CHAPTER 22

Troubleshooting

This chapter describes how to troubleshoot locally managed deployments (standalone Content Engines). This chapter contains the following sections:

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.

For information about troubleshooting a Content Router, Content Distribution Manager, or a Content Engine that is registered with a Content Distribution Manager (as opposed to standalone Content Engines that are not registered with a Content Distribution Manager), see the *Cisco ACNS Software Configuration Guide for Centrally Managed Deployments, Release 5.5*.

Overview

Troubleshooting is a process of elimination, testing, and validation. It often requires you to use prior knowledge of network topology and business policies, apply protocol knowledge and application behavior, and work with network and application teams to resolve a problem.

Common problems can often be resolved by finding answers to questions such the following:

Problem	Troubleshooting Tool	Reference
• Is the Content Engine saving bandwidth? What is the HTTP performance?	**show statistics http savings** **show statistics http performance**	Troubleshooting with HTTP Statistics Tools, page 22-2
• Are all HTTP caching parameters enabled?	**debug http cache**	Troubleshooting with the Debug Tool, page 22-4
• What are the primary reasons for the cache miss?	**show statistics http miss-reason**	Troubleshooting with HTTP Statistics Tools, page 22-2
• What can you observe about the application settings?	**debug http header**	Troubleshooting with the Debug Tool, page 22-4
• Does a particular URL appear to be associated with performance issues?	**http monitor url**	Troubleshooting URL Performance, page 22-5
• Are client claims that certain URLs are not reachable valid?	**test-url**	Troubleshooting with the HTTP CLI Client, page 22-7

You can use the Web application troubleshooting tools described in this chapter to search out the root causes of many common problems.

Troubleshooting with HTTP Statistics Tools

Use the **show statistics http savings** EXEC command to verify HTTP bandwidth savings, as shown in the following example:

```
CE# show statistics http savings
                        Statistics - Savings
                       Requests                          Bytes
          ------------------------------------------------------------
  Total:             1559097021                  6421538629735
   Hits:             1041965581                  1890163375706
   Miss:              517131440                  4531375254029
Savings:                   66.8 %                        29.4 %
```

Use the **show statistics http performance** EXEC command to verify HTTP performance as shown in the following example:

```
CE# show statistics http performance
                           Statistics - Performance
                         Avg          Min             Max            Last
                  ------------------------------------------------------------
Requests / Second:         -            -             432             199
   Bytes / Second:         -            -        18992735          703016
Seconds / Request:     0.760        0.000     4291967.812           0.191
    Seconds / Hit:     0.083        0.000     4291367.331           0.030
   Seconds / Miss:     2.124        0.000     4291967.812           0.557
                  ------------------------------------------------------------
      Object Size:                  4118.8 Avg
                                       0 Min
                                2147484671 Max
                                  3530.7 Last
```

