



Viewing and Modifying TCP Stack Parameters on Standalone Content Engines

This chapter describes TCP stack parameters and how best to optimize them for caching purposes on standalone Content Engines. It explains how to view or modify TCP stack parameters, and contains the following sections:

- [TCP Stack Overview, page 20-2](#)
- [Viewing or Modifying TCP Parameters on Standalone Content Engines, page 20-2](#)
- [Displaying TCP Configuration Information, page 20-5](#)
- [TCP-Over-Satellite Extensions, page 20-6](#)



Note

For complete syntax and usage information for the CLI commands used in this chapter, see the *Cisco ACNS Software Command Reference, Release 5.5* publication.

TCP Stack Overview

Caches are typically deployed by customers for any of the following reasons:

- To save bandwidth
- To accelerate the delivery of content
- To apply policies that determine what content is viewed (content filtering)
- To increase the throughput of HTTP streams over TCP end to end

Another reason is to fine tune the TCP stack parameters to improve the performance of TCP end to end. Queries sent between a server and a client and the replies generated are defined as transactions. For data transactions between client and servers, the size of windows and buffers is important.

The relevant TCP parameters to maximize cache performance and throughput include the ability to tune timeout periods, client and server receive and send buffer sizes, and TCP window scaling behavior.



Note

Because of the complexities involved in TCP parameters, care is advised in tuning these parameters. In nearly all environments, the default TCP settings are adequate. Fine tuning of TCP settings is for network administrators with adequate experience and full understanding of TCP operation details.

Viewing or Modifying TCP Parameters on Standalone Content Engines

You can use the Content Engine GUI or CLI to view or modify TCP parameters on a standalone Content Engine.

- From the Content Engine GUI, choose **System > TCP**. Use the displayed TCP window to view or modify TCP parameters for this Content Engine. The existing TCP parameters are displayed in the TCP window. To modify a TCP parameter, change the value of a field and click **Update**. [Table 20-1](#) describes the fields in the TCP window and the related CLI command. For more information on the TCP window fields, click the **HELP** button in the window.
- From the Content Engine CLI, use the **tcp** global configuration command to modify the TCP parameters, as described in [Table 20-1](#).



Note

By default, the Content Engine does not automatically send out keepalives. To configure a standalone Content Engine to send out TCP keepalives on an HTTP connection, you must enter the **http tcp-keepalive enable** global configuration command. For more information on this topic, see the [Configuring Standalone Content Engines to Send out TCP Keepalives, page 7-67](#).

Table 20-1 TCP CLI Configuration Parameters

Window Fields	TCP CLI Commands	Descriptions
Send Buffer		
To Server	tcp server-send-buffer <i>kbytes</i>	Server send buffer size in kilobytes (that is, the TCP outgoing window size [1–512 KB]). The default is 8 KB.

Table 20-1 TCP CLI Configuration Parameters (continued)

Window Fields	TCP CLI Commands	Descriptions
To Client	tcp client-send-buffer <i>kbytes</i>	Client send buffer size in kilobytes (that is, the TCP outgoing window size [1–512 KB]). The default is 32 KB.
Receive Buffer		
To Server	tcp server-receive-buffer <i>kbytes</i>	Server receive buffer size in kilobytes (that is, the TCP incoming window size [1–512 KB]). The default is 32 KB.
To Client	tcp client-receive-buffer <i>kbytes</i>	Client receive buffer size in kilobytes (that is, the TCP incoming window size [1–512 KB]). The default is 8 KB.
R/W Timeout		
To Server	tcp server-rw-timeout <i>seconds</i>	Interval after which the Content Engine times out trying to read or write to the network (1–3600). The default is 120 seconds.
To Client	tcp client-rw-timeout <i>seconds</i>	Interval after which the Content Engine times out trying to read or write to the network (1–3600). The default is 120 seconds.
Keepalive		
Timeout	tcp keepalive-timeout <i>seconds</i>	Length of time that the Content Engine keeps a connection open before disconnecting.
Interval	tcp keepalive-probe-interval <i>seconds</i>	Length of time that the Content Engine keeps an idle connection open (1–3600 seconds). The default is 300 seconds.
Count	tcp keepalive-probe-cnt <i>count</i>	Number of times the Content Engine retries a connection (1–10 attempts). The default is 4 attempts.
Congestion Window base value	tcp cwnd-base <i>segments</i>	Initial congestion window value (1–10 segments). The default is 2 segments.
Initial Slow Start Threshold value	tcp init-ss-threshold <i>value</i>	Threshold for slow start (2–10 segments). The default is 2 segments.
Retransmit Timer Increment factor	tcp increase-xmit-timer-value <i>value</i>	Factor (1–3) used to modify the length of the retransmit timer by 1 to 3 times the base value determined by the TCP algorithm. The default is 1, which leaves the times unchanged. Note Modify this factor with caution. It can improve throughput when TCP is used over slow reliable connections but should never be changed in an unreliable packet delivery environment.
Maximum Segment Size		
To Server	tcp server-mss <i>maxsegsize</i>	Maximum packet size sent to servers. The default is 1460 bytes.
To Client	tcp client-mss <i>maxsegsize</i>	Maximum packet size sent to clients. The default is 1432 bytes
Others		
Satellite	tcp server-satellite tcp client-satellite	Server and client TCP compliance with RFC 1323. See the “TCP-Over-Satellite Extensions” section on page 20-6.
Type of Service	type-of-service <i>enable</i>	TCP Type of Service. The default is disabled.
Ecn	ecn <i>enable</i>	TCP explicit congestion notification.
TCP memory limits	tcp memory-limit	Configure TCP memory limits. See the next section, “Configuring TCP Memory Limits.”

