



## Advanced Routing Configurations

---

This appendix explains how to pre-position and serve content that is acquired from multiple origin servers by using either a WCCP-enabled router or a Content Router. The information is presented in the following five case examples:

- [Case 1: Understanding How Windows Media Content is Served When a Content Router is Used for Request Redirection, page C-1](#)
- [Case 2: Acquiring Dynamically Created Content and Serving It Using a Content Router, page C-2](#)
- [Case 3: Acquiring Statically Created Content and Serving It Using a Content Router, page C-7](#)
- [Case 4: Acquiring Dynamically Created Content and Serving It Using WCCP Routing, page C-9](#)
- [Case 5: Acquiring Statically Created Content and Serving It Using WCCP Routing, page C-12](#)

### Case 1: Understanding How Windows Media Content is Served When a Content Router is Used for Request Redirection

This section describes the basic exchange that takes place among clients, Content Routers, and Content Engines when a request is made for a Windows Media file. To describe this exchange, consider the following typical scenario:

You want to deliver a video training module to company employees. The video is an Active Streaming Format (ASF) file and will be delivered on demand to employee desktop Windows Media Players, which use the Microsoft Media Server (MMS) protocol. Your network uses a Content Router to intercept and redirect requests to the best-suited Content Engine where you have pre-positioned the video file for accelerated delivery. Because the MMS protocol does not understand redirection, you correctly publish the following URL on your company's internal website:

```
http://cr-fqdn/video.asf.asx
```

When you have an ACNS Content Router in your network and you are using MMS to stream a file, you must add the extension, “.asx” to the end of the publishing URL. Without the .asx extension, content delivery to the client will fail. The .asx extension is the file extension for a Microsoft Advanced Streaming Redirector (ASX) file, which is a metadata file that contains redirection URLs embedded in it. The ASX file is an important part of the exchange that takes place among clients, Content Routers, and Content Engines when you want to stream a file using the MMS protocol.



**Note**

---

You *do not* need to add the .asx extension for files that are to be played back using HTTP or RTSP.

---

When company employees click the link to this publishing URL in their web browsers, the following actions occur:

1. The web browser client sends the request for the file (video.asf) to the Content Router (cr-fqdn):  
`http://cr-fqdn/video.asf.asx`
2. The Content Router responds to the client by sending to the client an ASX file that contains the redirection URL for the best-suited Content Engine with the requested video file stored in its cache:  
`http://ce-name.ce.crfqdn/video.asf.asx`
3. The client resends the request to the Content Engine using this new URL.
4. The Content Engine returns an ASX file that contains the actual URLs for the content.
  - In ACNS 5.1 and 5.2 software, the ASX file contains two URLs, one for MMS (mms://) and one for HTTP (http://). The client first tries the MMS URL, and if that fails, the client tries the HTTP URL, and so forth.
  - In ACNS 5.3 software and later releases, the ASX file contains a third URL for RTSP (rtsp://). To send and receive streaming content by using the RTSP protocol, both the Windows Media client and the Windows Media server must be using Windows Media Version 9.
5. The client chooses which protocol to use based on which version of Windows Media Player is present on the client PC, and it sends the chosen URL back to the Content Engine to fetch the content.

`mms://ce-name.ce.crfqdn:1755/video.asf`




---

**Note** For live streams, automatic failover to the HTTP URL does not occur. With ACNS 5.1 or 5.2 software, if the MMS URL for a live event fails, the stream fails. Beginning with ACNS 5.3 software, if the MMS URL for a live event fails, the RTSP URL is tried. If both streaming protocols fail, the stream fails.