Use the **show statistics http miss-reason** EXEC command to show the reasons for a cache miss, as shown in the following example:

```
CE# show statistics http miss-reason
Statistics - No hit reasons
                                  Reason:             No. of Requests
                    ---------------------------------------------------
                             not_in_cache:              269562
                                dmbuf_low:                   0
                           none_get_method:                  0
                         ftp_not_anonymous:                  0
                        http_not_anonymous:                  0
                            suspicious_url:                  0
                                  ie_5_ims:                  0
                                has_if_match:                0
                       has_invalid_if_range:                 0
                    has_if_unmodified_since:                 0
                           has_invalid_range:                0
             has_more_than_supported_range:                  0
                      has_pragma_no_cache:                1260
                        has_authorization:                   0
                   has_cache_control_no_cache:               0
                               is_https:                     0
                           invalid_ims:                      0
                         cert_check_fail:                    0
                        second_validation:                   0
                        invalid_ims_reply:                   0
                            ims_200_reply:                   0
                          xfs_open_error:                  650
        has_unknown_length_transfer_pending:                 0
         object_in_cache_older_than_clients:                 0
    object_in_cache_expired_cannot_verified:                 0
                         different_protocol:                 0
                                other_error:               620

                        Statistics - Validate reasons
                                  Reason:             No. of Requests
                    ---------------------------------------------------
                                reval_all:                   0
                               reval_text:                   0
                                  max_age:                   0
                                min_fresh:                   0
                                max_stale:                   0
                           response_say_so:                  0
                             object_expired:                 0
                          reval_no_cache_req:                0
                                rule_refresh:                0

                        Statistics - No store reasons
                                  Reason:             No. of Requests
                    ---------------------------------------------------
                                dmbuf_low:                   0
                           none_get_method:                  0
                         ftp_not_anonymous:                  0
                        http_not_anonymous:                  0
                            suspicious_url:                  0
                                 has_range:                  0
                          has_authorization:                 0
                   has_cache_control_no_store:               0
                                invalid_ims:                 0
                              cert_check_fail:               0
                            second_validation:               0
                            invalid_ims_reply:               0
                                 url_too_long:               0
```

```
                    http_0_9_reply:                   0
                   header_too_long:                   0
          http_unknown_verion_reply:                   0
      http_none_cachable_reply_status:                 0
          http_unknow_reply_status:                    0
                        has_cookie:                270772
                   object_too_big:                     0
               has_pragma_no_cache:                   46
             cache_control_no_cache:                  46
             cache_control_no_store:                  46
              cache_control_private:                   0
                      has_multipart:                   0
                     invalid_expire:                   0
                invalid_last_modified:                 0
                       invalid_date:                   0
                  content_length_0:                    0
                           has_vary:                   0
                 transfer_encoding:                    0
                  three_to_two_way:                    0
                    xfs_open_error:                  650
has_unknown_length_transfer_pending:                   0
                         other_error:                620
            weird_server_pipe_though:                  0
            incorrect_content_length:                  0
                       rule_no_store:                  0
```

Increasing the Cache Hit Rate

Use the following global configuration commands for cache tuning and transport tuning to increase the cache hit rate:

- http cache-cookies
- http cache-cookies
- http cache-vary-user-agent enable
- http cache-chunk-encoded enable
- http smart-range enable
- http cache-authenticated {all | basic | ntlm}
- http fast-response enable (application specific, not always necessary; disables Nagle's algorithm)
- http tcp-keepalive enable (forces the Content Engine to send KAL probes so connections are not closed)

Troubleshooting with the Debug Tool

Use the **debug http** EXEC command to troubleshoot issues such as the following:

- When you are not sure what header parameters have been set by the applications, use the **debug http header** command.
- If the Content Engine is caching objects when it is expected not to cache them, use the **debug http hit** command.

- If the Content Engine is not caching objects when it is expected to cache them, use the **debug http miss** command.
- If you want to verify Content Engine requests or responses for caching parameters, use the **debug http cache** command.

Troubleshooting URL Performance

To configure a Content Engine to monitor the performance of specific URLs, use the **http monitor url** *url* global configuration command.

This command enables you to specify up to 10 URLs that you want the Content Engine to monitor and allows you to specify the time interval (in seconds) for monitoring the specified URL(s), as well as the acceptable delay (in seconds) before the URL is retrieved. The Content Engine maintains statistics about the various response characteristics for each of the monitored URLs. (You can use the new **show statistics http monitor** command to view these statistics, as described later in this section.)

```
ContentEngine(config)# http monitor url ?
  WORD  URL for monitoring
```

The **http monitor url** *url* command has two command options, the **acceptable-delay** and **interval** options. As the following sample output indicates, the **acceptable-delay** option is used to specify the acceptable delay in seconds (the maximum number of seconds that the specified monitored URL should be retrieved within). The default acceptable delay is 60 seconds.

```
ContentEngine(config)# http monitor url http://www.abc.com/ ?
  acceptable-delay  Threshold time in seconds before which the URL should be
  retrieved.(default is 60 seconds)
  interval          Interval in seconds for monitoring the URL.(default is 60 seconds)
  <cr>
```

As the following sample command output indicates, the **acceptable-delay** option is used to specify the acceptable delay, which is the maximum number of seconds that the specified URL should be retrieved within.

```
ContentEngine(config)# http monitor url http://www.abc.com/ acceptable-delay ?
  <1-3600>  Acceptable delay in seconds
```

Note If you use the **http monitor url** *url* command to configure the same URL with a different interval or acceptable-delay setting, the most recently configured setting takes precedence and overrides any previously configured settings for that particular URL.

