia RTSP transparent redirection, a Layer 4 switch redirects RealMedia requests to the Content Engine (acting as a transparent proxy server). RTSP transparent redirection is used to support RealMedia transparent caching on a standalone Content Engine. To enable transparent redirection of RTSP requests through Layer 4 switching, enter the **rtsp L4-switch enable** global configuration command. After you enter the command, a message appears indicating that Layer 4 switching for RTSP has been enabled on the Content Engine:

```
ContentEngine(config)# rtsp L4-switch enable
Turn on l4 switch
```

### Authentication Traffic Bypass

Some websites, because of IP authentication, do not allow the Content Engine to connect directly on behalf of the client. To preserve transparency and to avoid a disruption of service, the Content Engine can use authentication traffic bypass to automatically generate a dynamic access list for these client/server pairs. Authentication bypass triggers are also propagated upstream and downstream in the case of hierarchical caching. When a client/server pair goes into authentication bypass, it is bypassed for an amount of time set by the **bypass timer** command (20 minutes by default).

### Dynamic Traffic Bypass

The following two scenarios describe typical dynamic traffic bypass situations:

Scenario 1—Dynamic Bypass upon Receiving a Web Server Error

A user issues an HTTP request from a web browser. The request is transparently intercepted and redirected to the Content Engine. The Content Engine accepts the incoming TCP connection from the web browser, determines that the request is for an object not in storage (cache miss), issues a request for the object from the origin web server, but receives some kind of error (for instance, a protocol or authentication error) from the web server.

The Content Engine has already accepted the TCP connection from the web browser and the three-way TCP handshake has taken place. The Content Engine detects that the transaction with the web server is failed but does not know the cause (the origin web server is performing authentication based on user source IP address, incompatibility between the TCP stacks, and so forth).

By default, if the Content Engine receives an error from the origin server, the Content Engine sends a 200 OK response back to the browser with instructions to refresh the URL as follows:

```
HTTP/1.0 200 OK
Cache-Control; no-cache
Connection: Close
```

This refresh instruction causes the client to send the request again. On the connection retry, the Content Engine does not accept the connection. It passes the request back to the WCCP-enabled router or switch unintercepted. The router then sends the flow toward the origin web server directly from the web browser, which bypasses the Content Engine.

Scenario 2—Dynamic Bypass upon Receiving an Unsupported Protocol

When the Content Engine receives non-HTTP requests over TCP port 80, the Content Engine issues a retry response, closes the connection, and does not accept subsequent connections in the same manner as in scenario 1.

**Note** Non-HTTP includes nonconforming HTTP as well as different protocols such as Secure Shell (SSH), Simple Mail Transfer Protocol (SMTP), or Network News Transport Protocol (NNTP). An example of nonconforming HTTP is the failure of a web server to issue two carriage return and line feeds at the end of the HTTP header section.

These two scenarios implement the WCCP return-path functionality in WCCP, which is a mechanism that allows a Content Engine to return traffic to the WCCP-enabled router or switch, telling the router or switch to forward the packets as if the Content Engine was not present.

Typically, approximately 3 percent of all HTTP traffic flows have some kind of failure condition. These failed flows are automatically retried using authentication bypass or dynamic client bypass, demonstrating that the failure conditions were preexisting and not due to the deployment of transparent caching.

### Overload Bypass

If a Content Engine becomes overwhelmed with traffic, it can use the bypass load feature to reroute the overload traffic.

When the Content Engine is overloaded and the **bypass load** command is enabled, the Content Engine bypasses a bucket. If the load remains too high, another bucket is bypassed, and so on until the Content Engine can handle the load. The time interval between one bucket being bypassed and the next is set by the **out-interval** option. The default is 4 seconds.

When the first bucket bypass occurs, a time interval must elapse before the Content Engine begins to again service the bypassed buckets. The duration of this interval is set by the **time-interval** option. The default is 10 minutes.

When the Content Engine begins to service the bypassed traffic again, it begins with a single bypassed bucket. If the load is serviceable, the Content Engine picks up another bypassed bucket, and so on. The time interval between picking up one bucket and the next is set by the **in-interval** option. The default is 60 seconds.

### Static Bypass

The **bypass static** command permits traffic from specified sources to bypass the Content Engine. The types of traffic sources are as follows:

- Specific web client to a specific web server
- Specific web client to any web server
- Any web client to a specific web server

Wildcards in either the source or the destination field are not supported.

To clear all static configuration lists, use the **no** form of the command.

**Note** You must not exceed 50 bypass list entries for any one Content Engine.

## Examples

The following example forces HTTP traffic from a specified client to a specified server to bypass the Content Engine:

```
ContentEngine(config)# bypass static 10.1.17.1 172.16.7.52
```

The following example forces all HTTP traffic destined to a specified server to bypass the Content Engine:

```
ContentEngine(config)# bypass static any-client 172.16.7.52
```

The following example forces all HTTP traffic from a specified client to any web server to bypass the Content Engine:

```
ContentEngine(config)# bypass static 10.1.17.1 any-server
```

The following example forces all authenticated HTTP traffic to bypass the Content Engine for 24 hours:

```
ContentEngine(config)# bypass auth-traffic enable
ContentEngine(config)# bypass timer 1440
```

A static list of source and destination addresses helps to isolate instances of problem-causing clients and servers. You can display the list as follows:

- To display static configuration list items, use the **show bypass list** command as follows:

```
ContentEngine# show bypass list
Client              Server          Entry type
------              ------          ----------
10.1.17.1:0         172.16.7.52:0   static-config
any-client:0        172.16.7.52:0   static-config
10.1.17.2:0         any-server:0    static-config
```

- The total number of entries in the bypass list is reported by the **show bypass summary** command as follows:

```
Total number of HTTP connections bypassed = 0
        Connections bypassed due to system overload              = 0
        Connections bypassed due to authentication issues        = 0
        Connections bypassed due to facilitate error transparency = 0
        Connections bypassed due to static configuration         = 0

Total number of entries in the bypass list = 3
        Number of Authentication bypass entries = 0
        Number of Error bypass entries          = 0
        Number of Static Configuration entries  = 3
```

## Related Commands

**clear bypass**
**http l4-switch**
**rtsp l4-switch**
**rule**
**show bypass**
**show statistics bypass**

# cache

To perform cache-related actions, use the **cache** EXEC command.

**cache** {**clear** [**force**] | **reset** | **synchronize**}

To clear the disk of all cached content, use the **cache clear** EXEC command.

## Syntax Description

| | |
|---|---|
| **clear** | Clears the cache. |
| **force** | (Optional) Forces deletion of all cached objects. |
| **reset** | Resets the cache (unmounts, formats, and mounts cache file system [cfs] volumes). |
| **synchronize** | Synchronizes the cache. |

## Defaults

No default behavior or values

## Command Modes

EXEC

## Usage Guidelines

The **cache clear** command removes all cached contents from the currently mounted cfs volumes. Objects being read or written are removed when they stop being busy. The equivalent to this command is the **clear cache** or **cfs clear** command.

**Caution** The **cache clear** command is irreversible, and all cfs cached content will be erased.

The **cache clear force** deletes all cfs objects, whether busy or not, and may generate broken GIF or HTML messages for objects that were being read from the disk when the command was executed. If an object is being written to the Content Engine disk when a **cache clear force** command is executed, the application stops caching that object but still delivers the object from the web server to the client.

The **cache synchronize** command synchronizes the cache file system and the media file system contents from memory to disk. Although synchronization is performed at regular intervals while the Content Engine is operating, this command can be used to ensure that all data is written to disk before you reset or turn off the Content Engine. Synchronization can also be done using the **cfs sync** and **mediafs sync** commands.

## Examples

The following example forces deletion of all cached objects:

```
ContentEngine# cache clear force
```

## Related Commands

**cfs**
**clear cache**

# cd

To change from one directory to another directory, use the **cd** EXEC command.

**cd** *directoryname*

## Syntax Description

| | |
|---|---|
| *directoryname* | Directory name. |

## Defaults

No default behavior or values

## Command Modes

EXEC

## Usage Guidelines

Use this command to maneuver between directories and for file management. The directory name becomes the default prefix for all relative paths. Relative paths do not begin with a slash (/). Absolute paths begin with a slash (/).

