APPENDIX B

Creating Manifest Files

This appendix describes the process for creating Manifest files used to acquire and distribute content within the CDS network. This appendix includes the following topics:

For information about using a Manifest file in a delivery service, see the ""Identifying Content Using a Manifest File" section on page 5-36.

Introduction

The CDS is used to ingest, distribute, and deliver multi-format content to different client devices. To specify the content to be prefetched and to control the delivery of the prefetched content, an XML file called a *Manifest file* is used. Third-party asset management systems can interoperate with the CDS by using this Manifest file interface. Each delivery service in the CDS can be configured with or without a Manifest file. The Manifest file can also be automatically generated by using the CDSM. The Manifest file is primarily used in prefetch ingest and hybrid ingest.

The Manifest file is specified in the CDSM in the following ways:

- External Manifest File Specification—The Manifest file is hosted on an external server and a URL pointing to that server is configured in the delivery service. The Manifest file can be fetched using FTP, HTTP, HTTPS and CIFS protocols.
- GUI Configured—The CDSM GUI can generate a Manifest file. The CDSM provides the required elements for the user to create a Manifest file and to specify the attributes in the Manifest file. Only commonly used attributes are supported by the CDSM.

The Manifest file is processed by the Content Acquirer. The Content Acquirer parses the Manifest file, creates the metadata based on the attributes in the file, and prefetches the content specified. For live content and content that is ingested on demand, the Content Acquirer creates the metadata and does not fetch the actual content. The metadata created by the Content Acquirer is propagated to all the Service Engines participating in the delivery service.

Manifest File Requirements

The Manifest file needs to support different attributes and tags to support content prefetching and hybrid ingest. The basic requirements for a Manifest file are the following:

- Specify Content to Be Prefetched—There are two ways to specify prefetched content. One is to use a single item, where users specify a single URL and the Content Acquirer ingests only the content pointed to by this URL. Another way is by using a crawler item, where users specify a crawl job with parameters like start-url, depth, prefix, and reject or accept. In this case, the Content Acquirer crawls the origin server to fetch content based on the parameters.
- Specify Schedule Information—To instruct the Content Acquirer when to ingest the content and how often to check the server for updates.
- Specify Publish Information—Information about how content is accessed by the end users; for example, the playserver attribute specifies which server to use for playing the content, the cdn-url attribute specifies which URL is used by end-users to access the content, the *serveStartTime* and *serveStopTime* attributes instruct the CDS when it can serve the content and provides additional metadata for playing.
- Specify Live Streaming Content—The Manifest file can also be used to specify live stream splitting.
- Specify Metadata for Hybrid Ingest Content—For hybrid ingest, the Manifest file can be used to specify the content serve start and stop time for content ingested on demand.

Working with Manifest Files

This section provides Manifest file samples for carrying out specific tasks. Each sample has an associated explanation of its purpose and function. The Manifest file can specify a single content object, a website crawler job, or an FTP server crawler job to acquire prefetched content or to acquire information about live content that is distributed to edge Service Engines later.

Specifying a Single Content Item

Use the <item> tag to specify a single content item, object, or URL. The required *src* attribute is used to specify the relative path portion of the URL. If the server *name* attribute is omitted, the server *name* attribute in the last specified <server> tag above it is used. If there are no <server> tags close by in the Manifest file, the server that hosts the Manifest file is used, which means that the relative URL is relative to the Manifest file URL.

The following example provides an example of a Manifest file that specifies single content items:

```
<CdnManifest>
    <item src="http://www.my-server/test.html" />
    <item src="test.html" />
    <server name="my-origin-server-one">
        <host name="http://www.my-server-one.com/eng/" />
    </server>
    <server name="my-origin-server-two">
        <host name="http://www.my-server-two.com/eng/" />
    </server>
    <item src="project-two.html" />
    <item server="my-origin-server-one" src="project-one.html" />
</CdnManifest>
```

For a single item, you specify the item's URL in the *src* attribute. There are two ways to specify the item URL:

- Specify the *src* attribute with the absolute URL as shown in the following format:

  ```
  proto://username:password@/domain-name:port/file-path/file-name
  ```

  In the example, the first <item> tag uses the full path.

- Specify the origin server information using the <server><host> tags and use the *src* attribute to specify only the relative path.

  In the example, every <item> tag except the first one uses a relative path. The second <item> tag uses the Manifest file server, where test.html is relative to the Manifest file URL. The second <item> tag, "project-two.html," uses "my-origin-server-two." The third <item> tag, "project-one.html," uses "my-origin-server-one."

Specifying a Crawl Job

The crawler feature methodically and automatically searches acceptable websites and makes a copy of the visited pages for later processing. The crawler starts with a list of URLs to visit and identifies every web link in the page, adding these links to the list of URLs to visit. The process ends after one or more of the following conditions are met:

- Links have been followed to a specified depth.
- Maximum number of objects has been acquired.
- Maximum content size has been acquired.

By crawling a site at regular intervals using the time-to-live (or *ttl)* attribute, these links and their associated content can be updated regularly to keep the content fresh. Use the <crawler> tag to specify the website or FTP server crawler attributes. Table B-1 lists the attributes, states whether these attributes are required or optional, and describes their functions.

Table B-1 Website or FTP Server Crawl Job Attributes

Attribute	Description
start-url	(Required) Identifies the URL to start the crawl job from. It can be a full path or a relative path. If it is a relative path, the <server><host> tags are required to specify the origin server information.
depth	(Optional) Defines the level of depth to crawl the specified website. The depth is defined as the level of a website's URL links or FTP server's directory, where 0 is the URL or directory from which the crawl job starts. 0 = Acquire only the starting URL. 1, 2, 3, ... = Acquire the starting URL and its referred files to the depth specified. –1 = Infinite or no depth restriction. If the depth is not specified, the default is used. The default is 20. **Note** It is not advisable to specify a depth of –1 because it takes a long time to crawl a large website and is wasteful if all the content on that particular website is not required.

Table B-1 Website or FTP Server Crawl Job Attributes (continued)

Attribute	Description

prefix	(Optional) Combines the hostname from the <server> tag and this field to create a full prefix. Only content with URLs that match the full prefix are acquired, as shown in this example:
	`<server name="xx"> <host name="www.cisco.com" proto="https" port=433/> </server>`
	with the following <crawler> tag:
	`prefix="marketing/eng/"`
	The full prefix is "https://www.cisco.com:433/marketing/eng/." Only URLs that match this prefix are crawled. If a web page refers to ".../marketing/ops," the marketing/ops page and its children are not acquired.
	If the prefix is omitted, the crawler checks the default full prefix, which is the hostname portion of the URL from the server. In the example, the default full prefix is "https://www.cisco.com:433."

accept	(Optional) Uses a regular expression to define acceptable URLs to crawl, in addition to having acceptable URLs match a prefix. For example, accept="stock" means that only URLs that meet two conditions are crawled: the URL matches the prefix and also contains the regular expression string "stock."
reject	(Optional) Uses a regular expression to reject a URL if it matches the expression. The URL is first checked for a possible prefix match and then checked for a reject regular expression. If a URL does not match the prefix, it is immediately rejected. If a URL matches both the prefix and the reject regular expression, it is rejected by the expression.
max-number	(Optional) Specifies the maximum number of crawl job objects that can be acquired.
maxTotalSizeInMB *maxTotalSizeInKB* *maxTotalSizeInB*	(Optional) Specifies the maximum size of content that this crawl job can acquire. The size can be expressed in bytes (B), kilobytes (KB), or megabytes (MB). **Note** The maximum size of the file that is acquired is going to be less than the amount of disk space required to store the file. Files, when stored, contain overhead that contributes to the amount of disk space used for the delivery service. This overhead is approximately 20 KB per file. File size and storage overhead need to be taken into account when you are configuring the delivery service disk quota. This attribute replaces the *max-size-in-B/KB/MB* attribute. The *max-size-in-B/KB/MB* attribute continues to be supported for backward compatibility only.
externalPrefixes	(Optional) Specifies additional prefixes for crawl jobs to crawl multiple protocols or multiple websites. Prefixes are separated with a bar (|).
externalServers	(Optional) Specifies additional hosts for crawl jobs. Can be used for multiple host crawl jobs where each host has a different user account. This attribute can be used to see the <host> tag with the proper authentication information.

Note If you specify both the *max-number* and *maxTotalSizeIn* attributes as the criteria to use to stop a crawl job, the condition that is met first takes precedence. The crawl job stops either when the maximum number of objects is acquired or when the maximum content size is reached, whichever occurs first. For example, if the crawl job has acquired the maximum number of objects specified in the Manifest file but has not yet reached the maximum content size, the crawl job stops.

The following is an example of a website crawl job:

```
<server name="cisco">
    <host name="http://www.cisco.com/jobs/" />
</server>
<crawler
    server="cisco"
    start-url="eng/index.html"
    depth="10"
    prefix="eng/"
    reject="\.pl"
    maxTotalSizeIn-MB="200"
/>
```

This website crawl job example contains the following attributes:

- The *start-url* path is http://www.cisco.com/jobs/eng/index.html.
- Search to a website link *depth* of 10.
- Search URLs with the *prefix* http://www.cisco.com/jobs/eng/.
- Reject URLs containing .pl (Perl script pages).
- Only crawl until 200 megabytes in total content size are acquired.

If the server *name* attribute is omitted, the server *name* in the last specified <server> tag above it is used. If there are no <server> tags close by in the Manifest file, the server that hosts the Manifest file is used, which means that the relative URL is relative to the Manifest file URL.

Understanding the Prefix Attribute

When the *prefix* attribute is specified in the crawler tag, it refers to the prefix that must be added to the *start-url* when the Content Acquirer starts crawling a directory. This specifies the scope of the crawl, as shown in the following example:

```
<CdnManifest>
<crawler start-url="http://172.19.227.33/"
         prefix="test/9"
         depth="2"
         />
</CdnManifest>
```

In this example, the crawl starts at http://172.19.227.33/test/9.

When the *prefix* attribute is specified in the match tag, it specifies a filter that provides a short list of content that must be acquired, after a crawl job is started from a given *start-url*. When the Content Acquirer crawls, it could find several resources that need to be fetched. Each of the resources is identifiable using a URL. The *prefix* attribute in the match tag specifies the criteria to match before a URL is obtained. All URLs that match the given *prefix* are acquired.