---

6. The Content Engine serves the video file to the client Windows Media Player by using the MMS protocol.

## Case 2: Acquiring Dynamically Created Content and Serving It Using a Content Router

In this case, the web pages on the origin server contain links to content that has been dynamically generated through scripts that are written using JavaScript or a similar script. The links referred to in the JavaScript scripts are impossible to follow and download because the acquirer in ACNS software does not support the acquisition of dynamic content. This section explains how to specify the acquisition of such dynamic content in an environment where requests are routed by a Content Router. In this scenario, the web pages on the origin server also have links to web pages hosted on other “external” origin servers. This example assumes that you are familiar with the setup required to route requests using a Content Router. (For more information and background on content routing, see [Chapter 4, “Setting Up Content Request Routing in the ACNS Network.”](#))

For the purpose of this example, assume that you have three origin servers that publish content: www-server, mms-server, and file-server. The origin server (www-server) is the main server that contains links to web pages on the other two “external” servers. The main origin server (www-server) contains the following two files that you want to pre-position in the ACNS network: 01.asx and

index.html. The index.html file has links to the 01.asx file on the same server (www-server), a URL on mms-server and file-server, and a JavaScript with links to URLs on mms-server. The 01.asx file has a link to a URL on mms-server.

**File: index.html**

```
<HTML>
<HEAD>
<TITLE>Two-server combo</TITLE>
</HEAD>
<BODY>
<H1>Index </H1>
<ul>
<li>
<a href="01.asx">
Start Windows Media Presentation1
</a>
<li>
<a href="http://mms-server/DCARoot/wmload.asf">
One Windows Media file
</a>
<li>
<a href="http://file-server.cisco.com/biff/http10k.txt">
Regular http file
</a>
<script language="JavaScript">
function doRedir(url)
{
document.location.href = url;
}
</script>
<li>
Click <a href="javascript:
doRedir('mms://mms-server/DCARoot/mbr/MBR_BillGInterview_Low.wmv');">here</a> for BillG
Interview.
<br>
<li>
Click <a href="javascript:
doRedir('mms://mms-server/DCARoot/mbr/MBR_BillGSpeech_Low.wmv');">here</a> for BillG
Speech.
</BODY></HTML>
```

**File: 01.asx**

```
<ASX Version = "3.0">
<AUTHOR> Biff</Author>
<Entry><Ref href = "http://mms-server/DCARoot/Consumer_100K.wmv" />
<Entry><Ref href = "mms://mms-server/DCARoot/ColorBars1_500K.wmv" />
</Entry>
</ASX>
```

To acquire this dynamic content using a Content Router, follow these steps:

- 
- Step 1** Make a copy of the published content on the main origin server, www-server, and place it in a new folder. (In this example, the duplicate folder is Folder2.)

The request-routed FQDN used for the Content Router that is serving the content is different from the origin server FQDN of the pages linked to the www-server, so you need to make a copy of the published content for pre-positioning purposes.

Also, to serve the content properly by using the Content Router, you need to make certain changes to the content itself before it is pre-positioned.

**Step 2** In all the files in Folder2, except for the Window Media meta files (that is, except for files ending with .asx, .wax, and .wvx extensions), make the following changes:

- a. Replace all references to external servers with corresponding request-routed FQDNs. (In this example, the request-routed FQDN is *ourtest.unicorn.com*.)
- b. In all URLs that reference Windows Media files (that is, filenames with .wma, .wmv, and .asf extensions), append the .asx extension to the filename.
- c. In all URLs that reference Windows Media files, replace mms:// with http://.

If there is a reference to a Windows Media meta file (files with .asx, .wax, or .wvx extensions), then the content that the meta file refers to must be pre-positioned using the same channel as the content located on the main origin server. ACNS software does not support ASX files that have pre-positioned content in other channels.

If the links to URLs are specified using JavaScript, then directory indexing can be enabled on the server on which those files reside so that all the required files can be easily acquired from that origin server.

On the secondary origin servers, you can make a duplicate virtual host that listens on a different port, with directory indexing enabled and with specific access rights. In this configuration, the main web page can still remain functional without exposing those files to regular customers.

After making the changes described in [Step 2](#), the content in Folder2 is changed as follows (changes are shown in bold):

```
[root@PC-45 Folder1]# cd ../Folder2
[root@PC-45 Folder2]# ls
01.asx  index.html

1. index.html:
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
  <HEAD>
    <TITLE>Two-server combo</TITLE>
  </HEAD>
  <BODY>
    <H1>Index </H1>
    <ul>
      <li>
        <a href="01.asx">
          Start Windows Media Presentation1
        </a>

      <li>
        <a href="http://ourtest.unicorn.com/DCARoot/wmload.asf.asx">
          One Windows Media file
        </a>

      <li>
        <a href="http://ourtest.unicorn.com/biff/http10k.txt">
          Regular http file
        </a>

      <script language="JavaScript">
        function doRedir(url)
        {
          document.location.href = url;
        }
      </script>
    </li>
```

```

Click <a href="javascript:
doRedir('http://ourtest.unicorn.com/DCARoot/mbr/MBR_BillGInterview_Low.wmv.asx');">here</a
> for BillG Interview.
<br>
<li>
Click <a href="javascript:
doRedir('http://ourtest.unicorn.com/DCARoot/mbr/MBR_BillGSpeech_Low.wmv.asx');">here</a>
for BillG Speech.

</BODY></HTML>