## Examples

The following example shows how to use a relative path:

```
ContentEngine(config)# cd local1
```

The following example shows how to use an absolute path:

```
ContentEngine(config)# cd /local1
```

## Related Commands

**deltree**
**dir**
**lls**
**ls**
**mkdir**
**pwd**

# cdm

To configure the Content Distribution Manager IP address to be used for the Content Engines or Content Routers, or to configure the role and GUI parameters on a Content Distribution Manager device, use the **cdm** global configuration command. To negate these actions, use the **no** form of this command.

**cdm** {**ip** {*hostname* | *ip-address* | **role** {**primary** | **standby**} | **ui port** *port-num*}

**no cdm** {**ip** | **role** {**primary** | **standby**} | **ui port**}

## Syntax Description

| | |
|---|---|
| **ip** | Configures the Content Distribution Manager hostname or IP address. |
| *hostname* | Hostname of the Content Distribution Manager. |
| *ip-address* | IP address of the Content Distribution Manager. |
| **role** | Available from the Content Distribution Manager CLI only. Configures the Content Distribution Manager role to either primary or standby. |
| **primary** | Configures the Content Distribution Manager to be the primary Content Distribution Manager. |
| **standby** | Configures the Content Distribution Manager to be the standby Content Distribution Manager. |
| **ui** | Available from the Content Distribution Manager CLI only. Configures the Content Distribution Manager GUI port address. |
| **port** | Configures the Content Distribution Manager GUI port. |
| *port-num* | Port number (1–65535). |

## Defaults

No default behavior or values

## Command Modes

global configuration

## Usage Guidelines

In the ACNS 5.3 software release and later releases, you can use the **cdm ui port** global configuration command to change the Content Distribution Manager GUI port from the standard number 8443 as follows:

```
ContentDistributionManager(config)# cdm ui port 35535
```

**Note** The **role** and **ui** options are available on Content Distribution Manager devices only. Changing the Content Distribution Manager GUI port number automatically restarts the Centralized Management System (CMS) service if this has been enabled.

The **cdm ip** command associates the device with the Content Distribution Manager so that the device can be approved as a part of the network.

After the device is configured with the Content Distribution Manager IP address, it presents a self-signed security certificate and other essential information, such as its IP address or hostname, disk space allocation, and so forth, to the Content Distribution Manager.

### Configuring Devices Inside a NAT

In an ACNS network, there are two methods for a device registered with the Content Distribution Manager (Content Engines, Content Routers, or standby Content Distribution Manager) to obtain configuration information from the primary Content Distribution Manager. The primary method is for the device to periodically poll the primary Content Distribution Manager on port 443 to request a configuration update. You cannot configure this port number. The backup method is when the Content Distribution Manager pushes configuration updates to a registered device as soon as possible by issuing a notification to the registered device on port 443. This method allows changes to take effect in a timelier manner. You cannot configure this port number even when the backup method is being used. ACNS networks do not work reliably if devices registered with the Content Distribution Manager are unable to poll the Content Distribution Manager for configuration updates. Similarly, when a receiver Content Engine requests content and content metadata from a forwarder Content Engine, it contacts the forwarder Content Engine on port 443.

All the above methods become complex in the presence of Network Address Translation (NAT) firewalls. When a device (Content Engines at the edge of the network, Content Routers, and primary or standby Content Distribution Managers) is inside a NAT firewall, those devices that are inside the same NAT use one IP address (the inside local IP address) to access the device and those devices that are outside the NAT use a different IP address (the inside global IP address) to access the device. A centrally managed device advertises only its inside local IP address to the Content Distribution Manager. All other devices inside the NAT use the inside local IP address to contact the centrally managed device that resides inside the NAT. A device that is not inside the same NAT as the centrally managed device is not able to contact it without special configuration.

If the primary Content Distribution Manager is inside a NAT, you can allow a device outside the NAT to poll it for getUpdate requests by configuring a *static translation* (inside global IP address) for the Content Distribution Manager’s inside local IP address on its NAT, and using this address, rather than the Content Distribution Manager’s inside local IP address, in the **cdm ip** *ip-address* global configuration command when you register the device to the Content Distribution Manager. If a Content Engine or Content Router is inside a NAT and the Content Distribution Manager is outside the NAT, you can allow the Content Engine or Content Router to poll for getUpdate requests by configuring a static translation (inside global IP address) for the Content Engine or Content Router's inside local address on its NAT and specifying this address in the Use IP Address field under the NAT Configuration heading in the Device Activation window.

**Note** Static translation establishes a one-to-one mapping between your inside local address and an inside global address. Static translation is useful when a host on the inside must be accessible by a fixed address from the outside.

### Standby Content Distribution Managers

The Cisco ACNS software implements a standby Content Distribution Manager. This process allows you to maintain a copy of the ACNS network configuration. If the primary Content Distribution Manager fails, the standby can be used to replace the primary.

For interoperability, when a standby Content Distribution Manager is used, it must be at the same software version as the primary Content Distribution Manager in order to maintain the full Content Distribution Manager configuration. Otherwise, the standby Content Distribution Manager detects this status and does not process any configuration updates that it receives from the primary Content Distribution Manager until the problem is corrected.

**Note** We recommend that you upgrade your standby Content Distribution Manager first and then upgrade your primary Content Distribution Manager. We also recommend that you create a database backup on your primary Content Distribution Manager and copy the database backup file to a safe place before you upgrade the software.

### Switching a Content Distribution Manager from Warm Standby to Primary

If your primary Content Distribution Manager becomes inoperable for some reason, you can manually reconfigure one of your warm standby Content Distribution Managers to be the primary Content Distribution Manager. Configure the new role by using the global configuration **cdm role primary** command as follows:

```
DeviceName# configure
DeviceName(config)# cdm role primary
```

This command changes the role from standby to primary and restarts the management service to recognize the change.

**Note** Check the status of recent updates from the primary Content Distribution Manager. Use the **show cms info** EXEC command and check the time of the last update. To be current, the update time should be between 1 and 5 minutes old. You are verifying that the standby Content Distribution Manager has fully replicated the primary Content Distribution Manager configuration. If the update time is not current, check whether there is a connectivity problem or if the primary Content Distribution Manager is down. Fix the problem, if necessary, and wait until the configuration has replicated as indicated by the time of the last update. Make sure that both Content Distribution Managers have the same Coordinated Universal Time (UTC) configured.

If you switch a warm standby Content Distribution Manager to primary while your primary Content Distribution Manager is still online and active, both Content Distribution Managers detect each other, automatically shut themselves down, and disable management services. The Content Distribution Managers are switched to halted, which is automatically saved in flash memory.

For more information on how to return halted Content Distribution Managers to an online status, see the *Cisco ACNS Software Configuration Guide for Centrally Managed Deployments*.

## Examples

The following example configures an IP address and a primary role for a Content Distribution Manager:

```
ContentDstributionManager(config)# cdm ip 10.1.1.1
ContentDstributionManager(config)# cdm role primary
```

The following example configures a new GUI port to access the Content Distribution Manager GUI:

```
ContentDstributionManager(config)# cdm ui port 8550
```

The following example configures the Content Distribution Manager as the standby Content Distribution Manager:

```
CDM(config)# cdm role standby
Switching CDM to standby will cause  all configuration settings made on this CDM
 to be lost.
Please confirm you want to continue  [no]?yes
Restarting CMS services
```

The following example configures the standby Content Distribution Manager with the IP address of the primary Content Distribution Manager by using the **cdm ip** *ip-address* global configuration command. This command associates the device with the primary Content Distribution Manager so that it can be approved as a part of the network.

```
CDM-4630# cdm ip 10.1.1.1
```

# cdnfs

To manage the ACNS network file system (cdnfs), use the **cdnfs** EXEC command.