As the following sample command output indicates, the **interval** option specifies the monitoring interval (that is, how frequently the Content Engine should monitor requests for a specific URL). The monitoring interval is specified in seconds. The default monitoring interval is 60 seconds.

```
ContentEngine(config)# http monitor url http://www.abc.com/ acceptable-delay 100
interval ?
  <1-3600>  Monitor interval in seconds
```

In the following example, the Content Engine is configured to monitor the URL named "http://www.abc.com/" using the default values (an interval of 60 seconds and an acceptable delay of 60 seconds):

```
http monitor url http://www.abccorp.com/
```

In the following example, the Content Engine is configured to monitor the URL named "http://www.abc.com/." The Content Engine is configured to wait up to 100 seconds for the URL to be retrieved and to monitor requests for this URL every 100 seconds.

```
ContentEngine(config)# http monitor url http://www.abc.com/ acceptable-delay 100
interval 100
```

If it takes more than 100 seconds for the URL to be retrieved, the specified acceptable delay is exceeded. The Content Engine tracks the response time (minimum and maximum delay time) as well as the number of times that the acceptable delay is exceeded for a particular URL. These statistics are shown in the output from the new **show statistics http monitor** EXEC command. (An example of the output from the **show statistics http monitor** EXEC command is provided below.)

To display statistics for the monitored URLs, use the **statistics http monitor** EXEC command. The following example shows the statistics that are reported for each of the monitored URLs:

```
ContentEngine# show statistics http monitor
HTTP Monitor URL statistics
---------------------------

Monitor URL                              = http://www.abc.com/
Total requests                           = 118
Failed requests                          = 30
Requests above acceptable delay          = 37
Minimum response time                    = 8.183 seconds
Maximum response time                    = 210.021 seconds

Monitor URL                              = http://www.abccorp.com/
Total requests                           = 275
Failed requests                          = 44
Requests above acceptable delay          = 26
Minimum response time                    = 0.071 seconds
Maximum response time                    = 164.061 seconds
```

In the command output shown above, note the following information:

- "Failed requests" are requests that did not succeed (for example, the request failed to resolve the domain name of that URL).
- "Requests above acceptable delay" are the requests that took longer than the specified acceptable delay (the maximum number of seconds specified by the acceptable-delay setting).

The output of the **show running-configuration** EXEC command includes information about the URL monitoring configuration. In the following excerpt from the **show running-configuration** command output, this particular information is highlighted in bold:

```
ContentEngine# show running-configuration
!
!
hostname sust-7320-ce1
!
http persistent-connections timeout 300
http proxy incoming 8080
http proxy outgoing preserve-407
http tcp-keepalive enable
http monitor url http://www.abc.com/ interval 100 acceptable-delay 100
http monitor url http://www.abccorp.com/
!
ftp proxy incoming 8080
!
clock timezone US/Eastern -5 0
.
.
.
```

Only the non-default values are displayed in the output from the **show running-configuration** command. Consequently, because the Content Engine was configured to use the default values to monitor the URL "http://www.abccorp.com," the above sample output does not display these values for that URL.

To display a list of monitored URLs, including the interval and acceptable delay setting for each monitored URL, use the **show http monitor** EXEC command:

```
ContentEngine# show http monitor

Monitor URL: http://www.abc.com/
Monitor Interval: 100
Acceptable Delay: 100

Monitor URL: http://www.abccorp.com/
Monitor Interval: 60
Acceptable Delay: 60
```

Troubleshooting with the HTTP CLI Client

In the ACNS 5.2 software and later releases, an HTTP CLI client is supported. This capability allows you to test connectivity and debug caching issues. The **test-url** EXEC command provides debugging capabilities to the end user from the Content Engine CLI. With this command, you can test the URL accessibility and connectivity through the web server and the proxy server. The functionality of the URL over the HTTP, HTTPS, and FTP protocols can be tested. This command tests the functionality of a URL over the HTTP and FTP proxy servers. The following command output is displayed to the standard output (the console):

```
ContentEngine# test-url ?
  ftp    For request over FTP
  http   For request over HTTP
  https  For request over HTTPS
```

Table 22-1 describes the **test-url** commands.