In the following example, only URLs that match “http://linux-1.cisco.com/icons” are acquired.

```
<CdnManifest>
```

```
<options timeZone="PDT" />
<crawler host="http://linux-1.cisco.com"
         start-url="test/MPEG_files"
         depth="1" >
<matchRule>
    <match prefix="http://linux-1.cisco.com/icons/" />
</matchRule>
</crawler>
</CdnManifest>
```

The *prefix* attribute in the crawler tag and the prefix in the match tag can coexist.

Writing Common Regular Expressions

A regular expression is a formula for matching strings that follow a recognizable pattern. The following special characters have special meanings in regular expressions:

. * \ ? [ ] ^ $

If the regular expression string does not include any of these special characters, then only an exact match satisfies the search. For example, “stock” must match the exact substring “stock.”

Scheduling Content Acquisition

Two attributes, *ttl* and *prefetch*, are used to schedule content acquisition. Use *ttl* to specify the frequency of checking the content for freshness, in minutes. For example, to check for page freshness every day, enter ttl=“1440.”

In the following example, page freshness is scheduled to be checked once a day:

```
<item
    src="index.html"
    ttl="1440"
/>
```

In the following example, page freshness is scheduled to be crawled and checked every hour to a link *depth* value of 2:

```
<crawler
    start-url="index.html"
    depth="2"
    ttl="60"
/>
```

If the content is not yet available at a particular URL, the *prefetch* attribute can be used to specify the start time for acquisition at the specified URL. For example, prefetch=“2002-06-28 18:35:21” means the content acquisition job can only start on June 28, 2002 and at the specified time.

The following example schedules a crawl of this website every hour to a link *depth* value of 2 to start on November 9, 2001 at 8:45 a.m.

```
<crawler
    start-url="index.html"
    depth="2"
    prefetch="2001-11-09 08:45:12"
    ttl="60"
/>
```

Specifying Shared Attributes

Attributes in single <item> tags can be shared or have the same attribute values. Instead of writing these attributes individually for every <item> tag, you can extract them and place them in a higher-level tag called <item-group>, where these attributes can be shared from this higher-level tag. You can create an <item-group> tag at a level below the <CdnManifest> tag, and write <item> tags into it as subtags, moving shared attributes into the <item-group> tag, as shown in the following example:

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

<server name="cisco-cco">
  <host name="http://www.cisco.com"
        proto="http" />
</server>

<item-group
    server="cisco-cco"
    ttl="1440"
    type="prepos" >

   <item src="jobs/index.html"/>
   <item src="jobs/index1.html"/>
   <item src="jobs/index2.html"/>
   <item src="jobs/index3.html"/>
   <item src="jobs/index4.html"/>
   <item src="jobs/index5.html"/>

</item-group>

</CdnManifest>
```

You can also use the <options> tag to share attributes at the top-most level of the Manifest file. Shared attributes in the <options> tag can be shared by every <item> tag or by the <crawler> tag in the Manifest file. However, if a shared attribute is specified in both the <item-group> and the <item> tags or the <options> and <item> tags, attribute values in the <item> tags take precedence over the <item-group> and <options> tags.

The following example illustrates this precedence rule. The first <item> tag takes the *ttl* value 1440 from the <options> tag, but the second <item> uses its own *ttl* value of 60.

```
<options
   ttl="1440" >
<item src="index.html" />
<item src="index1.html" ttl="60" />
```

Specifying a Crawler Filter

With a rule-based crawler filter, you can crawl an entire website and only acquire contents with certain predefined characteristics. In contrast, crawler attributes in the <crawler> tag do not act as filters but only define the attributes for crawling. The <matchRule> tag is designed to act as a rule-based filter. You can define rule-based matches for file extensions, size, content type, and timestamp. In the following example, the crawl job is instructed to crawl the entire website starting at “index.html,” but to acquire only files with the .jpg extension and those larger than 50 kilobytes.

```
<crawler
   start-url="index.html" >
   <matchRule>
```

```
        <match minFileSizeIn-KB="50" extension="jpg" />
    </matchRule>
</crawler>
```

There can be multiple <match> subtags within a <matchRule> tag. Table B-2 lists and describes the <match> subtag attributes.

Table B-2 <match> Subtag Attributes

Attribute	Description
mime-type	Specifies match of these MIME-types.
extension	Specifies match of files with these extensions.
time-before	Specifies match of files modified before this time (using the Greenwich mean time [GMT] time zone) in yyyy-mm-dd hh:mm:ss format.
time-after	Specifies match of files modified after this time (using the Greenwich mean time [GMT] time zone) in yyyy-mm-dd hh:mm:ss format.
minFileSizeInMB *minFileSizeInKB* *minFileSizeInB*	(Optional) Specifies match of content size equal to or larger than this value. The size can be expressed in megabytes (MB), kilobytes (KB), or bytes (B).
maxFileSizeInMB *maxFileSizeInKB* *maxFileSizeInB*	(Optional) Specifies match of content size equal to or smaller than this value. The size can be expressed in megabytes (MB), kilobytes (KB), or bytes (B).
prefix	(Optional) Specifies a prefix as a match rule to filter out websites during a crawl job.
url-pattern	(Optional) Specifies a regular expression as a match rule to filter out certain URLs.

A <match> subtag can specify multiple attributes. Attributes within a <match> tag have a Boolean AND relationship. In the following example, to satisfy this match rule, a file must have an .mpg type file extension and its size must be larger than 50 kilobytes.

```
<match extension="mpg" minFileSizeIn-KB="50" />
```

There is a Boolean OR relationship between the <match> rules themselves. A <matchRule> tag can have multiple <match> subtags, but only one of these subtags must be matched. The <matchRule> tag can be specified as a subtag of the <crawler> tag, or a subtag of the <item-group> tag. If there is a subtag in an <item-group> tag, it is shared by every <crawler> tag within that <item-group> tag.

Note The *accept* or *reject* attributes can be mistakenly used in the <crawler> tag for a crawler filter.

For example, to crawl files with the extension .mpg, simply specifying accept="\.mpg" is not correct. In this case, although specifying accept="\.mpg" is not technically incorrect, no crawling occurs. Pages with URLs that do not match the *accept* constraint are not searched. For example, if the starting URL is index.html, this HTML file is parsed and any links not containing .mpg are rejected. If the .mpg files are located in the second or lower link levels, they are not fetched because the links connecting them have been rejected.

To properly crawl for the .mpg extension, use <matchRule>. Specify <matchRule> <match extension="mpg" />. The whole site is crawled and only those files with the .mpg extension are retained.

The *url-pattern* attribute in the match tag specifies a filtering criteria for the crawl. As the Content Acquirer identifies resources that must be acquired, it validates the URL of those resources and content against the specified URL pattern and acquires them only if the pattern matches.

In the following example, the *url-pattern* value is a regular expression. The meaning of the regular expression is to not match URLs that have an mpeg extension. Only items that do not match the mpeg extension are acquired.

```
<CdnManifest>
<options timeZone="PDT" />
<crawler host="http://172.19.227.33"
         start-url="AD" >
<matchRule>
    <!-- exclude mpeg extension -->
    <match url-pattern="[.](?!mpeg$).*$" />
</matchRule>
</crawler>
</CdnManifest>
```

Specifying Content Priority

A priority can be assigned to content objects to define their order of importance. The CDS 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 Service Engines.

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

Three factors combine to determine content priority:

- Delivery Service priority—Content Distribution Priority drop-down list in the Acquisition and Distribution Properties area of the Delivery Service Definition page in the CDSM
- Item index—Content order listed in the Manifest file
- Item priority—Priority of the attributes specified in the <item> or <crawler> tag

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

- If there is a priority value for this content specified in the Manifest file *priority* attribute, use the following formula:

  Content priority = Delivery service priority * 10000 + Item priority

  In this formula, Item priority can be any integer and is unrestricted.

Tip If you want a particular content object to have the highest priority, specify a very large integer value for 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 = Delivery service priority * 10000 + 10000 – Item index

  In this formula, Item index is the order in which content is listed in the Manifest file.

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

Generating a Playserver List

The CDS software supports playservers that play back the following prefetched content types on the CDS network: HTTP, HTTPS, RTSP, and RTMP (Movie Streamer, Windows Media, Flash Media Streaming).

The CDS software checks whether the requested protocol matches the list in the playserver table. If it matches, the request is delivered. If it does not match, the request is rejected.

You can generate a playserver list in the following ways:

- By configuring playserver attributes in an <item> tag
- By configuring playserver MIME-type extension names in a <playServerTable> tag

To create the playserver list directly through the Manifest file, configure playserver attributes of the playserver list in an <item> tag. If an <item> tag does not have a playserver attribute, its playserver list is generated through the <playServerTable> tag. If the <playServerTable> tag is omitted in the Manifest file, a built-in default <playServerTable> tag is used to generate the playserver list. Multiple servers are separated by commas, as shown in the following example:

```
<item src="video.mpg" playServer="wmt,http" />
```

You can also generate the playserver list that supports these streaming media types through the <playServerTable> tag. The <playServerTable> tag maps content into a playserver list based on the MIME-type extension name. If there is a <playServerTable> tag in the Manifest file, use that tag.

To generate the playserver list though the <playServerTable> tag, use MIME-type extension names to configure which playserver can play the particular prefetched content, as shown in the following example:

```
<playServerTable>
<playServer name="wmt">
         <extension name="wmv" />
         <extension name="wma" />
         <extension name="wmx" />
         <extension name="asf" />
</playServer>
<playServer name="http">
         <contentType name="application/pdf" />
         <contentType name="application/postscript" />
         <extension name="pdf" />
         <extension name="ps" />
</playServer>
</playServerTable>
```

The <playServerTable> tag is used to generate a playserver list for each content type. In the preceding example, any Portable Document Format (.pdf) or PostScript (.ps) file uses HTTP to play the content.

Customized Manifest Playserver Tables and the HTTP Playserver

In general, you do not need to specify your own playserver table or playserver in the Manifest file. A default playserver table maps appropriate file extensions or MIME-types to the proper playservers.

When you use the default playserver table, the HTTP playserver is always included in the playserver list, and this allows prefetched content to be played using HTTP. If the default playserver table does not meet your needs, you can customize your playserver lists by defining your own playserver table or by specifying a *playServer* attribute in the Manifest file.

The HTTP playserver is included in the default playserver table. However, if you specify your own playserver table or *playServer* attribute in the <item> or <crawler> tags, you must add the HTTP playserver to play HTTP content or other content using HTTP.

Specifying Attributes for Content Serving

Certain attributes in the Manifest file can be specified to control the manner in which content is served by the Service Engines. These attributes can be specified in the <item> and <crawler> tags. These same attributes can also be specified in the <item-group> or <options> tags, so they can be shared by their <item> and <crawler> subtags. Table B-3 lists and describes these content-serving attributes.

Table B-3 Attributes for Content Serving

Attribute	Description
serveStartTime	(Optional) Designates a time in yyyy-mm-dd hh:mm:ss format at which the CDS software is allowed to start serving the content. If the serving start time is omitted, content is ready to serve once it is distributed to the Service Engine.
serveStopTime	(Optional) Designates a time in yyyy-mm-dd hh:mm:ss format at which the CDS software temporarily stops serving the content. If the serving stop time is omitted, the CDS software serves the content to the Service Engine until the content is removed by modifying the Manifest file or renaming the delivery service.

Table B-3 Attributes for Content Serving (continued)

Attribute	Description
ignoreQueryString	Playback attribute that can be used with the <options>, <item-group>, <item>, and <crawler> tags. If *ignoreQueryString* is set to true, then the CDS software ignores any string after a question mark (?) in the request URL for playback. If this attribute is omitted, then the default value is false. For example, content with the request URL url=http://web-server/foo has been prefetched. If a user requests content with the URL url=http://web-server/foo?id=xxx and the *ignoreQueryString* attribute is set to false, then the CDS software does not use prefetched content from the request URL http://web-server/foo. However, if the *ignoreQueryString* attribute is set to true, then the CDS software treats the request URL http://www-server/foo?id=xxx the same as http://www-server/foo and returns prefetched content.
wmtRequireAuth	(Optional) Determines whether users need to be authenticated before the specified content is played. When *wmtRequireAuth* is set to true, the Service Engine requires authentication to play back the specified content to users and communicates with the origin server to check credentials. If the requests pass the credential check, the content is played back from the Service Engine. If this attribute is omitted, a heuristic approach is used to determine the setting: if the specified content is acquired by using a username and password, *wmtRequireAuth* is set to true; otherwise, it is set to false. For FTP, if the username is anonymous, *wmtRequireAuth* is set to false. **Note** If *wmtRequireAuth* is true, the Origin Server field in the CDSM Content Origin page for this delivery service needs to point to the server that can authenticate the users. When users want to play back the content, the server specified in the Origin Server field is checked for authentication.

Specifying Time Values in the Manifest File

The following attributes require that you enter a time value in the format yyyy-mm-dd hh:mm:ss (year-month-day hour:minute:second):

- *prefetch*
- *serveStartTime*
- *serveStopTime*
- *expires*
- *time-before*
- *time-after*

In the Manifest file, the time string conforms to the yyyy-mm-dd hh:mm:ss format. A time zone designation can be specified optionally at the end of a time string to indicate the particular time zone used. If a time zone designation is omitted, the GMT time zone is used. Note that automatic conversion between daylight saving time and standard time within a time zone is not supported, but a special designation for daylight saving time can be used, such as PDT for Pacific daylight saving time. In the following example, the prefetch time is September 5, 2002 at 09:09:09 Pacific daylight saving time:

```
<options timeZone="PDT" />
```

```
<item src="index.html" prefetch="2002-09-05 09:09:09 PDT" />
```

Refreshing and Removing Content

Use the *ttl* (time-to-live) and *expires* attributes of the Manifest file to monitor and control the freshness of content objects, and remove them.

The *ttl* attribute is expressed in minutes and specifies how frequently the software checks the freshness of the content at the origin server. If the *ttl* attribute is specified inside an <item> tag, it applies to that item; if it is specified inside a <crawler> tag, the attribute applies to the crawl job.

For example, if you give the *ttl* attribute a value of 10, the software checks the item or crawl job every 10 minutes. If the item has been updated, then the updated file is reacquired.

Caution Sometimes a crawl job can be very large, crawling over thousands of files. The recrawl speed is 5000 files per hour for small files. It is time-consuming to recheck so many files. We strongly recommend that you specify a large *ttl* value for such crawl jobs (for example, 1440 minutes [daily]). Otherwise, the software continues to crawl the site over and over again, blocking other acquisition tasks.

If you omit the *ttl* attribute in the Manifest file, the time-to-live is assumed to be zero and the software does not recheck that item after it is acquired. A value of 0 (zero) for *ttl* means that the content is fetched only once and is never checked again unless you click the **Fetch Manifest Now** button in the CDSM or use the **acquirer start-delivery-service** EXEC command in the Content Acquirer CLI.

The **Fetch Manifest Now** button is located in the Delivery Service Content page in the CDSM. When you click this button, the 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 delivery service is rechecked and the updated content is downloaded.

If you assign a negative value to the *ttl* attribute, such as –1, that item is never to be rechecked. A negative *ttl* attribute value prevents the software from checking item freshness, even if you click the **Fetch Manifest Now** button or use the **acquirer start-delivery-service** command.

Note Configuring the update interval in the CDSM GUI (**Services > Service Definition > Delivery Services > Delivery Service Content**) sets the interval for checking updates to the Manifest file itself. This setting only pertains to checking the Manifest file; it does not pertain to checking the content.

The *failRetryInterval* attribute is sometimes confused with the *ttl* attribute. The fail and retry feature acts upon failed content or failed updates. If the acquisition of a single item or of some crawled content fails, the software automatically tries to refetch these failed objects after a default interval of 5 minutes. The fail and retry interval can also be specified by using the *failRetryInterval* attribute in the Manifest file.

The difference between the *failRetryInterval* attribute and the *ttl* attribute is that the *ttl* attribute is for successfully acquired content and the *failRetryInterval* attribute is for content acquisition failures. The *ttl* attribute must be specified for the software to recheck the content freshness, whereas the *failRetryInterval* attribute does not need to be specified unless you want to change the retry interval.

The *expires* attribute specifies the time the content is to be removed from the CDS network. If you do not specify a time when you set the *expires* attribute, content is stored in the CDS network until it is explicitly removed when you modify the Manifest file. The *expires* attribute uses the format yyyy-mm-dd hh:mm:ss (year-month-day hour:minute:second). In the following example, the content expires on June 12, 2003 at 2:00 p.m.

```
expires="2003-06-12 14:00:00 PST"
```

If the *expires* attribute is specified inside an <item> tag, it applies to that item; if it is specified inside a <crawler> tag, the attribute applies to the crawl job.

You can monitor the status of content replication and freshness by enabling and then viewing the transaction log files that reside on the Service Engines. To verify whether or not a content object or file was successfully imported to or refreshed on a particular Service Engine, take these actions:

- Enable the transaction log function on the Service Engine you want to monitor.
- View the transaction log entries for the content object or filename that resides on that Service Engine.

Specifying Live Content

Only Windows Media live contents can be specified in the Manifest file. Use the <item> tag and specify the *type* attribute as wmt-live, as shown in the following example. The live stream for the wmt-live content type is url=rtsp://www.company-web-site.org/tmp/ceo-talk.

```
<CdnManifest>
<server name="wmt-server">
<host name="rtsp://www.company-web-site.org" />
</server>
<item src="/tmp/ceo-talk" type="wmt-live" >
</item>
<!--
This is a "wmt-live" streaming content type specified by the "type" attribute. The live
stream URL is
rtsp://www.company-web-site.org/tmp/ceo-talk.
-->
</CdnManifest>
```

Note If you are using the Manifest file for live streaming, the origin server configured for the delivery service should be the same as the encoder IP address.

Note Existing live content is deleted and replaced with the content specified in the Manifest file under the following conditions:

- You create two delivery services that use the same content origin server in the following way:
  1. Create a live streaming delivery service by using the CDSM GUI.
  2. Use a Manifest file to set up live streaming on a prefetch/caching delivery service by using the *type* attribute with wmt-live as the value and wmt-live as the *src*.
- You assign an SE to the prefetch/caching delivery service with the live Manifest file assigned, the existing live content is overwritten with the content specified in the Manifest file if the program name is the same (in this example, wmt-live).

This is because the content is the latest assigned, whichever was assigned last, whether it was by the prefetch/caching delivery service or the live streaming delivery service.

Specifying Hybrid Ingest Content

For hybrid ingested content, the content is not prefetched into the CDS network. Instead, the content is ingested dynamically based on the user request. This type of ingest is called *dynamic ingest* or *on-demand ingest*. To control the play back of the on-demand content, a new type of ingest has been introduced called *hybrid ingest*. In this method, the metadata for on-demand contents can be specified in the Manifest file. However, the actual content is not acquired by the Content Acquirer.

Hybrid ingest is supported by specifying “cache” as the value for the *type* attribute inside the <item> tag.

Note This mode of ingest is supported only for single items; crawling is not supported.

Following is an example of a Manifest file for hybrid ingest content:

```
<CdnManifest>
<server name="web-server">
<host name="http://www.company-web-site.org" />
</server>
<item src="/tmp/ceo-talk.wmv" type="cache"
    serveStartTime="2007-01-12 14:00:00 PST"
    serveStopTime="2007-04-12 14:00:00 PST"
>
</item>
</CdnManifest>
```

Note For type="cache", <host> and <server> tags are not used.

Note Currently, only *serveStartTime* and *serveStopTime* are supported for type=”cache.”

Manifest Validator Utility

Because correct Manifest file syntax is so important to the proper deployment of prefetched content on your CDS network, Cisco makes available a Manifest file syntax validator. The Manifest Validator, a Java-based command-line interface that verifies the correctness of the syntax of the Manifest file you have written or modified, is built into the CDSM.

The Manifest Validator utility tests each line of the Manifest file to identify syntax errors where they exist and determine whether or not the Manifest file is valid and ready for use in importing content into your CDS network.

Running the Manifest Validator Utility

To access the Manifest Validator, do the following:

Step 1 Choose **Services > Service Definition > Delivery Services > Tools > Manifest Validator**.

Note You must first create a new delivery service or edit an exiting delivery service before you can access the Manifest Validator.

Step 2 In the **Manifest File** field, enter the URL of the Manifest file you want to test.

Step 3 Click **Validate**.

The Manifest Validator checks the syntax of your Manifest file to make sure that source files are named for each content item in the Manifest file. It then checks the URL for each content item to verify that the content is placed correctly and then displays the output in the lower part of the page. The Manifest Validator does not determine the size of the item.

Alternatively, click **Validate** in the Delivery Service Content page. The results are displayed in a new window.

Valid Manifest File Example

The following text is an example of a valid Manifest file:

```
<CdnManifest>
<item
        src="tmp/mao's.html"
        priority="20"
        />
<server name="my-dev'box">
<host name="http://128.107.150.26"
        proto="http" />
</server>