```

```

2. 01.asx:
(remains unchanged)
<ASX Version = "3.0">
<AUTHOR> Biff</Author>
<Entry><Ref href = "http://mms-server/DCARoot/Consumer_100K.wmv" />
<Entry><Ref href = "mms://mms-server/DCARoot/ColorBars1_500K.wmv" />
</Entry>
</ASX>

```

Now the content is ready to be acquired in the ACNS network. The next thing you must do is create the manifest file.

**Step 3** In the manifest file for the channel, you need to configure multiple jobs, as follows:

- To acquire files from the main origin server's Folder2 folder without crawling external servers, configure a crawl task or single item, whichever is convenient.
- To acquire files from a second origin server, configure directory crawling or single items.
- To acquire files from a third origin server, configure directory crawling or single items, whichever is convenient.

#### Sample Manifest File

```

<?xml version="1.0" ?>
<CdnManifest>
  <server name="cr-test1">
    <host name="http://www-server/biff/Folder2" proto="http"/>
  </server>

  <!-- get content from first origin server. Step 2a above-->
  <crawler server="cr-test1" start-url="index.html" depth="2" ttl="5" >
  </crawler>

  <!-- get content from the second origin server. Step 2b above-->
  <item src="http://mms-server/DCARoot/wmload.asf" />
  <!-- referenced in asx file -->
  <item src="http://mms-server/DCARoot/Consumer_100K.wmv" />
  <item src="mms://mms-server/DCARoot/ColorBars1_500K.wmv" />
  <!-- These two ensure that the links from the JavaScript are acquired-->
  <item src="mms://mms-server/DCARoot/mbr/MBR_BillGInterview_Low.wmv" />
  <item src="mms://mms-server/DCARoot/mbr/MBR_BillGSpeech_Low.wmv" />

  <!-- get content from the third origin server. Step 2c above-->
  <item src="http://file-server.cisco.com/biff/http10k.txt" />

</CdnManifest>
~

```

Alternatively, since not all links are from JavaScript, you can define an additional crawling task. The manifest file will look like this:

```
<?xml version="1.0" ?>
<CdnManifest>
  <server name="cr-test1-1">
    <host name="http://www-server/biff/Folder2" proto="http"/>
  </server>

  <server name="cr-test1-2">
    <host name="http://www-server/biff/Folder1" proto="http"/>
  </server>

  <!-- get content from first origin server-->
  <crawler server="cr-test1-1" start-url="index.html" depth="2" ttl="5" >
  </crawler>

  <!-- These two ensure that the links from the JavaScript which cannot be crawled are
  acquired-->
  <item src="mms://mms-server/DCARoot/mbr/MBR_BillGInterview_Low.wmv" />
  <item src="mms://mms-server/DCARoot/mbr/MBR_BillGSpeech_Low.wmv" />

  <!-- get content from the second and third origin servers using a crawl-->
  <crawler server="cr-test1-2" start-url="index2.html" depth="2" ttl="5"

externalPrefixes="http://file-server.cisco.com/|mms://mms-server/|http://mms-server/">
  <matchRule>
    <match match-url="http://file-server.cisco.com/.*/>
    <match match-url="mms://mms-server/.*/>
    <match match-url="http://mms-server/.*/>
  </matchRule>
</crawler>
</CdnManifest>
```