Table 22-1 test-url CLI Commands

test-url Content Engine CLI Commands	Description

test-url http *test-URL*	Tests the connectivity of the specified URL (the test URL). Tests the connectivity of the URL (content request over HTTP) through the HTTP proxy server (the Content Engine). Also tests HTTP request authentication, proxy authentication, HTTP HEAD request, and a custom header.
	`ContentEngine# test-url http ?`
	`custom-header Custom header to send to server`
	`head-only To fetch only header information`
	`use-http-proxy Test a url through a proxy. This option can be used only with http protocol`

Table 22-1 test-url CLI Commands (continued)

test-url Content Engine CLI Commands	Description

test-url https *test-URL*	Tests the connectivity of the specified URL (content request over HTTPS). Also tests HTTPS request authentication and the HTTPS HEAD request.
	`ContentEngine# test-url https ?`
	`WORD The test url. Format https://domain.com/path or https://user:password@domain.com/path`

test-url ftp *test-URL*	Tests the connectivity of the specified URL (test URL). Tests the connectivity of the URL (content request over FTP [FTP-over-HTTP]) through the FTP proxy server (Content Engine). Also tests the HTTP authentication for FTP-over-HTTP requests, and proxy authentication.
	`ContentEngine# test-url ftp ?`
	`WORD The test url. Format ftp://domain.com/path or ftp://user:password@domain.com/path`
	`ContentEngine# test-url ftp use-ftp-proxy ?`
	`use-ftp-proxy Test a url through a proxy. This option can be used only with http protocol`

When using the **test-url** EXEC command, you can specify the test URL (**test-url** *protocol test-URL* EXEC command) in either of the following formats:

http://*domain.com/path* or **http://***user***:***password***@***domain.com/path*

In the following example, the test URL (www.cisco.com) is an HTTP request, and the request succeeds:

```
ContentEngine# test-url http http://www.cisco.com
--17:34:50--  http://www.cisco.com/=> `/dev/null'
Len - 21 , Restval - 0 , contlen - 0 , Res - 134728056Resolving www.cisco.com
done.
Connecting to www.cisco.com[192.168.0.0]:80... connected.
HTTP request sent, awaiting response...
 1 HTTP/1.1 200 OK
 2 Date: Fri, 25 Jun 2006 17:09:06 GMT
 3 Server: Apache/1.0 (Unix)
 4 Content-Type: text/html
 5 Set-Cookie: CP_GUTC=172.16.0.0.133781088183346835; path=/; expires=T
19-Jun-29 17:09:06 GMT; domain=.cisco.com
 6 Via: 1.1 Application and Content Networking System Software 5.5.1
 7 Connection: Close
 (-1 to go)
    [<=>                                  ] 0             --.--K/s
en - 1185        ELen - 0        Keepalive - 0
    [ <=>                                 ] 56,249         1.38M/s

17:34:50 (1.38 MB/s) - `/dev/null' saved [56249]
```

In the following example, the test URL (gmail.google.com) is an HTTPS request that fails because the host is not found:

```
ContentEngine# test-url https https://gmail.google.com head-only
--17:43:13--  https://gmail.google.com/=> `/dev/null'
Len - 25 , Restval - 0 , contlen - 0 , Res - 134728056Resolving gmail.google.com
...
Resolving gmail.google.com failed: Host not found.
```

In the following example, the test URL is an FTP-over-HTTP request using the Content Engine as the FTP proxy for the request:

```
ContentEngine# test-url ftp ftp://1234567:pAB6rB7x@smartfilter.com  use-ftp-proxy ?
     WORD  Proxy-url. Format http://proxyIp:proxyPort or
     http://proxyUser:proxypasswd@proxyIp:proxyPort
ContentEngine# test-url ftp ftp://1234567:pAB6rB7x@smartfilter.com use-ftp-proxy
172.16.0.0:8080
```

Troubleshooting in Real Time

To obtain traffic output in real time, use the **type-tail** EXEC command with the **follow** option. This command can be used to follow any working.log or syslog.txt files. You can also use the **tcpdump** EXEC command to capture all of the traffic on a Content Engine interface.