**cdnfs** {**browse** | **cleanup** {**info** | **start** | **stop**} | **delete-unused-ecdnfs-files** | **lookup** *url*}

## Syntax Description

| | |
|---|---|
| **browse** | Browses the cdnfs directories and files. |
| **cleanup** | Cleans up the unwanted entries in the cdnfs. |
| **info** | Summarizes the information about unwanted entries without starting the cleanup process. |
| **start** | Starts the cleanup of unwanted entries in the cdnfs. |
| **stop** | Stops the cleanup of unwanted entries in the cdnfs. |
| **delete-unused-ecdnfs-files** | Deletes the unused ecdnfs legacy data files. |
| **lookup** | Performs a lookup of a specified URL in the cdnfs. |
| *url* | URL to look up. |

## Defaults

No default behavior or values

## Command Modes

EXEC

## Usage Guidelines

The ACNS network file systems (cdnfs) stores the pre-positioned ACNS network content to be delivered by all supported protocols. You can configure the cdnfs size of each Content Engine using the **disk configure** command.

The **cdnfs cleanup** command cleans up the content of deleted channels from the acquisition and distribution database. In certain cases, the acquirer is not notified by the Centralized Management System (CMS) about deleted channels, and it fails to clear all unified name space (UNS) content. In such cases, the **cdnfs cleanup** EXEC command can be used to clean up all UNS content associated with deleted channels.

**Note** You can use the **cdnfs cleanup start** to clean up the orphan content. The orphan content is content that is not associated with any channel to which a Content Engine is subscribed.

The **cdnfs browse** command is an interactive command and has the following subcommands used to view ACNS network files and directories:

```
ContentEngine# cdnfs browse

------  CDNFS interactive browsing  ------
dir, ls:   list directory contents
cd,chdir:  change current working directory
info:      display attributes of a file
more:      page through a file
cat:       display a file
exit,quit: quit CDNFS browse shell
```

```
/>dir
                    www.gidtest.com/
/>cd www.gidtest.com
/www.gidtest.com/>dir
764 Bytes           index.html
/www.gidtest.com/>info index.html

CDNFS File Attributes:
  Status                 3  (Ready)
  File Size              764 Bytes
  Start Time             null
  End Time               null
  Last-modified Time     Sun Sep  9 01:46:40 2001

Internal path to data file:
/disk06-00/d/www.gidtest.com/05/05d201b7ca6fdd41d491eaec7cfc6f14.0.data.html
  note: data file actual last-modified time: Tue Feb 15 00:47:35 2005

/www.gidtest.com/>
```

Because the cdnfs is empty in this example, the **ls** command does not show any results. Typically, if the cdnfs contained information, it would list the websites as directories, and file attributes and content could be viewed using these subcommands.

The **cdnfs cleanup** command synchronizes the state of the acquisition and distribution database with the content stored on the cdnfs. You should use this command after replacing a failed disk drive.

Use the **cdnfs delete-unused-ecdnfs-files** command to delete the previous version of the data files from previously released ACNS software ecdnfs files.

**Note** When you enter the **cdnfs delete-unused-ecdnfs-files** command, the previous versions of the reused files are not deleted; entering this command deletes the previous versions of the unused legacy data files only.

**Note** To migrate content from the ACNS E-CDN 4.x software to the ACNS 5.x software, first export your ACNS 4.x E-CDN content using the Content Distribution Manager, and then import it into the Content Distribution Manager running the ACNS 5.x software using the Content Distribution Manager GUI (create a manifest and channels, assign the Content Engine to channels, and so on). The cdnfs software that resides on the Content Engine reuses relevant legacy E-CDN application files.

Use the **cdnfs lookup** command to look up and, if present, obtain information on a specified URL in the cdnfs.

## Examples

The following example shows how to delete previous versions of the E-CDN application legacy files:

```
ContentEngine# delete-unused-ecdnfs-files
```

The following example shows the result of a lookup on a live streaming file. Typically, the File Size field is larger than zero. The Live Stream Route... information appears only for live streaming entries.

```
ContentEngine# cdnfs lookup rtsp://10.107.192.3/Soccer

CDNFS File Attributes:
  Status                 3  (Ready)
  File Size              0 Bytes
  Start Time             null
  End Time               null
  Allowed Playback via   HTTP WMT
  cdn_uns_id             d2CkEFiNwwaVNx+qI9KLeQ..
  channelId              131
  no_redirect_to_origin  1
  wmt-live               1

  Live Stream Route for WMT Media stream is :
-->Next Hop = 10.1.21.6
-->Next Hop = 10.107.150.203
-->Last Hop = 10.107.192.3
```

The Status field is displayed as Ready if the pre-positioned content is available on cdnfs, regardless of whether the content is available during playback or not.

The following example shows the output of the **cdnfs cleanup info** command:

```
ContentEngine# cdnfs cleanup info
Gathering cleanup information. This may take some time....
(Use Ctrl+C or 'cdnfs cleanup stop' to interrupt)
..............................

Summary of garbage resource entries found
-------------------------------------------
Number of entries    : 605
Size of entries (KB) : 60820911
```

## Related Commands

**show cdnfs**
**show statistics cdnfs**

# cdp (global configuration)

To configure Cisco Discovery Protocol (CDP) options, use the **cdp** global configuration command. To disable CDP on all interfaces, use the **no** form of this command.

**cdp** {**enable** | **holdtime** *seconds* | **timer** *seconds*}

**no cdp** {**enable** | **holdtime** *seconds* | **timer** *seconds*}

## Syntax Description

| | |
|---|---|
| **enable** | Enables CDP globally. |
| **holdtime** | Sets the length of time in seconds that a receiver keeps CDP packets before they are discarded. The default is 180 seconds. |
| *seconds* | Length of time that a receiver keeps the CDP packet in seconds (10–255). |
| **timer** | Sets the interval between the CDP advertisements in seconds. The default is 60 seconds. |
| *seconds* | Interval in seconds (5–254). |

## Defaults

CDP is enabled by default.

**holdtime**: 180 seconds

**timer**: 60 seconds

## Command Modes

global configuration

## Usage Guidelines

When enabled with the **cdp enable** command, CDP obtains protocol addresses of neighboring devices and discovers the platform of those devices. It also shows information about the interfaces used by your router. CDP is media- and protocol-independent and runs on Cisco-manufactured equipment.

Use of SNMP with the CDP Management Information Base (MIB) allows network management applications to learn the device type and the SNMP agent address of neighboring devices and to send SNMP queries to those devices. CDP uses the CISCO-CDP-MIB.

Each device configured for CDP sends periodic messages, known as advertisements, to a multicast address. The **cdp timer** *seconds* command specifies the rate at which CDP packets are sent. Each device advertises at least one address at which it can receive SNMP messages. The advertisements also contain Time-To-Live or hold-time information. To set the hold time, use the **cdp holdtime** *seconds* command to specify the period of time in seconds that a receiver is to keep CDP packets. Each device also listens to the periodic CDP messages sent by others to learn about neighboring devices.

## Examples

The following example shows that three command lines are entered in sequence. CDP is first enabled, the hold time is set to 10 seconds for keeping CDP packets, and then the rate at which CDP packets are sent (15 seconds) is set.

```
ContentEngine(config)# cdp enable
ContentEngine(config)# cdp holdtime 10
ContentEngine(config)# cdp timer 15
```

## Related Commands

**cdp** (interface configuration)
**clear cdp counters**
**clear cdp table**
**show cdp**

# cdp (interface configuration)

To enable Cisco Discovery Protocol (CDP) on an interface, use the **cdp** interface configuration command. To disable CDP on an interface, use the **no** form of this command.

**cdp enable**

**no cdp enable**

## Syntax Description

| | |
|---|---|
| **enable** | Enables CDP on an interface. |

## Defaults

No default behavior or values

## Command Modes

interface configuration

## Usage Guidelines

Using the **cdp enable** command in global configuration mode enables CDP globally on all the interfaces. If you want to control CDP behavior per interface, then use the **cdp enable** command in interface configuration mode. Note that the interface level control overrides the global control.

If CDP has been enabled on all interfaces, by default, you can use the **no cdp enable** command in interface configuration mode to disable CDP on a particular interface.

