



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 19-1](#)
- [Viewing or Modifying TCP Parameters on Standalone Content Engines, page 19-2](#)
- [TCP-Over-Satellite Extensions, page 19-4](#)



Note

For complete syntax and usage information for the CLI commands used in this chapter, refer to the *Cisco ACNS Software Command Reference, Release 5.2* 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

The fourth reason is an often overlooked and much less understood property of deploying caching, and it can often have a huge benefit on 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, and fine-tuning the TCP stack parameters therefore becomes the key to maximizing this benefit.

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 19-1](#) 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 19-1](#).

Table 19-1 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.
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.

Table 19-1 TCP CLI Configuration Parameters (continued)

Window Fields	TCP CLI Commands	Descriptions
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>maxsegment</i>	Maximum packet size sent to servers. The default is 1460 bytes.
To Client	tcp client-mss <i>maxsegment</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 19-4.
Type of Service	type-of-service enable	TCP Type of Service. The default is disabled.
Ecn	ecn enable	TCP explicit congestion notification.

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

```
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 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-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, thereby affecting performance.

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