The following example shows real-time output from the working log:

```
CE# type-tail working.log follow
```

```
1143265877.603 18 10.82.163.18 TCP_IMS_HIT/304 148 GET http://www.cisco.com/swa/i/flyout_arrow_red.gif -
NONE/- - ALLOW "-"
1143265877.683 29 10.82.163.18 TCP_IMS_HIT/304 149 GET http://www.cisco.com/swa/i/spacer_grey.gif - NONE/- -
ALLOW "-"
1143265877.683 29 10.82.163.18 TCP_IMS_HIT/304 148 GET http://www.cisco.com/swa/i/flyout_arrow_red.gif -
NONE/- - ALLOW "-"
1143265877.690 6 10.82.163.18 TCP_IMS_HIT/304 149 GET http://www.cisco.com/swa/i/spacer_grey.gif - NONE/- -
ALLOW "-"
1143265877.792 45 10.82.163.18 TCP_IMS_HIT/304 148 GET http://www.cisco.com/swa/i/flyout_arrow_red.gif -
NONE/- - ALLOW "-"
1143265877.793 45 10.82.163.18 TCP_IMS_HIT/304 149 GET http://www.cisco.com/swa/i/spacer_grey.gif - NONE/- -
ALLOW "-"
1143265877.798 5 10.82.163.18 TCP_IMS_HIT/304 148 GET http://www.cisco.com/swa/i/flyout_arrow_red.gif -
NONE/- - ALLOW "- "
1143265877.802 239 10.82.163.18 TCP_MISS/200 238 GET
http://www.cisco.com/swa/j/zag2_vs_log1.asc?Log=1&title=Cisco%20Systems,%20Inc&
basepage=http://www.cisco.com/&cb=1143243947972 - DIRECT/www.cisco.com - ALLOW "-"
1143265877.901 44 10.82.163.18 TCP_IMS_HIT/304 149 GET http://www.cisco.com/swa/i/spacer_grey.gif - NONE/- -
ALLOW "-"
1143265877.901 44 10.82.163.18 TCP_IMS_HIT/304 148 GET http://www.cisco.com/swa/i/flyout_arrow_red.gif -
NONE/- - ALLOW "-"
1143265914.190 123 10.82.163.18 TCP_MISS/200 3760 GET http://www.google.com/ - DIRECT/www.google.com - ALLOW
"-"
1143265914.881 89 10.82.163.18 TCP_REFRESH_HIT/304 8794 GET http://www.google.com/intl/en/images/logo.gif -
DIRECT/www.google.com - ALLOW "-"
1143265930.721 103 10.82.163.18 TCP_MISS/200 3760 GET http://www.google.com/ - DIRECT/www.google.com - ALLOW
"-"
1143265930.957 37 10.82.163.18 TCP_IMS_HIT/304 176 GET http://www.google.com/intl/en/images/logo.gif - NONE/-
- ALLOW "-"
```

Troubleshooting with the Log Viewing Tool

In the ACNS 5.2 software and later releases, you can use the **less** EXEC command to specify the filename of the ACNS software log that you want to view:

```
ContentEngine# less ?
  WORD  file name
```

Troubleshooting with the Telnet Client

In the ACNS 5.2.1 software and later releases, a Telnet CLI client is supported. This Telnet client allows you to specify a destination port. This allows you to test websites by attempting to connect with Telnet to the website from the Content Engine CLI. To support this feature, the **telnet** EXEC command was added.

```
ContentEngine# telnet ?
  Hostname or A.B.C.D  Remote host or IP address
```