Configuring TCP Memory Limits

In the ACNS 5.3.3 software and later releases, you can also use the CLI to configure TCP memory limits on a standalone Content Engine. The TCP memory limit settings allow you to control the amount of memory that can be used by the TCP subsystem's send and receive buffers.



Caution

Do not modify the default values unless you know what you are doing. The default values are device dependent and have been chosen after extensive testing. They should not be changed under normal conditions. Increasing these values can result in the TCP subsystem using more memory, which might cause the system to become responsive. Decreasing these values can result in increased response times and lower performance.

For Content Engines that are registered with a Content Distribution Manager, you can also configure TCP memory limits through the Content Distribution Manager GUI. For information about how to configure TCP memory limits for Content Engines that are registered with a Content Distribution Manager, see the *Cisco ACNS Software Configuration Guide for Centrally Managed Deployments, Release 5.5*.

To configure the TCP memory limit settings for a standalone Content Engine, use the **tcp memory-limit** global configuration command.

Table 20-2 lists the CLI command options that were added in the ACNS 5.3.3 software release.

Table 20-2 tcp memory-limit CLI Command Options

CLI Command Options	Function
low-water-mark <i>megabytes</i>	Specifies the TCP limit low-water mark. This value specifies the lower limit (in MB) of the memory pressure mode, below which TCP enters into the normal memory allocation mode. The range is 4–600.
high-water-mark-pressure <i>megabytes</i>	Specifies the TCP memory limit high-water mark-pressure. This value specifies the upper limit (in MB) of the normal memory allocation mode, beyond which TCP enters into the memory pressure mode. The range is 5–610.
high-water-mark-absolute <i>megabytes</i>	Specifies the TCP memory limit high-water mark-absolute. This value specifies the absolute limit (in MB) on TCP memory usage. The range is 6–620.

In this example, the low-water mark is set to 4 MB and the high-water-mark-pressure is set to 5 MB:

```
ContentEngine(config)# tcp memory-limit low-water-mark 4 high-water-mark-pressure 5
```

Table 20-3 describes the default values for each command parameter, which are based on the total amount of memory for the device.

Table 20-3 Default TCP Memory Limit Settings

Total System Memory	Low	Pressure	Absolute
1 GB, 2 GB, or 4 GB	360 MB	380 MB	400 MB
512 MB	180 MB	190 MB	200 MB
256 MB	25 MB	28 MB	30 MB

The following conditions must be satisfied whenever these default values are changed:

- The low water mark must be a number that is less than the high water mark pressure setting.
- The high water mark pressure must be a number that is less than the high water mark absolute setting:

```
low-water-mark < high-water-mark-pressure < high-water-mark-absolute
```

Displaying TCP Configuration Information

To display current TCP configuration information, use the **show tcp EXEC** command. The default 8 KB incoming window size for the client buffer is used here:

```
ContentEngine# show tcp
==TCP Configuration==
TCP keepalive timeout 300 sec
TCP keepalive probe count 4
TCP keepalive probe interval 75 sec
TCP server R/W timeout 120 sec
TCP client R/W timeout 120 sec
TCP server send buffer 8 k
TCP server receive buffer 32 k
TCP client send buffer 32 k
TCP client receive buffer 8 k
TCP server max segment size 1460
TCP satellite (RFC1323) disabled
TCP client max segment size 1432
TCP explicit congestion notification disabled
TCP type of service disabled
TCP cwnd base value 2
TCP initial slowstart threshold value 2
TCP memory_limit - Low water mark: 25 MB, High water mark (pressure): 28 MB,
High water mark (absolute): 30 MB
TCP increase(multiply) retransmit timer by 1
```

In this example, the **tcp client-receive-buffer** global configuration command is used to change the TCP incoming window size to 100 KB:

```
ContentEngine(config)# tcp client-receive-buffer 100
```

You can now verify the configuration change with the **show tcp EXEC** command.

```
ContentEngine# show tcp
==TCP Configuration==
TCP keepalive timeout 300 sec
TCP keepalive probe count 4
TCP keepalive probe interval 75 sec
TCP server R/W timeout 120 sec
TCP client R/W timeout 120 sec
TCP server send buffer 8 k
TCP server receive buffer 32 k
TCP client send buffer 32 k
TCP client receive buffer 100 k
TCP server max segment size 1460
TCP satellite (RFC1323) disabled
TCP client max segment size 1432
TCP explicit congestion notification disabled
TCP type of service disabled
TCP cwnd base value 2
TCP initial slowstart threshold value 2
TCP increase(multiply) retransmit timer by 1
TCP memory_limit - Low water mark: 25 MB, High water mark (pressure): 28 MB,
High water mark (absolute): 30 MB
```

TCP-Over-Satellite Extensions

The Content Engine has the ability to turn on TCP-over-satellite extensions (as documented in RFC 1323) to maximize performance and end-to-end throughput over satellite-type connections.

The large number of satellites available to network infrastructures has increased the amount of bandwidth available in the air. Taking advantage of these connections through satellite-type connections has created new challenges in the use of TCP transactions and acknowledgments:

- Latency—Round trip times to satellites orbiting 24,000 miles above the earth are 550 milliseconds for a single satellite hop. Window size must be set to prevent low-throughput connections.
- Bit errors—Packet loss can occur in a land-based device-to-satellite connection in addition to the losses caused by regular network congestion.
- Asymmetric bandwidth—Return bandwidth from satellites can be narrower than receiving bandwidth, which affects performance.

To set the TCP connection so that it complies with RFC 1323, use the **tcp server-satellite** and **tcp client-satellite** global configuration commands.