## Examples

The following example enables CDP on FastEthernet interface (slot 1/port 1):

```
ContentEngine# configure
ContentEngine(config)# interface FastEthernet 1/1
ContentEngine(config-if)# cdp enable
```

**Note** You must enable CDP globally on the Content Engine before you enable CDP on an interface. Otherwise, the following message is displayed in the command output:

```
CONTENTENGINE(config-if)# cdp enable
Cannot enable CDP on this interface, CDP Global is disabled
```

## Related Commands

**cdp** (global configuration)
**interface**
**show cdp**
**show interface**
**show running-config**
**show startup-config**

# cfs

To configure the cache file system (cfs) of the Content Engine, use the **cfs** EXEC command. To remove the cfs settings, use the **no** form of this command.

**cfs** {**clear** *partition* [**force**] | **format** *partition* | **mount** *partition* | **reset** *partition* | **sync** *partition* | **unmount** *partition*}

**no cfs** {**clear** *partition* [**force**] | **format** *partition* | **mount** *partition* | **reset** *partition* | **sync** *partition* | **unmount** *partition*}

## Syntax Description

| | |
|---|---|
| **clear** | Deletes the nonbusy objects from the specified cache file system (cfs) volume. |
| *partition* | Partition number (for example, disk00/00, disk00/01, disk01/00). |
| **force** | (Optional) Forcibly deletes all objects from the specified cfs volume. |
| **format** | Erases and formats or creates a file system for caching. |
| **mount** | Mounts a cache file system. |
| **reset** | Resets (unmounts-formats-mounts) a cache file system. |
| **sync** | Synchronizes a cache file system. |
| **unmount** | Unmounts a cache file system. |

## Defaults

No default behavior or values

## Command Modes

EXEC

## Usage Guidelines

Cache objects retrieved from the web are saved and manipulated with the cache file system (cfs) on a cfs partition of the hard disk. This process does not affect the sysfs, swfs, or mediafs partitions. The **cfs** commands are used to manage the cache object file system.

The **cfs clear** command deletes the nonbusy objects from the specified cfs volume. A nonbusy object is an object that is not being accessed (read or written). The **cfs clear** command (without force) deletes all possible objects without generating a broken GIF or HTML message to the client.

The **cfs clear force** command deletes all objects, busy or nonbusy, and may generate broken GIF or HTML messages for objects that were being read from the disk when the command was executed. If an object is being written to the Content Engine disk when a **cfs clear force** command is executed, the application stops caching that object but still delivers the object from the web server to the client.

The **cfs reset** command unmounts, formats, and mounts a specified volume. Unmounting a volume can result in broken GIF or HTML messages for objects that are being read from the disk (cache hits) when the command is executed. When a cfs volume is reset, all cfs data on that volume is lost.

**Note** You can enter the **cfs reset** command on unmounted volumes.

The **cfs format** command creates the cache file system internal dbs for the cfs partition of the disk if the volume is unmounted. It formats the cfs partition to prepare it for a cfs mount. The **cfs mount** command creates and maps data structures in memory to the cfs partition.

**Caution** All cached content is erased with the **format** option.

The **cfs unmount** command frees the in-memory data structures that map to the physical (disk) cfs partition.

The **cfs sync** command synchronizes the cache file system contents from the memory to the disk. Although synchronization is performed at regular intervals while the Content Engine is running, this command can be used to ensure that all data is written to disk before you reset or turn off the Content Engine. Synchronization can also be done with the **cache synchronize** command.

## Examples

The following example synchronizes the cache file system contents from the memory to disk05:

```
ContentEngine# cfs sync disk05
```

## Related Commands

**cache clear**
**clear cache**
**show cfs**

# channel

To assign, create, delete, add, modify, or otherwise configure a channel, use the **channel** EXEC command. This command is available on Content Distribution Manager devices only.

**channel assign** *site-name channel-name* {**channel-root** *root-ce-name* | **content-engine** {**all** | *ce-name*} | **device-group** {**all** | *devicegroup-name*}}

**channel create** *site-name channel-name* [**description** *channel-desc*] [**multicast-enabled**] [**priority** {**high** | **low** | **normal**}] [**skip-encryption**] [**weak-certificate**]

**channel delete** *site-name* {**all** | *channel-name*}

**channel manifest-add** *site-name channel-name manifest-url disk-quota check-manifest-interval* [**password** *password* **username** *username*]

**channel manifest-fetch** *site-name channel-name*

**channel manifest-modify** *site-name channel-name* [**disk-quota** *disk-quota*] [**manifest-url** *url*] [**password** *password*] [**time-to-live** *check-manifest-interval*] [**username** *username*]

**channel modify** *site-name channel-name* [**description** *channel-desc*] [**multicast** {**disable** | **enable**}] [**new-channel-name** *channel-name*] [**priority** {**high** | **low** | **normal**}] [**skip-encryption** {**disable** | **enable**}] [**weak-certificate** {**disable** | **enable**}]

**channel un-assign** *site-name channel-name* {**content-engine** {**all** | *ce-name*} | **device-group** {**all** | *devicegroup-name*}}

## Syntax Description

| | |
|---|---|
| **assign** | Assigns Content Engines or device groups to this website and channel. |
| *site-name* | Name of the website belonging to the content provider that is associated with this channel. |
| *channel-name* | Name of the channel to which Content Engines or device groups are to be assigned. |
| **channel-root** | Assigns root Content Engines to this channel. |
| *root-ce-name* | Name of the root Content Engine to be assigned to this channel. |
| **content-engine** | Assigns Content Engines to this channel. |
| **all** | Assigns every Content Engine to this channel. |
| *ce-name* | Name of the Content Engine to be assigned to this channel. |
| **device-group** | Assigns device groups to this channel. |
| **all** | Assigns every device group to this channel. |
| *devicegroup-name* | Name of the device group to be assigned to this channel. |
| **create** | Creates a newly named channel. |
| *site-name* | Name of the website that is associated with this channel. |
| *channel-name* | Name of the new channel. |
| **description** | (Optional) Describes the new channel. |
| *channel-desc* | Description of the new channel. |
| **multicast-enabled** | (Optional) Enables multicast networking on this new channel. |

| | |
|---|---|
| **priority** | (Optional) Sets the priority level on this new channel. |
| **high** | Sets the channel to high priority. |
| **low** | Sets the channel to low priority. |
| **normal** | Sets the channel to normal priority. |
| **skip-encryption** | (Optional) Omits encryption requirements for the validation certificate. Encryption for distribution of content (during replication of content from one Content Engine to another) is disabled. However, a checksum will still be calculated to ensure content integrity. |
| **weak-certificate** | (Optional) Enables weak encryption for the validation certificate. |
| **delete** | Deletes the named channels. |
| *site-name* | Name of the website that is associated with this channel. |
| **all** | Deletes every channel of the named website. |
| *channel-name* | Name of the channel to be deleted. |
| **manifest-add** | Adds the pre-positioned content described by the manifest file to this channel. |
| *site-name* | Name of the website that is associated with this channel. |
| *channel-name* | Name of the channel to which to add pre-positioned content. |
| *manifest-url* | URL where the manifest file is to reside. |
| *disk-quota* | Maximum disk space in megabytes allotted for the pre-positioned content retrieved by the manifest file. |
| *check-manifest-interval* | Frequency in minutes (0–52560000) with which the Content Engines assigned to the channel check for updates to the manifest file. (This setting is not the update interval for checking content.) |
| **password** | (Optional) Sets the password required to fetch the manifest file at its remote location. |
| *password* | Password required to fetch the manifest file at its remote location. |
| **username** | (Optional) Sets the username required to fetch the manifest file at its remote location. |
| *username* | Username required to fetch the manifest file at its remote location. The manifest username must be a valid ID. If the server allows an anonymous login, the user ID can be null. |
| **manifest-fetch** | Fetches the pre-positioned content described in the manifest file. |
| *site-name* | Name of the website from which to fetch content described by the manifest file. |
| *channel-name* | Name of the channel from which to fetch content described by the manifest file. |
| **manifest-modify** | Modifies the channel’s pre-positioned content and other parameters of the pre-positioned content described by the manifest file. |
| *site-name* | Name of the website to modify content described by the manifest file. |
| *channel-name* | Name of the channel to modify content described by the manifest file. |
| **disk-quota** | (Optional) Modifies the maximum disk space in megabytes allotted for the pre-positioned content retrieved by the manifest file. |
| *disk-quota* | New maximum disk space in megabytes allotted for the pre-positioned content retrieved by the manifest file. |
| **manifest-url** | (Optional) Modifies the URL where the manifest file is to reside. |