Troubleshooting with Ping

To send echo packets for diagnosing basic network connectivity on networks, use the **ping** EXEC command:

ping {*hostname* | *ip-address*} | *PING options*

In the ACNS 5.2.1 software and later releases, the functionality of the **ping** EXEC command supports all of the standard ping options as command arguments. (For more information on the standard ping options, see the Linux man page.)

```
ContentEngine# ping ?
  LINE  Destination Host or IP Address (or PING options)
```

To use the **ping** EXEC command with the *hostname* argument, be sure that DNS functionality is configured on your Content Engine. To force the timeout of a nonresponsive host, or to eliminate a loop cycle, press **Ctrl-C.**

In the following example, the host with an IP address of 172.19.131.189 is pinged from the Content Engine CLI:

```
ContentEngine# ping 172.19.131.189
PING 172.19.131.189 (172.19.131.189) from 10.1.1.21 : 56(84) bytes of data.
64 bytes from 172.19.131.189: icmp_seq=0 ttl=249 time=613 usec
64 bytes from 172.19.131.189: icmp_seq=1 ttl=249 time=485 usec
64 bytes from 172.19.131.189: icmp_seq=2 ttl=249 time=494 usec
64 bytes from 172.19.131.189: icmp_seq=3 ttl=249 time=510 usec
64 bytes from 172.19.131.189: icmp_seq=4 ttl=249 time=493 usec

--- 172.19.131.189 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.485/0.519/0.613/0.047 ms
```

In the ACNS 5.2.1 software and later releases, the functionality of the **ping** command supports all of the standard ping options. (For more information on the standard ping options, see the Linux man page.)

```
ContentEngine# ping ?
  LINE  Destination Host or IP Address (or PING options)
```

Troubleshooting with Traceroute

Traceroute is a widely available utility on most operating systems. Much like ping, it is a valuable tool for determining connectivity in a network. Ping allows the user to find out if there is a connection between two end systems. Traceroute does this as well, but additionally lists the intermediate routers between the two systems. Users can therefore see the routes that packets can take from one system to another.

To find the route to a remote host, when either the hostname or IP address is known, use the **traceroute** EXEC command.

```
ContentEngine# traceroute yahoo.com
traceroute to 66.218.71.113 (66.218.71.113), 30 hops max, 38 byte packets
***
***
***
***
10  p3-3.paloalto-cr2.bbnplanet.net (4.0.26.13)  3.219 ms  2.001 ms  2.097 ms
11  p7-1.paloalto-nbr2.bbnplanet.net (4.0.6.77)  3.133 ms  1.949 ms  2.076 ms
12  p4-0.paloalto-nbr1.bbnplanet.net (4.0.5.65)  2.755 ms  2.204 ms  2.037 ms
13  p1-0.paix-bi2.bbnplanet.net (4.0.6.98)  2.928 ms  2.146 ms  2.334 ms
14  p1-0.xpaix17-level3.bbnplanet.net (4.0.1.70)  3.397 ms  3.631 ms  3.081 ms
15  gige10-0.ipcolo4.SanJose1.Level3.net (64.159.2.42)  3.334 ms  2.999 ms  2.388 ms
16  cust-int.level3.net (64.152.69.18)  3.871 ms  3.031 ms *
17  ge-3-3-0.msr1.pao.yahoo.com (192.168.101.42)  3.695 ms ge-2-3-0.msr2.pao.yahoo.com
    (192.168.101.46)  6.998 ms *
18  vl16.bas1.scd.yahoo.com (66.218.64.146)  6.282 ms  5.091 ms  5.162 ms
19  w2.rc.scd.yahoo.com (66.218.71.113)  6.028 ms  5.782 ms  5.544 ms
```

Troubleshooting with the WMT Diagnostic Tools

In the ACNS 5.3.1 software and later releases, you can access the following three WMT diagnostic tools through the Content Engine CLI:

- asfhead—Examines the headers of a Windows Media file (for example, an .asf, .wmv, or .wma file).

  To access the asfhead tool, enter the **show wmt diagnostics header-info stream-file** EXEC command, as follows:

  ```
  ContentEngine# show wmt diagnostics header-info stream-file word
  ```

  where *word* is a local file.

- nschead—Examines the .nsc file headers.