- Step 4** In the Content Distribution Manager GUI, create a website with the appropriate request-routed FQDN for the Content Router to use. (In this example, the request-routed FQDN is *ourtest.unicorn.com*.)
- Step 5** On the DNS server, create a record that sends everything in the domain *ourtest.unicorn.com* to the Content Router that is designated to route the requests. Make sure that the Content Router is registered with the same Content Distribution Manager as the Content Engines in the network.
- Step 6** In the Content Distribution Manager GUI, create a channel using the website you created and one of the manifests created above.
- Step 7** Make sure that there is enough bandwidth available for the Content Engines to be assigned to the channel, and then assign the required Content Engines to the channel.
- Step 8** Check the replication status, and make sure that replication is complete.
- Step 9** To play back the pre-positioned content, the setup is similar to setting up Content Router request routing. Set the DNS server on the client to point to the DNS server that you configured with your Content Router information. (You might need to remove the entry for the secondary DNS server to ensure that the request goes only to the primary DNS server configured, especially on a Windows system.)
- Step 10** Download or play the content using the request-routed FQDN URL. In this example, the request-routed FQDN is as follows:  
<http://ourtest.unicorn.com/index.html>

**Note**

If there is a chance that content acquired from different origin servers will overlap when placed under the same request-routed FQDN, then use different request-routed FQDNs and different corresponding channels to acquire content from each origin server. The web page links need to be modified to refer to the appropriate request-routed FQDNs. (See Step 2a above.)

## Case 3: Acquiring Statically Created Content and Serving It Using a Content Router

In Case 2, the origin server has static links (which can be crawled) to content on the same origin server and on different origin servers. To acquire this content using a Content Router, follow these steps:

- Step 1** Make a copy of the published content on the main origin and place it in a new folder. (In this example the duplicate folder is Folder3.)
- Step 2** In Folder3, do the following:
- Replace all references to remote servers with corresponding request-routed FQDNs (in this example, *ourtest.unicorn.com*).
  - In all references to WMT files, append the *.asx* extension to the filename.
  - In all references to WMT files, convert *mms://* to *http://*.

**Note**

The main difference between this step and [Step 2](#) in Case 1 is that in this step, you have to specify the items linked from the JavaScript script specifically in the manifest file. You can use either directory index crawling or a single item, whichever is convenient. MMS crawling for other links from the main web page is applicable for both cases.

If there is an ASX file, then the content it refers to must belong to the same channel as the content from the main origin server. In such a case, do not make any changes to the ASX file. ACNS software does not support ASX files that have content that is pre-positioned in other channels.

After making the changes described in [Step 2](#), the content in Folder3 is changed as follows (changes are shown in bold):

```
[root@PC-45 Folder1]# cd ../Folder3
[root@PC-45 Folder3]# ls
01.asx  index.html

1. index.html:
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
  <HEAD>
    <TITLE>Two-server combo</TITLE>
  </HEAD>
  <BODY>
<H1>Index </H1>
<ul>
<li>
<a href="01.asx">
Start Windows Media Presentationl
```

```

</a>

<li>
<a href="http://ourtest.unicorn.com/biff/http10k.txt">
Regular http file
</a>
</BODY></HTML>

```

```

2. 01.asx (unchanged):
<ASX Version = "3.0">
<AUTHOR> Biff</Author>
<Entry><Ref href = "http://mms-server/DCARoot/Consumer_100K.wmv" />
<Entry><Ref href = "mms://mms-server/DCARoot/ColorBars1_500K.wmv" />
</Entry>
</ASX>

```

**Step 3** In the manifest file for the channel, configure multiple jobs, as follows:

- a. Configure multiple jobs to crawl files from the main origin server's Content Router folder, and acquire content only from the external servers.
- b. Configure multiple jobs to crawl files from the main origin server's Folder3 folder, and acquire content only from the main origin server. (See the following examples.)

#### Sample Manifest File for Case 2

```

-----
<?xml version="1.0" ?>
<CdnManifest>

  <server name="cr-test2-1">
    <host name="http://www-server/biff/Folder3" proto="http"/>
  </server>

  <server name="cr-test2-2">
    <host name="http://www-server/biff/Folder" proto="http"/>
  </server>

  <crawler server="cr-test2-1" start-url="index.html" depth="2" ttl="5" >
</crawler>

  <crawler server="cr-test2-2" start-url="index1.html" depth="2" ttl="5"

externalPrefixes="http://file-server.cisco.com|mms://mms-server|http://mms-server/">
  <matchRule>
    <match match-url="http://file-server.cisco.com/.*/>
    <match match-url="mms://mms-server/.*/>
    <match match-url="http://mms-server/.*/>
  </matchRule>
</crawler>
</CdnManifest>
~

```

**Step 4** Create a website in the Content Distribution Manager GUI with *ourtest.unicorn.com* (for example) as the request-routed FQDN.