| | |
|---|---|
| *url* | New URL where the manifest file is to reside. |
| **password** | (Optional) Modifies the password required to fetch the manifest file at its remote location. |
| *password* | New password required to fetch the manifest file at its remote location. |
| **time-to-live** | (Optional) Modifies the Time To Live in minutes for the Content Engines (assigned to the channel) to check for updates to the manifest file. |
| *check-manifest-interval* | Frequency in minutes with which the Content Engines assigned to the channel check for updates to the manifest file. Valid values are 0–52560000. The update interval is the interval for the root Content Engine to check the manifest file itself. (This setting is not the update interval for checking content.) |
| **username** | (Optional) Modifies the username required to fetch the manifest file at its remote location. If the server allows an anonymous login, the user ID can be null. |
| *username* | New username required to fetch the manifest file at its remote location. |
| **modify** | Modifies the channel names. |
| **description** | (Optional) Modifies the channel description. |
| *channel-desc* | New description of the channel. |
| **multicast** | (Optional) Disables or enables multicast networking. |
| **disable** | Disables multicast networking. |
| **enable** | Enables multicast networking. |
| **new-channel-name** | (Optional) Modifies the channel name to a new name. |
| *channel-name* | New name of the channel. |
| **priority** | (Optional) Sets the priority level on this new channel. |
| **high** | Sets the channel to high priority. |
| **low** | Sets the channel to low priority. |
| **normal** | Sets the channel to normal priority. |
| **skip-encryption** | (Optional) Disables or enables the omission of encryption requirements for the validation certificate. |
| **disable** | Disables the omission of encryption. |
| **enable** | Enables the omission of encryption. |
| **weak-certificate** | (Optional) Disables or enables weak encryption for the validation certificate. |
| **disable** | Disables weak encryption. |
| **enable** | Enables weak encryption. |
| **un-assign** | Removes Content Engines or device groups from this assigned channel. |
| *site-name* | Name of the website that is associated with this channel. |
| *channel-name* | Name of the channel from which to remove assigned Content Engines or device groups. |
| **content-engine** | Removes Content Engines from this assigned channel. |
| **all** | Removes every Content Engine from this assigned channel. |
| *ce-name* | Name of the Content Engine to be unassigned from this channel. |
| **device-group** | Removes device groups from this assigned channel. |

| | |
|---|---|
| **all** | Removes every device group from this assigned channel. |
| *devicegroup-name* | Name of the device group to be removed from this channel. |

## Defaults

No default behavior or values

## Command Modes

EXEC

## Usage Guidelines

Channels map the content from a website to the devices in your ACNS network. Before you can create a channel, you must first create a directory of content providers and provide URLs to their websites. Content Engines in the same location can be assigned to different channels.

A website is a collection of content objects from a single origin server. Websites can be classified as either routable or nonroutable. Routable websites are controlled and operated by the enterprise corporation where the content is owned. Routable website domain names are fully qualified domain names (FQDNs) that are recognizable to Content Routers. Routable websites support all ACNS software routing and edge intercept mechanisms: WCCP interception, proxy configuration, and simplified hybrid routing. Nonroutable websites are not controlled or operated by the enterprise corporation, and the content is not owned by the enterprise. For example, www.cnn.com or www.yahoo.com are nonroutable websites. Nonroutable websites support WCCP interception and proxy configuration only.

**Note** You must have predefined websites using the Content Distribution Manager GUI.

When configuring a channel in the Content Distribution Manager GUI, the administrator specifies a list of Content Engines that belong to the channel and which replicate the content of the channel and a root Content Engine to acquire the content from the origin server and publish it.

For any given channel, there is only one publisher of content (the root Content Engine) and multiple receivers of that content (the Content Engines that are assigned to that channel). The location that contains the root Content Engine for a given channel is called the root location. A channel can have only one configured root Content Engine. Other Content Engines in the root location can act as backup publishers, if the configured root Content Engine fails.

### Disk Quota

The channel quota is disk quota or maximum content storage size in MB for pre-positioning content for this channel. When configuring the channel quota, follow these guidelines:

- The total of channel quota in all subscribed channels should not exceed the cdnfs disk space allocation of the Content Engine.
- The total used disk space in a channel should not exceed the amount of disk space that you allocated for the channel using the **channel manifest-add** *site-name channel-name url disk-quota* command.

  Because of the overhead, the amount of disk space used by a file is always larger than the size of the file itself. To figure the amount of disk space needed for a file, follow these steps:

**a.** Divide the actual file size in kilobytes (KB) by the file system block size, which is a fixed 4-KB (4096-byte) unit, and then round up the result to the nearest integer. This formula provides the number of filled and partially filled 4-KB blocks used by a file.

  (File size in KB / 4096) rounded up to the next integer value = Total number of blocks per file

**b.** Multiply the total number of file system blocks used by 4 KB (4096 bytes) to calculate the actual disk space consumed in bytes.

Total blocks per file * 4096 = Total disk usage in bytes

**c.** Multiply 4 KB by 4 and add the product to the total disk space consumed. (The integer 4 represents disk space that is reserved for internal system usage.)

Total disk usage in bytes + (4096 bytes * 4) = Disk usage per file

Also, because the software attempts to reserve enough space for other minor internal system functions, it is helpful to configure your channel quotas (and pre-positioned disk space) with a modest amount (perhaps 10 percent) of extra space beyond the total disk space consumed.

Channel quota in kilobytes = (Total disk usage in kilobytes) + (0.1 * Total disk usage in kilobytes)

### Distribution Priority

The distribution priority setting determines the priority of content acquisition and distribution. You configure this setting using the **priority** keyword. The distribution priority values are **high** (750), **normal** (500), or **low** (250).

The priority of content acquisition also depends on the origin server. Requests from different origin servers are processed in parallel. Requests from the same origin server are processed sequentially by their overall priority.

### Content Priority

A priority can be assigned to content objects to define their order of importance. The ACNS software determines the order of processing from the level of priority of the content. The higher the content priority, the sooner the acquisition of content from the origin server and the sooner the content is distributed to the Content Engines.

Every content object acquired by running a crawler job has the same priority.

Three factors combine to determine content priority:

- Channel priority— Specified using the **priority** keyword when you create or modify a channel. The channel priority is 250 for low, 500 for normal, and 750 for high.
- Item index—Content order listed in the manifest file.
- Item priority—Priority of the attributes specified by checking the High priority content check box or in the <item> or <crawler> tag.

To calculate the content priority, use one of the following formulas:

If there is a priority value for this content specified (either by using the **priority** keyword or in the manifest file *priority* attribute), use the following formula:

content priority = channel priority * 10000 + item priority

where the *item priority* can be any integer and is unrestricted.

**Note** If you want a particular content object to have the highest priority, specify a very large integer value for the item priority in the content priority formula.

If an object does not have a priority value specified in the manifest file *priority* attribute, use the following formula:

content priority = channel priority * 10000 + 10000 – item index

where the *item index* is the order in which the content is listed in the manifest file.

Note If there is no priority specified for any items, the content is processed in the order that is listed in the manifest file.

### Weak Certificate Verification

You can enable weak certificate verification for the manifest file. This setting is applicable when the manifest file is fetched using the HTTPS protocol.

Note To use weak certification for the channel content, you need to specify weak certification within the manifest file.

When you specify weak authentication within the manifest file by setting the *sslAuthType* attribute as weak, and if certain errors occur during certificate verification by the acquirer module, the content from that site will continue to be acquired. Possible errors are as follows:

- Unable to decode the issuer's public key
- Certificate has expired
- Self-signed certificate
- Self-signed certificate in certificate chain
- Unable to get local issuer certificate
- Subject issuer mismatch
- Authority and issuer serial number mismatch
- The root Content Engine is not marked as trusted
- Unable to verify the first certificate
- Certificate is not valid
- Certificate has invalid purpose

### Configuring Channel Options for Content Replication

The channel configuration offers various transmission options for replicating content; a channel can be configured for multicast and unicast (multicast with failover to unicast) or for unicast-only transmission.