  The .nsc file is created when a multicast station is configured. This tool is useful for multicast debugging. To access the nschead tool, enter the **show wmt diagnostics header-info nsc-file** EXEC file, as follows:

  ```
  ContentEngine# show wmt diagnostics header-info nsc-file .nsc-filename
  ```

The following is a sample annotated output from the **show wmt diagnostics header-info stream-file** EXEC command. In this example, this command is used to display the headers of a .wmv file named 256.wmv. Annotations about this sample command output are highlighted in boldface text and enclosed within parentheses.

```
ContentEngine# show wmt diagnostics header-info stream-file 256.wmv
Description object:
VBR info object  (2 streams)  (VBR: variable bit rate file)
streamid 1 bitrate 34319 (0000860f)  (in this file there are 2 bit rates)
streamid 2 bitrate 195798 (0002fcd6)
Global properties object:
               GUID = f9 dd df 7e 6c 80 85 46 83 3c c1 23 4b 2f 4f 6b  (object ID)
          File Size = 8104309  (object size)
       Total Packets = 2816  (total number of data packets)
           Time_mkin = 0x0000000001c0debc
          Time_mkout = 0x00000000a9e1af60
       Send duration = 2817510000  (Specifies the time needed to send the file
                                    in 100-nanosecond units)
     Broadcast Flag = 0
```

```
       Seekable Flag = 1
 Minimum Packet Size = 2877  (Specifies the minimum data packet size in bytes)
      Max Packet Size = 2877  (Specifies the maximum data packet size in bytes)
     Max Bitrate = 230117  (Specifies the maximum instantaneous bit rate in
                              bits per second for the entire file)
Unknown object:
uuid: b5 3 bf 5f 2e a9 cf 11 8e e3 0 c0 c 20 53 65 len: 0x16
00000000: 11 d2 d3 ab ba a9 cf 11 8e e6 00 c0 0c 20 53 65
00000010: 06 00 00 00 00 00
Extended Content Description Object (two entries):
WMFSDKVersion (unicode) 7.00.00.1956
WMFSDKNeeded (unicode)  0.0.0.0000
Codec description object (2 streams)  (audio and video codec information)
                Type = 0x2
         Description = Windows Media Audio V7
                        32 kbps, 32 kHz, stereo
                  ID = 61 01
                Type = 0x1
         Description = Windows Media Video V7

                  ID = 57 4d 56 31
Index:
               Index #0
            Interval = 10000000
            Max_pkt# = 9
               Entry = 286
           Stream ID = 0
```