**Step 5** Using the website you created in [Step 4](#), create a channel in the Content Distribution Manager GUI.

- Step 6** On the DNS server, create a record that sends everything in domain *ourtest.unicorn.com* to the Content Router designated to route the requests. Make sure that the Content Router is registered with the same Content Distribution Manager as the Content Engines subscribed to the channel.
- 

## Case 4: Acquiring Dynamically Created Content and Serving It Using WCCP Routing

In Case 3, the origin server generates links to content that is dynamically generated through scripts that are written by using JavaScript or a similar script. The links referred to in the JavaScript scripts are impossible to follow and download because the acquirer in ACNS software does not support the acquisition of dynamic content.

This example assumes that you are familiar with configuring a router for WCCP and with any setup that is required to serve requests by using WCCP. For more information and background on WCCP routing, see [Chapter 4, “Setting Up Content Request Routing in the ACNS Network.”](#)

### Example of Files on the Main Origin Server

```
[root@PC-45 WCCP]# pwd
/var/ftp/biff/WCCP
[root@PC-45 WCCP]# ls
01.asx index.html
[root@PC-45 WCCP]#
1. index.html:
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<HEAD>
<TITLE>Two-server combo</TITLE>
</HEAD>
<BODY>
<H1>Index </H1>
<ul>
<li>
<a href="01.asx">
Start Windows Media Presentation1
</a>
<li>
<a href="http://mms-server/DCARoot/wmload.asf">
One Windows Media file
</a>
<li>
<a href="http://file-server.cisco.com/biff/http10k.txt">
Regular http file
</a>
<script language="JavaScript">
function doRedir(url)
{
document.location.href = url;
}
</script>
<li>
Click <a href="javascript:
doRedir('mms://mms-server/DCARoot/mbr/MBR_BillGInterview_Low.wmv');">here</a> for BillG
Interview.
<br>
<li>
```

```

Click <a href="javascript:
doRedir('mms://mms-server/DCARoot/mbr/MBR_BillGSpeech_Low.wmv');">here</a> for BillG
Speech.
</BODY></HTML>
2. 01.asx:
<ASX Version = "3.0">
<AUTHOR> Biff</Author>
<Entry><Ref href = "http://mms-server/DCARoot/Consumer_100K.wmv" />
<Entry><Ref href = "mms://mms-server/DCARoot/ColorBars1_500K.wmv" />
</Entry>
</ASX>

```

To serve such pre-positioned content by using WCCP routing, follow these steps:

- 
- Step 1** On the origin server, make a duplicate of the original folder. Copy it, for example, to Folder2.
- Step 2** In Folder2, replace all references to remote servers with corresponding FQDNs (in the example, www1.unicorn.com). No other change is required.

If the links to other files on the same origin server use JavaScript, then you need to enable directory indexing on the server so that all the required files are acquired from this origin server.

On the secondary origin servers, you can create a duplicate virtual host that listens on a different port with directory indexing enabled and with specific access rights. In this configuration, the main web page can still remain functional.

After making the changes described in [Step 2](#), the content in Folder2 is changed as follows:

```

[root@PC-45 WCCP]# cd ../Folder2
[root@PC-45 Folder2]# ls
01.asx  index.html

1. index.html:
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<HEAD>
  <TITLE>Two-server combo</TITLE>
</HEAD>
<BODY>
<H1>Index </H1>
<ul>
<li>
<a href="01.asx">
Start Windows Media Presentation1
</a>

<li>
<a href="http://www1.unicorn.com/DCARoot/wmload.asf">
One Windows Media file
</a>

<li>
<a href="http://www1.unicorn.com/biff/http10k.txt">
Regular http file
</a>

  <script language="JavaScript">
    function doRedir(url)
    {
      document.location.href = url;
    }
  </script>
<li>