When a channel is configured for multicast and unicast by specifying the **multicast-enabled** option, the receiver Content Engine uses unicast to download the content only after all carousel passes have been exhausted and after the preconfigured multicast transmission fails. In a multicast cloud configuration that uses a backup sender, when the channel is enabled for multicast and unicast, the failover to unicast occurs when the current active multicast sender has exhausted all the carousel passes for the file.

If the administrator wants the Content Engines to fall back to unicast (for example, with a multitier unicast deployment using a terrestrial multicast medium), the multicast cloud should be configured for a low number of carousel passes (such as 1, 2, or 3).

### Designating the Root Content Engine

A root Content Engine is used to acquire content for a channel. A channel can have only one root Content Engine. We recommend that you choose a root Content Engine that has enough bandwidth to access the content at the origin server.

The root Content Engine is the one Content Engine that is authorized to go directly to the origin web server for content. The root Content Engine then publishes the content to other Content Engines in the channel. You must designate a root Content Engine for content distribution to take place.

To designate one Content Engine to be the root Content Engine for a channel, use the **channel assign** *site-name channel-name* **channel-root** *root-ce-name* command.

**Note** You must create a channel and assign Content Engines or device groups to the channel before or at the same time as designating a Content Engine to be the root Content Engine.

### Adding and Removing Content Engines from Channels

To add a Content Engine to a channel, use the **channel assign** *site-name channel-name* **content-engine** *ce-name* command.

To perform a bulk addition of all Content Engines in various locations to the channel, use the **channel assign** *site-name channel-name* **content-engine all** command.

To remove a Content Engine from a channel, use the **channel un-assign** *site-name channel-name* **content-engine** *ce-name*.

If the Content Engine that you removed was the root Content Engine, and if there is at least one Content Engine still assigned to the channel, you must designate a new root Content Engine.

To perform a bulk removal of all Content Engines from the selected channel, use the **channel un-assign** *site-name channel-name* **content-engine all** command.

### Adding and Removing Device Groups from Channels

Device groups are assigned to channels using the **channel** EXEC command. Whenever a channel is created and additional device groups are added, or a channel assignment to the device group changes, devices in the group are notified of their assignment to the associated channel.

A many-to-many relationship exists between the device groups and the channels. A channel can have multiple device groups and device groups can belong to multiple channels.

You must have assigned Content Engines to a device group before you assign a device group to a channel.

To add a device group to a channel, use the **channel assign** *site-name channel-name* **device-group** *ce-name* command.

To perform a bulk addition of all device groups in various locations to the channel, use the **channel assign** *site-name channel-name* **device-group all** command.

To remove a device group from a channel, use the **channel un-assign** *site-name channel-name* **device-group** *ce-name*.

**Note** When a device group is removed from a channel, the Content Engines that were part of the device group are also removed from the channel. However, the Content Engines that are assigned directly to a channel continue to remain assigned. When a device is removed from a device group containing the original channel assignment, this device is also unassigned from channels. Similarly, when a channel is removed from a device group, the associated devices are also unassigned.

To perform a bulk removal of all device groups from the selected channel, use the **channel un-assign** *site-name channel-name* **device-group all** command.

### Manifest Files

The Cisco ACNS 5.x software manages the acquisition and distribution of the pre-positioned content through an Extensible Markup Language (XML)-based reference file called the manifest file. The manifest file lists the content that is to be used to populate Content Engines registered on a Cisco ACNS network. There should be one manifest file per channel.

The manifest file is placed on an origin server and identified by a unique URL. The location of the manifest file is specified when you enter the manifest file URL using the **channel manifest-add** command. The pre-positioned content is not stored on the Content Distribution Manager but is fetched from origin servers and distributed to Content Engines by a Content Engine that is a root Content Engine for the channel.

The Content Distribution Manager disseminates the manifest file URL to each of the root Content Engines on the ACNS network. The root Content Engine then parses the file and checks for any new or different information. After the root Content Engine determines what content is new, it fetches only that new content from the specified pre-positioned or live content from one or more origin servers.

The manifest file has the following features:

- Administrators and content providers can provide content on an origin server.
- Files can be imported over HTTP, HTTPS, or FTP while they are served using another streaming protocol based on a designated type of media playserver to play back the requested file.

Content acquisition and distribution can be controlled by setting prescheduled content availability dates and times. Two content acquisition methods can be configured within the manifest file. The first method specifies the acquisition of a single <item>. The second method specifies the content acquisition by crawling a website or FTP server with the <crawler> feature. Either of these two methods can schedule when the acquisition is to start and how often its content is to be checked for freshness.

If a proxy server is configured, requests to fetch the manifest file from the origin server will go through the proxy server. The proxy configuration applies only to manifest files and not to the content acquisition. To configure proxy server information for content acquisition, use the **acquirer proxy authentication** global configuration command.

Use the *ttl* option in the **channel manifest-add** command or the **time-to-live** *ttl* option in the **channel manifest-modify** command to specify the Time To Live in minutes of the pre-positioned content retrieved by the manifest file. Beyond the time interval specified by the TTL, the ACNS software checks to see if the manifest file has been updated, and the updated manifest file is downloaded and reparsed. Also, regardless of whether the manifest file has been updated, all content in the channel is rechecked and the updated content is downloaded. Use the *url* option in the **channel manifest-add** and **channel manifest-modify** commands to specify the address of the manifest file for the channel. The manifest URL must be a well-formed URL. If the protocol (FTP, HTTP, or HTTPS) for the URL is not specified, HTTP is used. If you do not specify a manifest file URL, this channel will have no content. For more information on protocols that are used by the ACNS software to acquire content, see the following sections.

### Using HTTP and HTTPS

Any standard web server supports the HTTP and HTTPS protocols. You can set up your web server as an origin server for the pre-positioned content intended for the ACNS network by moving the content over to the web server or by configuring the web server to access the desired content. The following two web servers are the most popular:

- Apache—Supported on UNIX, Linux, and Microsoft NT platforms
- Microsoft IIS—Supported only on Microsoft platforms

For the HTTP and HTTPS protocols, the content can be fetched as single content items by using the <item> tag in the manifest file, or the content can be fetched by using the crawling feature to crawl web server directories. The crawler crawls the folder hierarchy rather than parsing the HTML file. Therefore, if you want to use the crawl feature, you must enable directory indexing and make sure that the directory does not contain index.html, default.html, or home.html files.

**Tip** You might need to install SSL certificates to set up the web server for the HTTPS content acquisition. If your server is using an expired certificate, or a self-signed certificate, you should set sslAuthType to "weak" in the manifest file <host> tag.

### Using FTP

The root Content Engine acquirer supports acquiring files from FTP servers. When you use FTP, the content can be acquired as single content items by using the <item> tag in the manifest file, or the content can be fetched by using the crawling feature to crawl the FTP server directories. In FTP acquisition, the crawler crawls the folder hierarchy rather than parsing the HTML file. The following popular FTP servers are supported:

- Microsoft IIS 4.0, 5.0, 6.0—For Windows platforms
- Wu-2.6.1-18—For Linux platforms
- FTP Server—In SunOS (Version 5.6)
- proFTPd—For Linux platforms

Other supported Windows FTP servers are as follows:

- WS_FTP server
- Bulletproof FTP server
- SurgeFTP
- SlimFTPd

You can use other FTP servers, as long as the following FTP commands are supported:

- USER, PASS
- [SIZE, MDTM] [or] [LIST -a]
- PASV [or] PORT
- CWD ~ [or] CWD <SPACE> [or] CWD /
- RETR

### Secure FTP

The acquirer currently does not support secure FTP.