<item
        src="tmp/lu.html"
        priority="300"
        />
<item
        src="/tmp/first_grader.html"
        />
<server name="server0">
       <host name="http://umark-u5.cisco.com:8080/" />
</server>
<item  •src="a.gif"/>
<server name="server1">
       <host name="http://unicorn-web" />
</server>
<item  •src="Media/wmtfiles/DCA%20Disk%201/Microsoft_Logos/Logos_100k.wmv" />

</CdnManifest>
```

The final lines of the Manifest Validator output indicate whether the Manifest file is valid or not. Wait until the following message is displayed, indicating that the validator has completed processing the Manifest file:

```
Total Number of Error: 0
Total Number of Warning: 0
Manifest File is CORRECT.
```

If errors are found, the error messages reported appear before the preceding message.

Invalid Manifest File Example

The following text is an example of an invalid Manifest file:

```
<CdnManifest>
<item
        src="tmp/mao's.html"
        priority="20"
        />
<server name="my-dev'box">
<host name="http://128.107.150.26"
        proto="http" />
</server>
<item
        src="tmp/lu.html"
        priority="300"
        />
<item
        src="/tmp/first_grader.html"
        />
<server name="server0">
       <host name="http://umark-u5.cisco.com:8080/" >
</server>
<item src="a.gif"/>
<server name="server1">
       <host name="http://unicorn-web" />
</server>
<item src1="Media/wmtfiles/DCA%20Disk%201/Microsoft_Logos/Logos_100k.wmv" />
</CdnManifest>
```

In the preceding example, although there are no warnings, two errors are found, and this Manifest file is syntactically incorrect, as shown in the following message:

```
ERROR (/state/dump/tmp.xml.1040667979990 line: 23 col: 1 ):No character data is allowed by
content model
ERROR (/state/dump/tmp.xml.1040667979990 line: 23 col: 9 ):Expected end of tag 'host'
  Manifest File: /state/dump/tmp.xml.1040667979990
  Total Number of Error: 2
  Total Number of Warning: 0
  Manifest File is NOT CORRECT!
```

The following full-text output is an example of the invalid Manifest file after the Manifest Validator checks the file:

```
Manifest validated: http://qiwzhang-lnx/nfs-obsidian/Unicorn/my-single-bad.xml
The manifest is downloaded as /state/dump/tmp.xml.1040667979990 for validation, this file
will be removed when validation is completed.
Start CdnManifest
Start item
     priority=20
     src=tmp/mao's.html
End item

Start server
     name=my-dev'box
Start host
     name=http://128.107.150.26
     proto=http
     uuencoded=false
End host

End server

Start item
```

```
     priority=300
     src=tmp/lu.html
End item

Start item
     src=/tmp/first_grader.html
End item

Start server
     name=server0
Start host
     name=http://umark-u5.cisco.com:8080/
     uuencoded=false
ERROR (/state/dump/tmp.xml.1040667979990 line: 23 col: 1 ):No character data is allowed by
content model
ERROR (/state/dump/tmp.xml.1040667979990 line: 23 col: 9 ):Expected end of tag 'host'
Manifest File: /state/dump/tmp.xml.1040667979990
Total Number of Error: 2
Total Number of Warning: 0
Manifest File is NOT CORRECT!
```

Understanding Manifest File Validator Output

The Manifest Validator messages appear below the Manifest File in the Manifest Validator page.

Each output file has a similar structure and syntax. It clearly identifies any errors or warning messages arising from incorrect Manifest file syntax. Manifest files are determined by the validator to be either:

- CORRECT—Contains possible syntax irregularities but is syntactically valid and ready for deployment on your CDS network
- INCORRECT—Contains syntax errors and is unsuitable for deployment on your CDS network

Syntax Errors

The Manifest Validator issues syntax errors only when it cannot identify a source file for a listed content item, either because it is not listed or because it is listed using improper syntax. Files containing syntax errors are marked INCORRECT.

Syntax errors are identified in the output with the ERROR label. In addition to the label, the line and column number containing the error are provided, as well as the Manifest file attribute for which the error was issued. An error appears in the following example:

```
ERROR (/state/dump/tmp.xml.1040667979990 line: 23 col: 1 ):No character data is allowed by
content model
```

In the error example:

- `/state/dump/tmp.xml.1040667979990` is the Manifest file name
- `line: 23 col: 1` is the Manifest file line and column number where the error occurs
- `No character data is allowed by content model` describes the type of Manifest file error

Syntax Warnings

The Manifest Validator issues syntax warnings for a wide variety of irregularities in the Manifest file syntax. Files containing syntax warnings may be marked CORRECT or INCORRECT, depending on whether or not syntax errors have also been issued.

Syntax warnings are identified in the output with the WARNING label. In addition to this warning label, the line number for which the warning is issued is provided, as well as the Manifest file attribute, valid options, and the default value for that attribute for which the warning was issued.

Correcting Manifest File Syntax

Once you have identified syntax warnings, errors, and messages using the output from the Manifest Validator, you can correct your Manifest file syntax and then rerun the Manifest Validator on the corrected file to verify its correctness.

It is a good idea to review every warning and error in your Manifest file. Some warnings, although they still allow the Manifest Validator to find your Manifest file syntax to be correct, can be the source of problems when you deploy the identified content to your CDS network.

Manifest File Structure and Syntax

The CDS Manifest file provides powerful features for representing and manipulating CDS network data that can be easily edited using any simple text editor.

Table B-4 provides a summary list of the Manifest file tags, their corresponding attributes and subelements, and a brief description of each tag. Table B-5 shows an example of how tags are nested in a Manifest file. The sections that follow provide a more detailed description of the Manifest file tags, the data they contain, and their attributes.

Table B-4 Manifest File Tag Summary

Tag Name	Subelements	Attributes	Description
CdnManifest	<playServerTable/> <options/> <server/> <item/> <item-group/> <crawler/>	None	Marks the beginning and end of the Manifest file content.
playServerTable	<playServer/>	None	(Optional) Sets default mappings for media types.
playServer	<contentType/> <extension/>	***name***[1]	Names the media server type on the Service Engine responsible for playing content types and files with extensions mapped to it using <contentType> tags.
contentType	None	***name***	(Optional, but must have either <contentType> or <extension> tag.) Names the MIME-type content mapped to a playserver.
extension	None	***name***	(Optional, but must have either <contentType> or <extension> tag.) Names the file extension that is mapped to a playserver.

Table B-4 Manifest File Tag Summary (continued)

Tag Name	Subelements	Attributes		Description
options	<schedule/> <repeat/>	*enableCookies* *expires* *failRetryInterval* *ignoreOriginPort* *ignoreQueryString*	*prefetch* *priority* *wmtRequireAuth* *server* *sslAuthtype* *timeZone* *ttl* *type*	(Optional) Defines attributes specific to the Manifest file that can be shared.
server	<host/>	***name***		Defines *only* one host from which content is to be retrieved.
host	None	***name*** *disableBasicAuth* *noProxy* *ntlmUserDomain* *password* *port* *proto*	*proxyServer* *sslAuthType* *user* *userDomainName* *uuencoded*	Defines a web server or live server from which content is to be retrieved and later prefetched. The hostname can be specified as: proto://user:password@hostname :port
proxyServer	None	***serverName*** *disableBasicAuth* *ntlmUserDomain* *password*	*port* *user* *uuencoded*	Specifies proxy server information.
item	<contains/> <schedule/> <repeat/>	***src*** *authCookie* *cdn-url* *disableBasicAuth* *enableCookies* *expires* *failRetryInterval* *host* *ignoreOriginPort* *ignoreQueryString* *noProxy* *ntlmUserDomain* *password* *playServer* *port*	*prefetch* *priority* *proto* *proxyServer* *server* *serveStartTime* *serveStopTime* *sslAuthType* *ttl* *type* *user* *userDomainName* *uuencoded* *wmtRequireAuth*	Identifies specific content that is to be acquired from the origin server.

Table B-4 Manifest File Tag Summary (continued)

Tag Name	Subelements	Attributes		Description
crawler	<matchRule/> <schedule><repeat>	***start-url*** *accept* *authCookie* *cdnPrefix* *depth* *disableBasicAuth* *enableCookies* *expires* *externalPrefixes* *externalServers* *failRetryInterval* *host* *ignoreOriginPort* *ignoreQueryString* *keepExpiredContent* *keepFolder* *keepNoCacheContent* *keepQueryUrl* *max-number* *maxTotalSizeIn-MB* *noProxy* *ntlmUserDomain*	*password* *playServer* *port* *prefetch* *prefix* *priority* *proto* *proxyServer* *reject* *reportBrokenLinks* *serveStartTime* *serveStopTime* *server* *srcPrefix* *sslAuthType* *ttl* *type* *user* *userDomainName* *uuencoded* *wmRequireAuth*	Supports crawling of a website or FTP server.
item-group	<item/> <crawler/> <item-group/>	*cdnPrefix* *cdn-url* *disableBasicAuth* *enableCookies* *expires* *failRetryInterval* *host* *ignoreOriginPort* *ignoreQueryString* *noProxy* *password* *playServer* *prefetch* *priority*	*proto* *proxyServer* *requireAuth* *serveStartTime* *serveStopTime* *server* *srcPrefix* *sslAuthType* *ttl* *type* *user* *userDomainName* *uuencoded* *wmtRequireAuth*	Places shared attributes under one tag so that they can be shared by every <item> and <crawler> tag within that group.
matchRule	<match>	None		(Optional) Defines additional filter rules for crawler jobs.
match	None	*extension* *mime-type* *prefix* *minFileSizeIn-B* *minFileSizeIn-KB* *minFileSizeIn-MB*	*maxFileSizeIn-B* *maxFileSizeIn-KB* *maxFileSizeIn-MB* *time-after* *time-before* *url-pattern*	(Optional) Specifies the acquisition criteria of content objects before they can be acquired by the CDS network.
contains	None	*cdn-url*		(Optional) Identifies content objects that are embedded within the content item currently being described.

1. Attributes that are required for a tag are shown in ***boldface italic*** font.

Table B-5 Manifest File Nested Tag Relationships

<CdnManifest>				
	<playServerTable> <playServer>			
		<contentType /> <extension />		
			</playServerTable> </playServer>	
	<options>			
		Manifest file shared attributes	</options>	
	<server>			
		<host/>		
			</server>	
	<item>			
		<contains />		
			</item>	
	<crawler>			
		<matchRule/>		
			</crawler>	
	<item-group>			
		<contains />		
			</item-group>	
				</CdnManifest>

CdnManifest

The<CdnManifest> </CdnManifest> tag set is required and marks the beginning and end of the Manifest file content. At a minimum, each <CdnManifest> tag set must contain at least one item, or content object, that is fetched and stored.

Attributes

None

Subelements

The <CdnManifest> tag set can contain the following subelements:

- playServerTable

  The <CdnManifest> tag set can contain only one playServerTable subelement.

- options

  The <CdnManifest> tag set can contain only one options subelement.
- server
- item
- item-group
- crawler

Example