```

```

Click <a href="javascript:
doRedir('mms://www1.unicorn.com/DCARoot/mbr/MBR_BillGInterview_Low.wmv');">here</a> for
BillG Interview.
<br>
<li>
Click <a href="javascript:
doRedir('mms://www1.unicorn.com/DCARoot/mbr/MBR_BillGSpeech_Low.wmv');">here</a> for BillG
Speech.

</BODY></HTML>
2. 01.asx:
<ASX Version = "3.0">
<AUTHOR> Biff</Author>
<Entry><Ref href = "http://www1.unicorn.com/DCARoot/Consumer_100K.wmv" />
<Entry><Ref href = "mms://www1.unicorn.com/DCARoot/ColorBars1_500K.wmv" />
</Entry>
</ASX>

```

**Step 3** In the manifest file for the channel, configure multiple jobs, as follows:

- a. To acquire files from the main origin server's Folder2 without crawling external servers, configure a crawl task or single item, whichever is convenient.
- b. To acquire files from the second and third origin servers, configure directory crawling or single items, or configure crawling the main origin server website with external server crawling allowed.

#### Sample Manifest File

```

-----
<?xml version="1.0" ?>
<CdnManifest>
  <server name="wccp-test1-1">
    <host name="http://www-server/biff/Folder2" proto="http"/>
  </server>

  <server name="wccp-test1-2">
    <host name="http://www-server/biff/WCCP" proto="http"/>
  </server>

  <crawler server="wccp-test1-1" start-url="index.html" depth="2" ttl="5" >
  </crawler>

  <!-- These two come from the links from the JavaScript which cannot be crawled -->
  <!-- We can also use directory index crawling here -->
  <item src="mms://mms-server/DCARoot/mbr/MBR_BillGInterview_Low.wmv" />
  <item src="mms://mms-server/DCARoot/mbr/MBR_BillGSpeech_Low.wmv" />

  <crawler server="wccp-test1-2" start-url="index2.html" depth="2" ttl="5"

externalPrefixes="http://file-server.cisco.com/|mms://mms-server/|http://mms-server/">
  <matchRule>
    <match match-url="http://file-server.cisco.com/.*/>
    <match match-url="mms://mms-server/.*/>
    <match match-url="http://mms-server/.*/>
  </matchRule>
</crawler>
</CdnManifest>
~

```

**Step 4** Create a website in the Content Distribution Manager GUI with *www1.unicorn.com* (or a suitable name) as the origin server FQDN.




---

**Note** If the main website is hosted at abc.xyz.com (for example), then you can simply choose to configure the website FQDN as abc.xyz.com.

---

- Step 5** Using the website you created in [Step 4](#), create a channel in the Content Distribution Manager GUI.
- Step 6** Configure the router and the Content Engine for WCCP.
- Step 7** If the domain name is new, configure the client's DNS server to include a DNS resolution for the FQDN that you configured in the Content Distribution Manager GUI in [Step 4](#). If the domain name is the same as the main website FQDN, then you do not need to configure the DNS server.
- (The website that the FQDN resolves to should not matter because the content will all be served by the Content Engine if content replication is completed successfully.)
- Step 8** Make sure that there is enough bandwidth available for the Content Engines to be assigned to the channel. Then assign the required Content Engines to the channel.
- Step 9** Check the content replication status, and make sure that replication is complete.
- Step 10** Set the DNS server on the client to point to the DNS server that you configured with the FQDN information. (You may need to remove the entry for the secondary DNS server to ensure that the request goes only to the primary DNS server configured, especially on a Windows system.)
- Step 11** Configure the default gateway to be the appropriate interface of the router on which WCCP is enabled.
- Step 12** Download and play the content by using this FQDN URL:  
http://www1.unicorn.com/index.html




---

**Note** If there is a chance that content acquired from different origin servers will overlap when placed under the same FQDN, use different FQDNs and different corresponding channels to acquire content from each origin server.

---

## Case 5: Acquiring Statically Created Content and Serving It Using WCCP Routing

In Case 4, the origin server has static links (which can be crawled) to content that is on the same origin server and on different origin servers.