## Examples

The following example creates a channel se1, configures it for multicast and unicast, and enables weak certificate verification for the manifest file:

```
ContentDistributionManager# channel create southeast se1 description salesoffice
multicast-enabled weak-certificate
```

The following example assigns a root Content Engine to the channel se1:

```
ContentDistributionManager# channel assign southeast se1 channel-root sales
```

The following example shows the message that appears if there is no root Content Engine when you assign a Content Engine to the channel:

```
CDM# channel assign Website1 sample_channel content-engine CONTENTENGINE
Constraint: A root CE must be assigned for the channel.
```

The following example shows the message that appears if you unassign a root Content Engine from a channel that has been assigned with other Content Engines:

```
Mgt-CDM-507# channel un-assign agasweb2 dfg content-engine BACKUP-SENDER-53
Constraint: Cannot delete the root CE since there are other CE(s) assigned to the channel.
Select another root CE for the channel first, then do the deletion.
```

The following example displays the command output when you assign a root Content Engine to the channel:

```
CDM# channel assign Website1 sample_channel channel-root CONTENTENGINE
Operation completed successfully
```

The following example assigns a Content Engine CE507 to the channel sample_channel:

```
CDM# channel assign Website1 sample_channel content-engine CE507
Operation completed successfully
```

The following example assigns all Content Engines in the ACNS network to the channel sample_channel:

```
CDM# channel assign Website1 sample_channel content-engine all
Operation completed successfully
CDM#
```

The following example unassigns a Content Engine CE507 to the channel sample_channel:

```
CDM# channel un-assign Website1 sample_channel content-engine CE507
Operation completed successfully
```

The following example unassigns all Content Engines in the ACNS network to the channel sample_channel:

```
CDM# channel un-assign Website1 sample_channel content-engine all
Operation completed successfully
```

The following example shows the message that appears in the output of the command when you assign a device group to a channel before assigning Content Engines to the device group:

```
CDM# channel assign Website1 sample_channel device-group DG1
Constraint: Cannot assign an empty device group to a channel.
```

The following example assigns a device group DG2 to the channel sample_channel:

```
CDM# channel assign Website1 sample_channel device-group DG2
Operation completed successfully
```

The following example assigns all device groups in the ACNS network to the channel sample_channel:

```
CDM# channel assign Website1 sample_channel device-group all
Operation completed successfully
CDM#
```

The following example unassigns a device group DG1 to the channel sample_channel:

```
CDM# channel un-assign Website1 sample_channel device-group DG1
Operation completed successfully
```

The following example assigns all device groups in the ACNS network to the channel sample_channel:

```
CDM# channel un-assign Website1 sample_channel device-group all
Operation completed successfully
```

# channel-group

To add the current interface to an EtherChannel group, use the **channel-group** interface configuration command. To remove the interface from an EtherChannel group, use the **no** form of this command.

**channel-group** {**1** | **2**}

**no channel-group** {**1** | **2**}

## Syntax Description

| | |
|---|---|
| **1** | Interface that belongs to EtherChannel group 1. |
| **2** | Interface that belongs to EtherChannel group 2. |

## Defaults

No default behavior or values

## Command Modes

interface configuration

## Usage Guidelines

EtherChannel provides incremental trunk speeds between Fast Ethernet and Gigabit Ethernet or even at speeds greater than Gigabit Ethernet. EtherChannel combines multiple Fast Ethernet interfaces up to 400 Mbps or Gigabit Ethernet interfaces up to 2 Gbps. EtherChannel provides fault-tolerant, high-speed links between switches, routers, and servers.

EtherChannel for the ACNS 5.x software supports grouping of up to four same-speed network interfaces into one *virtual* interface. This grouping allows the addition or removal of a virtual interface that consists of two, three, or four Fast Ethernet or two Gigabit Ethernet interfaces; interoperability with Cisco routers, switches, and other networking devices or hosts supporting EtherChannel; and automatic failure detection and recovery based on each interface's current link status.

Use the **channel-group** command to add and remove the port-channel group ID number. The ID number is either 1 or 2. The **channel-group** and **ip address** commands add a physical Fast Ethernet port to a previously created Fast EtherChannel. The channel number is the same as the channel number specified when the **port-channel** interface command is used to create either a Fast Ethernet or a Gigabit Ethernet channel.

**Note** A channel cannot contain both Fast Ethernet and Gigabit Ethernet interfaces.

## Examples

The following example adds an interface to a channel group:

```
ContentEngine# config
ContentEngine(config)# interface fastEthernet 0/3
ContentEngine(config-if)# no ip address
ContentEngine(config-if)# channel-group 1
ContentEngine(config-if)# exit
```

The following example removes the group ID number from a channel group:

```
ContentEngine(config)# interface fastEthernet 0/3
ContentEngine(config-if)# no channel-group 1
ContentEngine(config-if)# exit
```

## Related Commands

**interface**
**port-channel**
**show interface**
**show running-config**
**show startup-config**

# clear

To clear the HTTP object cache, the hardware interface, statistics, archive working transaction logs, and other settings, use the **clear** EXEC command.

**clear bypass** {**counters** | **list**}

**clear cache** [**all** | **dns** [**domain** *domainname* | **hostname** *hostname*] | **http** [**url** *url*] | **media-real** | **wmt**]

**clear cdp** {**counters** | **table**}

**clear ip access-list counters** [*acl-num* | *acl-name*]

**clear logging**

**clear statistics** {**access-lists 300** | **all** | **authentication** | **bandwidth advanced errors** | **cifs-server** | **content-routing** | **distribution** {**all** | **mcast-data-receiver** | **mcast-data-sender** | **metadata-receiver** | **metadata-sender** | **unicast-data-receiver** | **unicast-data-sender**} | **dns-cache** | **ftp-native** | **ftp-over-http** | **history** | **http** {**all** | **cluster** | **ims** | **monitor** [**url** *url*] | **object** | **outgoing** | **proxy outgoing** | **requests** | **response** | **savings**} | **http-authcache** | **https** [**requests**] | **icap** | **icmp** | **icp** {**all** | **client** | **server**} | **ip** | **ldap** | **ntlm** | **pac-file-server** | **pre-load** | **radius** | **rtsp** {**proxy media-real** | **server cisco-streaming-engine**} | **rule** {**action** *action-type* | **all** | **rtsp**} | **running** | **tacacs** | **tcp** | **tftp** | **transaction-logs** | **tvout** | **udp** | **url-filter** {**http** {**local-list** | **N2H2** | **websense**} | **rtsp local-list** | **wmt local-list**} | **wmt**}

**clear transaction-log**

**clear users** {**administrative** | **request-authenticated**}

**clear wmt** {**incoming** | **outgoing** | **stream-id** *1-999999*}

## Syntax Description

| | |
|---|---|
| **bypass** | Clears the bypass commands. |
| **counters** | Clears all bypass counters. |
| **list** | Clears all bypass lists. |
| **cache** | Clears the HTTP objects from the cfs cache. |
| **all** | (Optional) Clears all cached objects. |
| **dns** | (Optional) Clears the cached DNS entries in the HTTP proxy. |
| **domain** | (Optional) Specifies the DNS cache domain name. |
| *domainname* | DNS cache domain name. |
| **hostname** | (Optional) Specifies the DNS cache hostname. |
| *hostname* | DNS cache hostname. |
| **http** | (Optional) Clears the HTTP objects cache. |
| **url** | (Optional) Clears the URL from the cfs cache. |
| *url* | HTTP or FTP URL. |
| **media-real** | (Optional) Clears the RealProxy cache content. |
| **wmt** | (Optional) Clears the WMT cache. |
| **cdp** | Resets the CDP statistical data. |