```
<CdnManifest>
    <server name="origin-server">
                <host name="www.name.com" proto="http" port="80" />
    </server>
    <item cdn-url= "logo.jpg" server="originserver"  src= "images/img.jpg" type="prepos"
              playServer="http" ttl="300"/>
</CdnManifest>
```

playServerTable

The <playServerTable> </playServerTable> tag set is optional and provides a means for you to set default mappings for a variety of media types. Mappings can be set for both MIME-type content (the preferred mapping) and file extensions. Playserver tables allow you to override default mappings on the Service Engine for content types from a particular origin server. Playservers can be any one of the following streaming servers: WMT, HTTP, QTSS, or FMS. If no <playServerTable> tag is configured in the Manifest file, a default <playServerTable> tag is used.

Using the Manifest file, you can map groups of single items as well as individual content objects to an installed playserver. The following are content item and Manifest file playserver mappings:

- Content item URL

  Playserver mappings appear immediately after the origin server name in place of the default <CdnManifest> tag.
- Manifest file as an attribute of the <item> or <item-group> tag

  Playserver mappings placed at this location are identified using the *playServer* attribute and only apply to the named item or group of items.
- Manifest file as a playserver table

  Mappings are grouped within the <playServerTable> and <playServer> tags and are applied to content served from the origin server as directed by the Manifest file.
- System-level

  Playserver mappings are configured during CDS software startup.

The <playServerTable> tags are enclosed within the <CdnManifest> tags and name at least one of four playservers, such as RealServer, to which certain MIME-types and file extensions are mapped.

Attributes

None

Subelements

The <playServerTable> element must contain at least one <playServer> tag.

playServer

The <playServer> </playServer> tag set is required for the <playServerTable> tag and names the media server type on the Service Engine that is responsible for playing the content types and files with extensions mapped to it using the <contentType> tags. The <playServer> tag is enclosed within <playServerTable> tags.

Note Do not confuse the <playServer> tag with the *playserver* attribute in an <item> or <item-group> tag. An <item> or <item-group> tag specifies a server type to be used for an individual content object or group of related content objects. Although both playserver settings accomplish the same task, <item> tag-level playserver settings take precedence over the content type and file extension mappings specified by the <playServer> tags in the <playServerTable> tag.

Attributes

The <playServer> tag name is required. Each <playServer> tag names the type of server to which content is mapped using the *name* attribute. The Service Engines support the following types of playservers:

- http: HTTP web server
- qtss: Apple QuickTime Streaming Server
- wmt: Microsoft Windows Media Technologies
- fms: Flash Media Streaming Server

Subelements

At least one of the following subelements must be present in a <playServer> tag set.

- <contentType />
- <extension />

contentType

The <contentType /> tag is optional, but either a <contentType /> or an <extension /> subelement must be present in a <playServer> tag set. The <contentType /> tag names MIME-type content that is to be mapped to a playserver. The <contentType /> tag must be enclosed within a <playServer> tag set. When both <contentType /> and <extension /> tags are present in a <PlayServerTable> tag for a particular media type, the <contentType /> mapping takes precedence.

Attributes

Each <contentType /> tag names a media content type that is to be mapped to the playserver using the *name* attribute. The *name* attribute is required.

Note The <contentType /> *name* value cannot exceed 32 characters.

Subelements

None

extension

The <extension /> tag is optional but either a <contentType /> or an <extension /> subelement must be present in a <playServer> tag set. The <extension /> tag names the file extension that is being mapped to a playserver.

The <extension /> tag follows the <playServer> tag. When both <contentType /> and <extension /> tags are present in the <playServer> tag for a particular media type, the <contentType /> mapping takes precedence.

Attributes

The *name* attribute is required and provides the file extension for a mapped content type. When files with the named extension are requested, the mapped playserver is used to serve them.

Subelements

None

Example

```
<CdnManifest>
<playServerTable>
<playServer name="wmt">
         <extension name="asf" />
</playServer>
<playServer name="http">
         <contentType name="application/pdf" />
         <contentType name="application/postscript" />
         <extension name="pdf" />
         <extension name="ps" />
</playServer>
</playServerTable>
<server name="test.origin.com/">
         <host name="http://tst.orgn.com" proto="http" />
</server>
<item
    src="pic1.mpg"
/>
</CdnManifest>
```

options

The <options/> tag is optional and used to define attributes specific to the Manifest file. Shared attributes can be inherited by <item> and <crawler> tags in the Manifest file. For example, *timeZone* is an attribute specific to the Manifest file that is used to set the time zone for all time-related values. Attributes such as *ttl* can exist as <options/> tags, and their values can be shared by all <item> and <crawler> tags within the Manifest file.

The <options/> tag set is enclosed within the <CdnManifest> tag set and specifies at least one global setting.

Note No more than one <options> tag is allowed per Manifest file.

If parameters are defined within the Manifest file <options/>, <item-group>, or <item> tags, the order of precedence from lowest to highest is <options/>, <item-group>, and <item>.

Attributes

The *timeZone* attribute specifies the time zone for time values of attributes such as *expires* and *prefetch*.

The following list of attributes can be shared by <item> and <crawler> tags. See the "item" section on page B-29 for descriptions of the following attributes:

- *enableCookies*
- *expires*
- *failRetryInterval*
- *ignoreOriginPort*
- *ignoreQueryString*
- *prefetch*
- *priority*
- *wmtRequireAuth*
- *server*
- *sslAuthType*
- *ttl*
- *type*

Subelements

<schedule><repeat>

(See the "item" section on page B-29 for descriptions of these subelements.)

server

The <server> and <host> tag fields configure the origin content source server. The <host> tag field inside the <server> tag field configures the content source host. Having multiple <host> tag fields in one <server> tag field is not supported.

Each <item> or <item-group> tag can have a *server* attribute that refers to this <server> tag field. The <server> </server> tag set is required and defines only one host from which content is to be retrieved. The <server> tags are contained within <CdnManifest> tags and contain one <host> tag that identifies the host from which content is retrieved.

Attributes

The *name* attribute is required and can be any name as long as it matches the *server* attribute values in the <item> or <crawler> tags.

Subelements

The <server> tag set can only contain one <host/> subelement.

host

The <host/> tag is required and defines a web server or live server from which content is to be retrieved and later prefetched. Only one host can be defined within a single <server> tag set. The <host/> tag must be enclosed within <server> tags.

Attributes

- *disableBasicAuth*

  The *disableBasicAuth* attribute is optional; if specified, basic authentication is disabled.

- *name*

  The *name* attribute is required and identifies the domain name or IP address of the host, unless the *proto* attribute field is empty. If the *proto* attribute field is empty, the *name* attribute must be a fully qualified URL, including scheme and domain name or IP address. It can also include subdirectories, such as http://www.abc.com/media.

  The *name* attribute can also contain the UNC path to an SMB server; for example, \\SMBserver\directory\.

- *noProxy*

  The *noProxy* attribute is optional. If set to true, no proxy is used for the origin server. The default is false.

- *ntlmUserDomain*

  The *ntlmUserDomain* attribute is optional and specifies the user domain name for NTLM authentication.

- *password*

  The *password* attribute is optional and identifies the password for the user account that is required to access the host server.

- *port*

  The *port* attribute is optional and identifies the TCP port through which traffic to and from the host passes. The port used depends on the protocol used. The default port for HTTP is 80. The *port* attribute is only required for a nonstandard port assignment. The port attribute can also be specified in the *name* attribute, such as name="http://www.cisco.com:8080/."

- *proto*

  The *proto* attribute is optional and identifies the communication protocol that is used to fetch content from the host. Supported protocols are HTTP, HTTPS, MMS-over-HTTP, or FTP. The default *proto* attribute is HTTP. The *proto* attribute can be empty if the *name* attribute is a fully qualified domain name (FQDN).

- *proxyServer*

  The *proxyServer* attribute is optional and specifies which proxy server to use if there are multiple <proxyServer> tags in the Manifest file. If no proxy server is specified, the server in the closest <proxyServer> tag is used.

- *sslAuthType*

  The *sslAuthType* attribute is optional and has two possible values for the type of SSL certificate verification:

  - *strong*—Strong authentication. If any errors occur during certificate verification by the acquirer module, content from that site is not acquired. The default *sslAuthType* attribute setting is strong.
  - *weak*—Weak authentication. If certain errors occur during certificate verification by the acquirer module, content from that site continues to be acquired. These errors are as follows:

    Unable to decode 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 Content Acquirer is not marked as trusted

Unable to verify the first certificate

Certificate is not yet valid

Certificate has invalid purpose

- *user*

  The *user* attribute is optional and identifies the secure login used for host access.

- *userDomainName*

  See the "item" section on page B-29 for a description of this attribute.

- *uuencoded*

  The *uuencoded* attribute is optional. If set to true, the password is not encoded. The *uuencoded* attribute default setting is false.

Subelements

None

proxyServer

The <proxyServer> tag specifies proxy server information. The <proxyServer> tag must be located at the top level of the Manifest file, directly under the <CdnManifest> tag; it cannot be used as a subtag of any other tags, as shown in this example:

```
<CdnManifest>
<proxyServer>
...
</CdnManifest>
</proxyServer>
```

Attributes

- *disableBasicAuth*

  The *disableBasicAuth* attribute is optional; if specified, basic authentication is disabled.

- *ntlmUserDomain*

  The *ntlmUserDomain* attribute is optional and specifies the user domain name for NTLM authentication.

- *password*

  The *password* attribute is optional and identifies the password for the user account that is required to access the proxy server.

- *port*

  The *port* attribute is optional and specifies the proxy port.

- *serverName*

  The *serverName* attribute is required and identifies the domain name or IP address of the proxy server.

- *user*

  The *user* attribute is optional and identifies the secure login used for proxy authentication.

- *uuencoded*

  The *uuencoded* attribute is optional and designates whether the password is to be encoded.

Subelements

None

item

The <item> </item> tag set identifies the specific content that is to be acquired. The <item> tag names a single piece of content or a content object on the origin server, such as a graphic, MPEG video, or RealAudio sound file. Content items can be listed individually or grouped using the <item-group> tag.

The <item> tag must be enclosed within the <CdnManifest> tag set and can also be enclosed within <item-group> tags.

Attributes

- *src*

  The *src* attribute is required and identifies the URL from which to fetch the content. The URL can be a full URL or a relative URL. A full URL has the following format:

  proto://username:password@/domain-name:port/file-path/file-name

  Protocols supported in the *src* attribute are HTTP, HTTPS, FTP, and SMB. For SMB, the URL must be written in UNC format (\\SMBserver\directory\file).

  If a relative path is used, the <server> and <host> tags are required to specify origin server information, as shown in this example:

```
<item src="http://user:password@www.cisco.com/HR/index.html" />
<server name="ftp-server" >
   <host name="ftp://ftp-server"  user="johw" password="wwww" />
</host>
<item src="data/video.asf" />
```

  Note A URL containing a question mark (?) is not supported. A Manifest file parsing error occurs if you specify a URL that contains a question mark.

  Note A URL containing a pound sign (#) is modified. All characters that follow a pound sign are discarded, including the pound sign itself.

- *host*

  The *host* attribute specifies the hostname if the source URL of the *src* attribute is a relative URL.

- *server*

The *server* attribute is optional and refers to the server name in the <server> tag. If the *server* attribute is omitted, the server listed in the closest <server> tag is used. If there is no <server> tag close to this item, the Manifest file server is used.

- *cdn-url*

  The *cdn-url* attribute is optional and is used when content needs to be acquired from one URL (the content acquisition URL) and published using another URL (the publishing URL). The *cdn-url* attribute is the relative CDS network URL that end users use to access this content. If no *cdn-url* attribute is specified, then the *src* attribute is used as the relative CDS network URL.

  In the following sample Manifest file, the content item being acquired contains the file path /RemAdmin/InternalReview/firstpage.htm. By specifying a new file path (RemAdmin/Production/firstpage.htm) using the *cdn-url* attribute, the publishing URL disguises the fact that the content originated from an "Internal Review."

```
<CdnManifest>
<server name="ultra-server">
    <host name="http://ultra-server" />
</server>
<item src="RemAdmin/InternalReview/firstpage.htm"
cdn-url="RemAdmin/Production/firstpage.htm" />
</CdnManifest>
```

  In the preceding example, *src is* the content acquisition URL and *cdn-url is* the publishing URL.

Note The content item file path (RemAdmin/InternalReview/firstpage.htm) is controlled by the Manifest file. The *cdn-url* attribute associates a file path with the content item in the Manifest file. The Manifest file allows the file path for the *cdn-url* attribute to be specified independently of the file path from which the content items are to be acquired from the origin server (*src* attribute), allowing the publishing URL to differ from the content acquisition URL.

  If the content requires playback authentication or is live content, the origin server from which the content is acquired has to be contacted. Therefore, two URLs must exist for the same content item, and the URL specified in the *cdn-url* attribute must exist on the origin server at all times.

  For example, if the content item "RemAdmin/Production/firstpage.htm" in the preceding example requires playback authentication, this content must exist on the "ultra-server" origin server. Otherwise, prefetched content playback fails.

  In general, you should not use the *cdn-url*, *cdnPrefix*, or *srcPrefix* attributes if playback authentication is required or if the content is live.

  If you use FTP to acquire content and the content type is not specified in the Manifest file and the *cdn-url* attribute is specified to alter your publishing URL, the *cdn-url* attribute must have the correct file path extension. Otherwise, the incorrect content type is generated and you cannot play the content.

  The following example correctly shows the publishing URL with the same file path extension (.jpg) as the origin server URL.

```
<item src="ftp://ftp-server.abc.com/pictures/pic.jpg"  cdn-url="pic.jpg" />
```

  The following example is incorrectly written, because it does not specify the file path extension (.jpg) in the *cdn-url* attribute.

```
<item src="ftp://ftp-server.abc.com/pictures/pic.jpg"  cdn-url="pic" />
```

- *type*

  The *type* attribute is optional and defines whether content is to be prefetched or live on the CDS network. The three *type* attributes are *prepos*, *cache*, and *wmt-live*. The *wmt-live type* attribute is used to deliver live content. The *cache* type corresponds to hybrid ingest method. If this field is left blank, the default type is *prepos*.

  Note For type="cache", <host> and <server> tags are not used.

  Note Currently, only *serveStartTime* and *serveStopTime* are supported for the type="cache" attribute.

Note Existing live content is deleted and replaced with the content specified in the Manifest file under the following conditions:

- You create two delivery services that use the same content origin server in the following way:
  1. Create a live streaming delivery service by using the CDSM GUI.
  2. Use a Manifest file to set up live streaming on a prefetch/caching delivery service by using the *type* attribute with wmt-live as the value and wmt-live as the *src*.
- You assign an SE to the prefetch/caching delivery service with the live Manifest file assigned, the existing live content is overwritten with the content specified in the Manifest file if the program name is the same (in this example, wmt-live).

This is because the content is the latest assigned, whichever was assigned last, whether it was by the prefetch/caching delivery service or the live streaming delivery service.

- *playServer*

  The *playServer* attribute is optional and names the server used to play back the content. Valid playservers are wmt (Windows Media Technologies), qtss (QuickTime Streaming Server), fms (Flash Media Streaming), and http (Web Engine). The value in this field is either one playserver or multiple playservers separated by commas. If a value for this attribute is not specified, the <PlayServerTable> tag in the Manifest file is used to generate the playserver list for this content. If the Manifest file does not have the <PlayServerTable> tag specified, it uses the default <PlayServerTable> tag.

- *prefetch*

  The *prefetch* attribute is optional and specifies a time (in yyyy-mm-dd hh:mm:ss [year-month-day hour:minute:second] format) for the first content acquisition or re-check after the Manifest file is parsed. The time zone for the time can be specified in the <options> tag. Note that the autoconversion between daylight saving time and standard time within a time zone is not supported, but a special designation for daylight saving time can be used, such as PDT for Pacific daylight saving time. In the following example, the prefetch time is September 5, 2002 at 09:09:09 Pacific daylight saving time.