For the sake of completeness, consider a case in which multiple channels are configured: one for each origin server from which content is to be acquired. This step is not necessary unless there is a risk of overlapping content that was acquired from different origin servers because the same relative URLs have been acquired from different origin servers.

### Example of Files on Main Origin Server

Original files:  
-----

```
[root@PC-45 WCCP]# pwd
/var/ftp/biff/WCCP
[root@PC-45 WCCP]# ls
01.asx  index.html
```

```
[root@PC-45 WCCP]#
1. index.html:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<HTML>
<HEAD>
  <TITLE>Two-server combo</TITLE>
</HEAD>
<BODY>
<H1>Index </H1>
<ul>
<li>
<a href="01.asx">
Start Windows Media Presentation1
</a>

<li>
<a href="http://mms-server/DCARoot/wmload.asf">
One Windows Media file
</a>

<li>
<a href="http://file-server.cisco.com/biff/http10k.txt">
Regular http file

</a>

</BODY></HTML>

2. 01.asx:

<ASX Version = "3.0">
<AUTHOR> Biff</Author>
<Entry><Ref href = "http://mms-server/DCARoot/Consumer_100K.wmv" />
<Entry><Ref href = "mms://mms-server/DCARoot/ColorBars1_500K.wmv" />
</Entry>
</ASX>
```

To serve such pre-positioned content by using WCCP routing, follow these steps:

- 
- Step 1** Create manifest files to acquire content from each origin server. No changes are required in the pre-positioned content files.

### Sample Manifest File

Manifest file for first channel (corresponding to the main origin server that is the starting point):

```
-----
<?xml version="1.0" ?>
<CdnManifest>
  <server name="wccp-test2-1">
    <host name="http://www-server/biff/WCCP" proto="http"/>
  </server>

  <crawler server="wccp-test2-1" start-url="index.html" depth="2" ttl="5" >
  </crawler>

</CdnManifest>
~
```

Manifest file for www2.unicorn.com:

```

-----
<?xml version="1.0" ?>
<CdnManifest>
  <server name="wccp-test2-2">
    <host name="http://www-server/biff/WCCP" proto="http"/>
  </server>

  <crawler server="wccp-test2-2" start-url="index.html" depth="2" ttl="5"
    externalPrefixes="http://file-server.cisco.com/">
    <matchRule>
      <match match-url="http://file-server.cisco.com/.*/>
    </matchRule>
  </crawler>
</CdnManifest>
~

Manifest file for www3.unicorn.com:
-----
<?xml version="1.0" ?>
<CdnManifest>
  <server name="wccp-test2-3">
    <host name="http://www-server/biff/WCCP" proto="http"/>
  </server>

  <crawler server="wccp-test2-3" start-url="index.html" depth="2" ttl="5"
    externalPrefixes="mms://mms-server/|http://mms-server/">
    <matchRule>
      <match match-url="mms://mms-server/.*/>
      <match match-url="http://mms-server/.*/>
    </matchRule>
  </crawler>
</CdnManifest>
~

```

- Step 2** Create one or more websites in the Content Distribution Manager GUI. For the origin server FQDN, use the hostname or IP address of the origin servers from which content is to be acquired.
- Step 3** Configure the router and the Content Engine for WCCP.
- Step 4** Configure the client's DNS server, and make sure that it includes DNS resolution for the FQDNs that you configured in the Content Distribution Manager GUI.
- Step 5** Create channels in the Content Distribution Manager GUI by using the websites that you created in [Step 2](#).
- Step 6** Make sure that there is enough bandwidth available for the Content Engines to be assigned to the channel. Then assign the required Content Engines to the channel.
- Step 7** Check the content replication status, and make sure that the replication for all channels is complete.
- Step 8** Set the DNS server on the client to point to the DNS server that you configured with the FQDN information. (You may need to remove the entry for the secondary DNS server to ensure that the request goes to only the primary DNS server configured, especially on a Windows system.)
- Step 9** Configure the default gateway to be the appropriate interface of the router on which WCCP is enabled.
- Step 10** Download and play the content by using the FQDN URL.

This information can be obtained on the root Content Engine by using the **show statistics acquirer content channel-id** *channel-id* command in the CLI. In this case, you start with `http://www-server/index.html`.