| | |
|---|---|
| **counters** | Clears the CDP counters. |
| **table** | Clears the CDP tables. |
| **ip access-list** | Clears the IP access list statistical information. |
| **counters** | Clears the IP access list counters. |
| *acl-name* | (Optional) Counters for the specified access list, identified using an alphanumeric identifier up to 30 characters, beginning with a letter. |
| *acl-num* | (Optional) Counters for the specified access list, identified using a numeric identifier (standard access list: 1–99; extended access list: 100–199). |
| **logging** | Clears the syslog messages saved in the disk file. |
| **statistics** | Clears the statistics as specified. |
| **access-lists** | Clears the access control list statistics. |
| **300** | Clears the group name-based access control list. |
| **all** | Clears all statistics. |
| **authentication** | Clears the authentication statistics. |
| **bandwidth** | Clears all bandwidth statistics. |
| **advanced** | Clears the advanced bandwidth statistics. |
| **errors** | Clears the advanced bandwidth error statistics. |
| **cifs-server** | Clears the CIFS server statistics. |
| **content-routing** | Clears all content routing statistics. |
| **distribution** | Clears the distribution statistics. |
| **all** | Clears the distribution statistics for every component. |
| **mcast-data-receiver** | Clears the distribution statistics for the multicast data receiver. |
| **mcast-data-sender** | Clears the distribution statistics for the multicast data sender. |
| **metadata-receiver** | Clears the distribution statistics for the metadata receiver. |
| **metadata-sender** | Clears the distribution statistics for the metadata sender. |
| **unicast-data-receiver** | Clears the distribution statistics for the unicast data receiver. |
| **unicast-data-sender** | Clears the distribution statistics for the unicast data sender. |
| **dns-cache** | Clears the DNS cache statistics. |
| **ftp-native** | Clears the native FTP caching statistics. |
| **ftp-over-http** | Clears the FTP-over-HTTP caching statistics. |
| **history** | Clears the statistics history. |
| **http** | Clears the cfs cache containing HTTP and FTP objects. |
| **all** | Clears all HTTP statistics. |
| **cluster** | Clears the healing mode statistics. |
| **ims** | Clears the HTTP if-modified-since (IMS) statistics. |
| **monitor** | Clears the statistics for all monitored URLs. |
| **url** | (Optional) Clears the statistics for a specific URL that has been monitored. |
| *url* | Monitored URL for which statistics are cleared. |
| **object** | Clears the HTTP object statistics. |
| **outgoing** | Clears the HTTP outgoing proxy statistics. |

| | |
|---|---|
| **proxy** | Clears the HTTP proxy-mode statistics. |
| **outgoing** | Clears the outgoing proxy monitor statistics. |
| **requests** | Clears the HTTP request statistics. |
| **response** | Clears the HTTP response statistics. |
| **savings** | Clears the HTTP savings statistics. |
| **http-authcache** | Clears the authentication cache statistics. |
| **https** | Clears the HTTPS statistics. |
| **requests** | (Optional) Clears the nontunneled HTTPS request statistics. |
| **icap** | Clears the ICAP statistics. |
| **icmp** | Clears the ICMP statistics. |
| **icp** | Clears the ICP statistics. |
| **all** | Clears all ICP statistics. |
| **client** | Clears the ICP client statistics. |
| **server** | Clears the ICP server statistics. |
| **ip** | Clears the IP statistics. |
| **ldap** | Clears the LDAP statistics. |
| **ntlm** | Clears the NTLM statistics. |
| **pac-file-server** | Clears the PAC file server statistics. |
| **pre-load** | Clears the preload statistics. |
| **radius** | Clears the RADIUS statistics. |
| **rtsp** | Clears the RTSP statistics. |
| **proxy** | Clears the RTSP-based proxy statistics. |
| **media-real** | Clears the RealMedia proxy statistics. |
| **server** | Clears the RTSP-based server statistics. |
| **cisco-streaming-engine** | Clears the Cisco Streaming Engine server statistics. |
| **rule** | Clears the rules statistics. |
| **action** | Clears the statistics of all the rules with the same action. |

| | |
|---|---|
| *action-type* | Specifies one of the following actions:<br>**allow**<br>**append-username-header**<br>**block**<br>**cache-non-cacheable**<br>**cache-only**<br>**dscp client cache-hit**<br>**dscp client cache-miss**<br>**dscp server**<br>**freshness-factor**<br>**insert-no-cache**<br>**no-auth**<br>**no-cache**<br>**no-persistent-connection**<br>**no-proxy**<br>**no-url-filtering**<br>**redirect**<br>**redirect-url-for-cdn**<br>**refresh**<br>**reset**<br>**rewrite**<br>**use-dns-server**<br>**use-icap-service**<br>**use-proxy**<br>**use-proxy failover**<br>**use-server**<br>**use-xforward-clt-ip**<br>See the “Actions” section for explanations of actions and patterns. |
| **all** | Clears the statistics of all the rules. |
| **rtsp** | Clears the statistics for the configured RTSP rules (rules configured for RTSP requests from RealMedia players [the RTSP rules] and rules configured for RTSP requests from Windows Media 9 players [the WMT-RTSP rules]). |
| **running** | Clears the running statistics. |
| **tacacs** | Clears the TACACS+ statistics. |
| **tcp** | Clears the TCP statistics. |
| **tftp** | Clears the TFTP statistics. |
| **transaction-logs** | Clears the transaction log export statistics. |
| **tvout** | Clears the TV-out statistics. |
| **udp** | Clears the UDP statistics. |
| **url-filter** | Clears the URL filter statistics. |
| **http** | Clears the URL filter for HTTP statistics. |
| **local-list** | Clears the local-list URL filter statistics. |
| **N2H2** | Clears the N2H2 URL filter statistics. |
| **websense** | Clears the Websense URL filter statistics. |
| **rtsp** | Clears the URL filter for Real-Time Streaming Protocol (RTSP) statistics. |

| | |
|---|---|
| **local-list** | Clears the local list URL filter for RTSP statistics. |
| **wmt** | Clears the URL filter Windows Media Technologies (WMT) statistics. |
| **local-list** | Clears the local list URL filter for WMT statistics. |
| **wmt** | Clears all WMT statistics. |
| **transaction-log** | Archives the working transaction log files. |
| **users** | Clears the connections (login) of authenticated users. |
| **administrative** | Clears the connections of administrative users authenticated through a remote login service. |
| **request-authenticated** | Clears the users authenticated by request. |
| **wmt** | Clears the WMT streams. |
| **incoming** | Clears all incoming WMT streams from the Content Engine. Also stops all of the Content Engine's WMT processes that are associated with the incoming WMT streams. |
| **outgoing** | Clears all outgoing WMT streams from the Content Engine. Also stops all of the Content Engine's WMT processes that are associated with the outgoing WMT streams. |
| **stream-id** | Clears the WMT streams that have the specified WMT stream ID. Also stops the Content Engine's WMT process that is associated with the specified stream ID. |
| *1-999999* | WMT stream ID to clear. |

## Defaults

No default behavior or values

## Command Modes

EXEC

## Usage Guidelines

The **clear cache** command removes all cached contents from the currently mounted cfs volumes. Objects being read or written are removed when they stop being busy. The equivalent to this command is the **cache clear** or **cfs clear** command.

**Caution** This command is irreversible, and all cached content will be erased.

The **clear cache force** command deletes all objects, whether busy or not, and may generate broken GIF or HTML messages for objects that were being read from the disk when the command was executed. If an object is being written to the Content Engine disk when a **clear cache force** command is executed, the application stops caching that object but still delivers the object from the web server to the client.

The **clear logging** command removes all current entries from the syslog.txt file, but does not make an archive of the file. It puts a "Syslog cleared" message in the syslog.txt file to indicate that the syslog has been cleared, as shown in the following example:

```
Feb 14 12:17:18 ContentEngine# exec_clear_logging:Syslog cleared
```

The **clear statistics** command clears all statistical counters from the parameters given. Use this command to monitor fresh statistical data for some or all features without losing cached objects or configurations.

The **clear transaction-log** command causes the transaction log to be archived immediately to the Content Engine hard disk. This command has the same effect as the **transaction-log force archive** command.

The **clear users administrative** command clears the connections for all administrative users who are authenticated through a remote login service, such as TACACS. This command does not affect an administrative user who is authenticated through the local database.

## Examples

The following example shows that the **clear bypass list** option purges all the entries in the bypass list:

```
ContentEngine# clear bypass list
```

The following example shows that the **clear transaction-log** option forces the working transaction log file to be archived:

```
ContentEngine# clear transaction-log
```

The following example shows that the **clear statistics http cluster** command resets the healing mode statistics:

```
ContentEngine(config)# clear statistics http cluster
```

## Related Commands

**cache clear**
**cfs clear**
**show interface**
**show statistics**
**show wccp**