The following is an excerpt of sample output from the **show wmt diagnostics header-info nsc-file** EXEC command. In this example, this command is used to display the headers of the .nsc file named testmcast.nsc. Annotations about this sample command output are highlighted in boldface text and enclosed within parentheses.

```
ContentEngine# show wmt diagnostics header-info nsc-file testmcast.nsc
Press Ctrl-C to abort, if no information is shown within 30 secs.

[Address]
Time To Live=0x00000005  (ttl)
IP Address=233.33.33.33  (Multicast address that is configured.)
IP Port=0x00000D10  (Multicast port that is configured.)
Delivery Mode=0x00000002
NSC Format Version=3.0  (version of NSC format)
[Description]
Description=
Auto Archive=0x00000000
Format1 id 4d2
Description1=http://192.168.192.4:8080  (Source of the media stream,
for example, the WMT server/encoder)

header information:

output_head() starting
Description object:
clip: WMT Live testing (title of the stream, configured
on the WMT server/encoder)
author:         John Doe  (author of the stream,
configured on the WMT server/encoder)
VBR info object  (2 streams)
streamid 1 bitrate 34995  (000088b3)
streamid 2 bitrate 253711  (0003df0f)
Global properties object:
                GUID = 33 36 59 e6 6a 7d 4e 49 99 5e cf 71 39 b9 61 54
              (guid: unique identifier of this stream)
```

```
           File Size = 2810  (size in bytes)
       Total Packets = 4294967295
           Time_mkin = 0x0000000001c4de45
          Time_mkout = 0x0000000000000000
       Send duration = 0
      Broadcast Flag = 1
       Seekable Flag = 0
 Minimum Packet Size = 1444
     Max Packet Size = 1444
         Max Bitrate = 288706
Unknown object:
uuid: b5 3 bf 5f 2e a9 cf 11 8e e3 0 c0 c 20 53 65 len: 0x707
00000000: 11 d2 d3 ab ba a9 cf 11 8e e6 00 c0 0c 20 53 65
00000010: 06 00 f1 06 00 00 a9 46 43 7c e0 ef fc 4b b2 29
00000020: 39 3e de 41 5c 85 27 00 00 00 00 00 00 00 01 00
00000030: 0c 65 00 6e 00 2d 00 75 00 73 00 00 00 cb a5 e6
00000040: 14 72 c6 32 43 83 99 a9 69 52 06 5b 5a 58 00 00
00000050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000060: 00 00 00 00 00 00 7d 00 00 88 13 00 00 00 00 00
00000070: 00 00 7d 00 00 88 13 00 00 00 00 00 00 00 03 00
00000080: 00 00 00 00 00 01 00 00 00 00 00 00 00 00 00 00
00000090: 00 00 00 00 00 cb a5 e6 14 72 c6 32 43 83 99 a9
000000a0: 69 52 06 5b 5a 6e 00 00 00 00 00 00 00 00 00 00
000000b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 68 ad 03
000000c0: 00 88 13 00 00 00 00 00 00 68 ad 03 00 88 13 00
000000d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 02 00 00
000000e0: 00 63 17 05 00 00 00 00 00 00 00 01 00 50 94 bd
000000f0: c6 7f 86 07 49 83 a3 c7 79 21 b7 33 ad 02 00 00
00000100: 00 00 00 5d 8b f1 26 84 45 ec 47 9f 5f 0e 65 1f
00000120: c5 af 5b 77 48 84 67 aa 8c 44 fa 4c ca e0 00 00
00000130: 00 00 00 00 00 04 00 00 00 01 00 0c 00 02 00 02
00000140: 00 00 00 49 00 73 00 56 00 42 00 52 00 00 00 00
00000150: 00 00 00 01 00 34 00 00 00 06 00 00 00 44 00 65
00000160: 00 76 00 69 00 63 00 65 00 43 00 6f 00 6e 00 66
00000170: 00 6f 00 72 00 6d 00 61 00 6e 00 63 00 65 00 54
00000180: 00 65 00 6d 00 70 00 6c 00 61 00 74 00 65 00 00
00000190: 00 4c 00 32 00 00 00 00 00 02 00 0c 00 02 00 02
000001a0: 00 00 00 49 00 73 00 56 00 42 00 52 00 00 00 00
000001b0: 00 00 00 02 00 34 00 00 00 0c 00 00 00 44 00 65
000001c0: 00 76 00 69 00 63 00 65 00 43 00 6f 00 6e 00 66
000001d0: 00 6f 00 72 00 6d 00 61 00 6e 00 63 00 65 00 54
000001e0: 00 65 00 6d 00 70 00 6c 00 61 00 74 00 65 00 00
000001f0: 00 4d 00 50 00 40 00 4c 00 4c 00 00 00 74 d4 06
00000200: 18 df ca 09 45 a4 ba 9a ab cb 96 aa e8 0a 05 00
00000210: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
.
.
.
000006e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000006f0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000700: 00 00 00 00 00 00 00
Extended Content Description Object  (3 entries):
WMFSDKVersion (unicode) 9.00.00.2980
WMFSDKNeeded (unicode)  0.0.0.0000
IsVBR (bool)    0
Codec description object  (2 streams)
               Type = 0x2
        Description = Windows Media Audio 9
                       32 kbps, 32 kHz, stereo (A/V) 1-pass CBR
                 ID = 61 01
               Type = 0x1
        Description = Windows Media Video 9

                 ID = 57 4d 56 33
```

```
Multicast object  (len 0x18):
00000000: 50 0f 1d 54 4b 5b cf 11 a8 fd 00 80 5f 5c 44 2b
00000010: 04 00 00 00 0a 00 00 00
ContentEngine#
```