```
<options timeZone="PDT" />
<item src="index.html" prefetch="2002-09-05 09:09:09 PDT" />
```

This attribute is used when you want to specify a future time for the acquirer to begin fetching content from the origin server. When a future time is specified, the acquirer does not acquire content before this time; however, it checks content freshness during its scheduled *ttl* interval. If a *prefetch* time is omitted, the content is acquired immediately.

After the Manifest file is parsed, if any items or crawl tasks have changed or new ones have been added and if the *prefetch* attribute specifies a future time, the acquirer checks and fetches the content or re-crawls the crawl jobs at the time specified by the *prefetch* attribute.

- *expires*

  The *expires* attribute is optional and designates a time in yyyy-mm-dd hh:mm:ss format when the content is to be removed from the CDS network. Additionally, you can specify the GMT time zone. If a time value is omitted, content is stored until it is removed when you modify the relevant Manifest file code.

- *ttl*

  The *ttl* attribute is optional and designates a time interval, in minutes, for revalidation of the content. If a time value is omitted, the content is fetched only once and its freshness is never checked again. Usually the *ttl* attribute is a positive value; however, you can also assign a negative value to the *ttl* attribute. The following table describes *ttl* attribute value ranges.

Note Revalidation is enabled by default for the Web Engine.

***ttl* Attribute Value**	**Action**
ttl > 0	Content is rechecked every *ttl* minute. Content is also rechecked if the Manifest file is reparsed and the content specification in the Manifest file has changed or if you click the **Refetch** button.
ttl = 0	Content is fetched only once and never checked again. Content is only rechecked if the Manifest file is reparsed and the content specification in the Manifest file has changed or if you click the **Refetch** button.
ttl < 0	Content is fetched only once and never checked again. Content will *not* be rechecked if the Manifest file is reparsed or if you click the **Refetch** button.

- *serveStartTime*

  The *serveStartTime* attribute is optional and designates a time in yyyy-mm-dd hh:mm:ss format when the CDS software is allowed to start serving the content. If the time to serve is omitted, content is ready to serve once it is distributed to the Service Engine or other edge device.

- *serveStopTime*

  The *serveStopTime* attribute is optional and designates a time in yyyy-mm-dd hh:mm:ss format when the CDS software temporarily stops serving the content. If the time to stop serving is omitted, the CDS software serves the content until it is removed when you modify the relevant Manifest file code.

- *priority*

  The *priority* attribute is optional and can be any integer value to specify the content processing priority. If a priority value is omitted, its index order within the Manifest file is used to set the priority.

- *wmtRequireAuth*

The *wmtRequireAuth* attribute is optional and determines whether users need to be authenticated before the specified content is played. When true, the Service Engine requires authentication to play back the specified content to users and communicates with the origin server to check credentials. If the requests pass the credential check, the content is played back from the Service Engine. If this attribute is omitted, a heuristic approach is used to determine the value: if the specified content is acquired by using a username and password, *wmtRequireAuth* is set to true; otherwise, it is set to false. For FTP, if the username is anonymous, *wmtRequireAuth* is set to false.

Note If *wmtRequireAuth* is true, the Origin Server field in the Content Origin page for this delivery service needs to point to the server that can authenticate users. When users want to play back the content, the server specified in the Origin Server field is checked for authentication.

- *failRetryInterval*

  The *failRetyryInterval* attribute specifies the retry interval, in minutes, when content acquisition fails. For example, failRetryInterval="10" means the CDS software retries content acquisition every 10 minutes after acquisition has failed. If the retry universal value is not specified, the default value is 5 minutes. (The minimum *failRetryInterval* value is accepted.) If a value of less than 5 minutes is specified, that value is converted to 5 minutes.

  The behavior differs between failed content acquisition of a single item and failed content acquisition of a crawl item.

  – For single item failure:

    ```
    if ( ttl != 0, ttl < retryInterval)
    ```

    The item is rechecked in accordance with the *ttl* attribute. Otherwise, the item is rechecked at the interval specified in the *failRetryInterval* attribute.

  – For crawl item failure:

    ```
    if ( ttl != 0 and ttl < retryInterval )
    always re-crawl
    ```

    If some items are not acquired (excluding 300 and 400 series status error codes), only failed items are rechecked as specified in the *failRetryInterval* attribute.

    When the *ttl* attribute interval occurs, all pages are recrawled.

  For example, if *ttl* = 10, and *failRetryInterval* = 4, the following actions occurs:

Number of Minutes	Action
0	Crawl
4	Recheck failed
8	Recheck failed
10	Recrawl
14	Recheck
18	Recheck
20	Recrawl

- *ignoreQueryString*

  The *ignoreQueryString* attribute is a playback attribute that can be used with the <options>, <item-group>, <item>, and <crawler> tags. If the value is set to true, then CDS software ignores any string after a question mark (?) in the request URL for playback. If this attribute is omitted, then the default value is false.

  For example, content with the request URL url=http://web-server/foo has been prefetched. If a user requests content with the URL url=http://web-server/foo?id=xxx and the *ignoreQueryString* attribute value is false, then CDS software does not use the prefetched content from the request URL http://web-server/foo.

  However, if the *ignoreQueryString* attribute is set to true, then the CDS software treats the request URL http://www-server/foo?id=xxx the same as http://www-server/foo and returns with prefetched content.

  Note How content is cached for dynamic ingests depends on the *ignoreQueryString* value and the protocol engine serving the content.
  If Windows Media Streaming is serving the content, and the *ignoreQueryString* is not set, the requested content is cached on the SE. If the *ignoreQueryString* value is set to true, Windows Media Engine caches the content on the SE. If the *ignoreQueryString* value is set to false, Windows Media Streaming does not cache the content on the SE.
  The Web Engine only supports the *ignoreQueryString* attribute for pre-positioned content. If the Web Engine is serving the content, and the *ignoreQueryString* attribute is not set, the requested content is not cached on the SE.

- *ignoreOriginPort*

  The *ignoreOriginPort* attribute allows playback of prefetched content from a port other than the standard port. If the *ignoreOriginPort* attribute is set to true, content can be played back without regard to the port specified in the request URL. The default for this attribute is false.

  This attribute is not intended to be used for content that is routed using a Service Router. It is intended to work only for explicit proxy routing. A typical usage scenario for the *ignoreOriginPort* attribute might be as follows:

  - The origin web server is not using port 80; it is using a nonstandard port number in the URL.
  - Users are using explicit proxy routing, where the original URL containing the non-standard port number is used for playback from the Service Engine.

  Prefetched content cannot be played back using a nonstandard port; prefetched content is served only on ports that are standard for the protocol. If the incoming URL contains a port number other than the protocol's standard port, you must set the *ignoreOriginPort* attribute to true for playback to succeed.

- *userDomainName*

  The *userDomainName* attribute is used in two instances: for NTLM authentication and for the SMB file import feature. If the origin server is using NTLM authentication, you must use this attribute to specify the user domain name for NTLM authentication. If a shared folder is protected and the user account is part of a domain, you must use this attribute to specify the domain name of the configured shared folder.

  Note Both *userDomainName* and *ntlmUserDomain* cannot coexist in the Manifest file; only one attribute can be used at a time.

- *enableCookies*

  The *enableCookies* attribute enables cookie support for the item. When this attribute is set to true, the Content Acquirer, after sending a request for an item to the origin server, parses the server response for cookie name/value pairs. If the server response contains a cookie that is valid and has not expired, the Content Acquirer stores the cookie in main memory.

  The Content Acquirer then returns the valid cookie to the server the next time the Content Acquirer sends a request for the item.

  A cookie is rejected if it contains any of the following rejection criteria, as found in RFC 2965:

  - The value for the Path is not a prefix of the request URI.
    For example, if the request is www.abc.com/aaa/bbb/ccc.html and the Path of the cookie returned is /aaa/ccc, then it is not valid because /aaa/ccc is not a prefix of /aaa/bbb/ccc [URL].
  - The value for the Domain contains no embedded dots or does not start with a dot.
  - The value for the request host is not a domain-match of the Domain.
  - The request host is a FQDN (not an IP address) and has the form HD, where D is the value of the Domain and H is a string that contains one or more dots.
  - The Path is not a prefix match of the request URL.

  Note The Content Acquirer does not use persistent memory to store cookies. If the Service Engine is restarted, all cookie information is lost.

  The *enableCookies* attribute can be used with the <item>, <crawler>, <item-group>, and <options> tags.

- *authCookie*

  The *authCookie* attribute enables the processing and sending of authentication cookies for the item. To enable this feature, the *authCookie* attribute must be set to true for the particular item that passes the user credentials and for which the server sends back the authentication cookies.

  The *authCookie* attribute can be used with the <item> and <crawler> tags. For example:

  <item src=http://abc.com/auth.cgi?id=10000 authCookie="true"/>

The following attributes described under the <host> tag attributes can also be specified by the <item> tag.

- *disableBasicAuth*
- *noProxy*
- *ntlmUserDomain*
- *password*
- *port*
- *proto*
- *proxyServer*
- *sslAuthType*
- *user*
- *uuencoded*

Subelements

- <contains />
- <schedule/> <repeat/>

  The <schedule/> <repeat/> subelement and its attributes specify a time for a recrawl or an item refetch to begin. You can have multiple <repeat> subelements under the <schedule> subelement. The attributes *time, start,* and *end* specify the day of the month or day of the week and the duration of the specified repeat. The *time* attribute is required, whereas *start* and *end* are optional attributes.

  Note The <schedule> element takes precedence over the *ttl* attribute.

  The *time* attribute uses either of the following formats:

  time=“dom:hh:mm” or

  time=“dow:hh:mm”

  In these formats, dom is the day of the month (0–30), dow is the day of the week (Sun, Mon, Tue, Wed, Thu, Fri, Sat, or *), hh is the clock hour (0–23 or *), and mm is the minute (0–59).

  For example:

```
<schedule>
    <repeat time="*:*:0" /> <!-- repeat every hour on the hour -->
    <repeat time="*:13:0" /><!-- repeat at 1300 every day -->
    <repeat time="Sun:2:30" /> <!-- repeat on Sundays at 2:30 -->
    <repeat time="4:2:30" /> <!-- repeat at 2:30 on the fourth day of the month -->
    <repeat time="Mon:*:30" /> <!-- On Monday, repeat every hour at 30 minutes past
        the hour -->
</schedule>
```

  The *start* and *end* attributes use the following format:

  start=“yyyy-mm-dd hh:mm:ss”

  end=“yyyy-mm-dd hh:mm:ss”

  For example:

```
<CdnManifest>
<item>
    <schedule>
        <repeat time="Sun:02:30" />
        <repeat time="*:*:34" start="2003-09-11 11:11:11 PST" end="2004-09-11 11:11:21
PST"/>
        <repeat time="21:02:35" start="2003-09-11 11:11:11 PST" end="2004-09-11
11:11:21 PST"/>
        <repeat time="21:02:35" end="2004-09-11 11:11:21 PST"/>
    </schedule>
</item>
.
.
.
```

Example

```
<item
    src="index.html"
    server="cisco.com"
    ttl="3000"
/>
```

crawler

The <crawler> </crawler> tag set supports crawling a website or an FTP server.

Attributes

- *start-url*

  The *start-url* attribute is required. It defines the URL at which to start the process of crawling the website or FTP server. It is identical to the *src* attribute used in the <item> tag. (See the "src" subsection in the "item" section on page B-29.)

- *host*

  The *host* attribute specifies the host name if the starting URL specified in the *start-url* attribute is a relative URL.

- *depth*

  The *depth* attribute is optional and defines the link depth to which a website is to be crawled or directory depth to which an FTP server is to be crawled. If the depth is not specified, the default is 20. The following are the general depth values:

  0 = Acquire only the starting URL
  1, 2, 3, ... = Acquire the starting URL and its referred files
  –1 = Infinite or no depth restriction

  Depth is defined as the level of a website or the directory level of an FTP server, where 0 is the starting URL.

- *prefix*

  The *prefix* attribute is optional and combines the hostname from the <server> tag with the value of the *prefix* attribute to create a full prefix. Only content with URLs that match the full prefix is acquired, as shown in this example:

  ```
  <server name="xx"> <host name="www.cisco.com" proto="https" port=433 /> </server>
  ```

  and with the following <crawler> tag:

  ```
  prefix="marketing/eng/"
  ```

  The full prefix is "https://www.cisco.com:433/marketing/eng/." Only URLs that match this prefix are crawled.

  If a prefix is omitted, the crawler checks the default full prefix, which is the hostname portion of the URL from the server. In the example, the default full prefix is "https://www.cisco.com:433."

- *accept*

  The *accept* attribute is optional and uses a regular expression to define acceptable URLs to crawl in addition to matching the prefix. For example, accept="stock" means that only URLs that meet two conditions are searched: the URL matches the prefix and contains the string "stock." (See the "Writing Common Regular Expressions" section on page B-6 for more information on using regular expressions.)

  Note the following two key differences between the *accept* attribute and the *prefix* attribute:

  – The *prefix* attribute uses an exact string match, while the *accept* attribute uses a regular expression.

  – The *prefix* attribute applies to a URL including all its links or subdirectories. However, the *accept* attribute allows the URL and its links and subdirectories to be evaluated separately.

- *reject*

  The *reject* attribute is optional and uses a regular expression to reject a URL if it matches the reject regular expression. The reject regular expression is checked after checking for a prefix URL match. If a URL does not match the prefix, it is immediately rejected. If a URL matches the prefix and the reject parameters, it is rejected by the particular reject constraint. (See the "Writing Common Regular Expressions" section on page B-6 for more information on using regular expressions.)

  Note the following two key differences between the *reject* attribute and the *prefix* attribute:

  - The *prefix* attribute uses an exact string match, while the *reject* attribute uses a regular expression.
  - The *prefix* attribute applies to a URL including all its links or subdirectories. However, the *reject* attribute allows the URL and its links and subdirectories to be evaluated separately.

- *max-number*

  The *max-number* attribute is optional and specifies the maximum number of crawler job objects that can be acquired.

- *maxTotalSizeInB/KB/MB*

  The *maxTotalSizeInB/KB/MB* attribute is optional and specifies the maximum total content size in bytes, kilobytes, or megabytes that this crawler job can acquire. The size attribute can be expressed in megabytes (MB), kilobytes (KB), or bytes (B).

  This attribute replaces the *max-size-in-B/KB/MB* attribute, which continues to be supported for backward compatibility only.

- *srcPrefix*

  The *srcPrefix* attribute is optional and must be used in conjunction with the *cdnPrefix* attribute to form a relative CDS network URL. If a *srcPrefix* attribute is not specified, or if the prefix of the relative source URL does not match the *srcPrefix* attribute, then the relative CDS network URL is the *cdnPrefix* value combined with the relative source URL. For example, if these content objects have the same source URL prefix "acme/pubs/docs/online/Design/" and you want to replace this prefix with a simple "online/," then specify srcPrefix="acme/pubs/docs/online/Design/" and cdnPrefix="online/."

- *cdnPrefix*

  The *cdnPrefix* attribute is optional and must be used in conjunction with the *srcPrefix* attribute.

- *wmtRequireAuth*

  The *wmtRequireAuth* attribute is optional and determines whether users need to be authenticated before the specified content is played. When true, the Service Engine requires authentication to play back the specified content to users and communicates with the origin server to check credentials. If the requests pass the credential check, the content is played back from the Service Engine. If this attribute is omitted, a process of discovery approach is used to determine the value: if the specified content is acquired by using a username and password, *wmtRequireAuth* is set to true; otherwise, it is set to false. For FTP, if the username is anonymous, *wmtRequireAuth* is set to false.

Note If *wmtRequireAuth* is set to true, the Origin Server field in the Content Origin page for this delivery service needs to point to the server that can authenticate the users. When users want to play back the content, the server specified in the Origin Server field is checked for authentication.

- *externalPrefixes*

  The *externalPrefixes* attribute is optional and specifies additional prefixes for crawl jobs to crawl multiple protocols or multiple websites. Prefixes are separated with a bar (|).

- *externalServers*

  The *externalServers* attribute is optional and can be used for multiple host crawling jobs where each host has a different user account. This attribute can be used to see to the <host> tag with the proper authentication information.

- *keepExpiredContent*

  The *keepExpiredContent* attribute can be used to acquire content during an HTTP or HTTPS crawl that is expired. When this attribute is set to true, expired content is fetched. When this attribute is set to false, expired content is discarded. If this attribute is not specified, the default is false.

- *keepFolder*

  The *keepFolder* attribute is used to fetch folders (a folder is indicated when the request URL ends with a forward slash "/"). If this attribute is set to false, folder URLs are not acquired.

- *keepNoCacheContent*

  The *keepNoCacheContent* attribute can be used to acquire content during an HTTP or HTTPS crawl that would normally not be cached. When this attribute is set to true, the acquirer fetches the content even though the content contains an HTTP cache control header indicating that the content is not to be cached. If this attribute is not specified, the default is false.

- *keepQueryUrl*

  The *keepQueryUrl* attribute can be used to fetch URLs that contain "?" in the URL string. If this attribute is set to true, URLs with "?" are fetched during HTML parsing for a crawl job if the URL meets the other crawling criteria set forth in the Manifest file.

  This attribute is useful when you want to acquire content from a database, for example, where multiple files are differentiated in the portion of the URL string after the "?". When this attribute is not set, the portion of the URL after the "?" is discarded. If multiple URLs are found where the portion of the URL string in front of the "?" is the same, these URLs appear as duplicates, and only the last "duplicate" URL found is fetched.

- *reportBrokenLinks*

  The *reportBrokenLinks* attribute is used to report links on an HTML web page that cannot be fetched. If this attribute is set to true, all broken links encountered during a website crawl are reported as errors. This attribute only applies to a website crawl, not to an index crawl. The default is false and broken links are not reported as errors.

The following attributes described under the <host> tag attributes can also be specified by the <crawler> tag:

- *disableBasicAuth*
- *noProxy*
- *ntlmUserDomain*
- *password*
- *port*
- *proto*
- *proxyServer*
- *sslAuthType*

- *user*
- *uuencoded*

The following attributes described under the <item> tag attributes can also be specified by the <crawler> tag.

- *authCookie*
- *enableCookies*
- *expires*
- *failRetryInterval*
- *ignoreOriginPort*
- *ignoreQueryString*
- *playServer*
- *prefetch*
- *priority*
- *serveStartTime*
- *serveStopTime*
- *server*
- *ttl*
- *type*
- *userDomainName*

Subelements

- <matchRule></matchRule>
- <schedule><repeat>

  (See the "item" section on page B-29 for descriptions of the <schedule><repeat> subelements.)

Example

```
<server name="cisco">
    <host name="http://www.cisco.com/jobs/" />
</server>
<crawler
    server="cisco"
    start-url="eng/index.html"
    depth="10"
    prefix="eng/"
    reject="\.pl"
    maxTotalSizeIn-MB="200"
/>
```

item-group

The <item-group> </item-group> tag set is used to place shared attributes under one tag so that they can be shared by every <item> and <crawler> tag within that group. When attributes are shared, it means that attributes can be defined at either the <item-group> tag level for group-wide control or on a per

<item> or per <crawler> tag basis. For example, if every <item> tag is using the same *server* and *ttl* attributes, you can create an <item-group> tag on top of these <item> tags and place the *server* and *ttl* attributes in the <item-group> tag.

Using shared attributes makes any Manifest file with many <item> tags more efficient by consolidating the <item> tags with shared attributes. If the same attribute value exists in both the <item-group> and <item> tags, the value in the <item> tag takes precedence over that value in the <item-group> tag.

The <item-group> tag must be enclosed within the <CdnManifest> tag set and contain one or more <item> or <crawler> tags.

Attributes

If an attribute value is present only at the <item-group> tag level, then it is inherited by its inner element in the <item> tag. If an attribute value is present in a crawler job, its attributes, whether inherited or owned, are propagated to the content fetched by the crawler job.

The following attributes can be shared across many <item> and <crawler> tags and are candidates for the <item-group> level tag. See the "item" section on page B-29 for detailed descriptions of the following attributes:

- *cdn-url*
- *enableCookies*
- *expires*
- *failRetryInterval*
- *host*
- *ignoreOriginPort*
- *ignoreQueryString*
- *playServer*
- *prefetch*
- *wmtRequireAuth*
- *serveStartTime*
- *serveStopTime*
- *server*
- *priority*
- *ttl*
- *type*
- *userDomainName*

The following attributes described under the <host> tag attributes can also be specified by the <item-group> tag.

- *disableBasicAuth*
- *noProxy*
- *ntlmUserDomain*
- *password*
- *port*
- *proto*

- *proxyServer*
- *sslAuthType*
- *user*
- *uuencoded*

Additionally, the following two attributes can be placed within the <item-group> tag. See the "crawler" section on page B-37 for a detailed description of the following two attributes:

- *srcPrefix*
- *cdnPrefix*

These two attributes convert the prefix of the *src-url* (content acquisition URL) to the *cdn-url* (publishing URL) for multiple content objects. These content objects are either implicitly specified by multiple <item> tags or acquired through a crawler job.

These two attributes can also be specified in the <crawler> tag. If you explicitly specify the *srcPrefix* attribute and *cdnPrefix* attribute for an individual <crawler> job, the <crawler> tag-level specification takes precedence over the <item-group> tag-level settings. If you do not specify these attributes for an individual <crawler> job, the <item-group> tag-level specification is inherited by the <crawler> job.

The *srcPrefix* and *cdnPrefix* attributes generate the relative CDS network URL using the following rules:

- If the *cdn-url* attribute is present in the <item> tag, the relative CDS network URL contains both the *cdnPrefix* attribute plus the *cdn-url* attribute. For example, if *cdnPrefix*="eng/spec" and *cdn-url*="e/f.html," the relative path in the URL is "eng/spec/e/f.html."
- If the *srcPrefix* attribute is not present in the <item> tag, the relative CDS network URL is the *cdnPrefix* attribute plus the relative source URL.
- If the prefix of the relative source URL does not match the *srcPrefix* attribute, the relative CDS network URL is the *cdnPrefix* attribute plus the relative source URL.
- To generate a relative CDS network URL, remove the matched prefix from the relative source URL and replace it with the *cdnPrefix* attribute.

The relative CDS network URL of <item> in the following example is "acme/default.htm."

```
<item-group cdnPrefix="acme/" >
    <item src="design/index.html" cdn-url="default.html" />
</item-group>
```

In the following example, content objects with the *srcPrefix* attribute, such as "design/plan/," have the relative CDS network URL as "acme/" plus relative source URLs stripped of "design/plan/." Other content objects with a prefix attribute that does not match "design/plan/" have "acme/" plus their original relative source URL.

```
<crawler
    start-url="design/plan/index.html"
    depth="-1"
    srcPrefix="design/plan/"
    cdnPrefix="acme/" />
```

Subelements

- <crawler></crawler>
- <item-group/>
- <item></item>
- <schedule><repeat>

  (See the "item" section on page B-29 for descriptions of the <schedule><repeat> subelements.)

Example

```
<!--grouped content items-->
<item-group server="origin-web-server" type="prepos" ttl="300" cdnPrefix="unicorn/" >
         <item cdn-url="newHQpresentation.rm" src="newHQpresentation.rm" />
         <item cdn-url="animatedlogo.mpg" src="animlogo.mpg" />
         <item cdn-url="companytheme.mp3" src="cotheme.mp3" />
         <item cdn-url="newHQlayout.avi" src="newHQ.mov" />
</item-group>
```

matchRule

The <matchRule> </matchRule> tag set is optional and defines additional filter rules for crawler jobs. It affects only <crawler> tasks and is not used by single <item> tags. The crawler parameters defined in the <crawler></crawler> tag set determine primarily the scope of a crawl search. If a content object does not meet the criteria specified by the crawler parameter, neither it nor its children are searched.

The <matchRule> tag, however, determines only whether or not the content objects should be acquired regardless of the scope of the search. If a web page matches the crawler parameters without the <matchRule> feature, its children are searched even though its content objects are not acquired.

In the following crawler job example that uses the <matchRule> tag, the entire website is searched, but only files with the .jpg file extension larger than 50 kilobytes are acquired.

```
<crawler  start-url="index.html"  depth="-1" >
    <matchRule>
        <match minFileSizeIn-KB="50" extension="jpg" />
    </matchRule>
</crawler>
```

The <matchRule> element can be nested within an <item-group> tag to define group-wide filter rules for <crawler> tags contained in the group. It can also be a subelement of a particular <crawler> job. The <crawler> tag-level setting overrides the <item-group> tag-level setting when both tags are present.

Note The matchRule element is not supported for FTP; it is only supported for HTTP.

If you define criteria locally for individual <crawler> jobs, any existing group-level criterion is entirely discarded for that <crawler> job. If your <item-group> tag match rule is set to A and your <crawler> tag specifies another match rule set to B, only B is to be used for the <crawler> tag rather than a combination of A and B. You can define at most one <matchRule> tag per <item-group> tag and at most one <matchRule> tag per <crawler> tag.

Attributes

None

Subelements

At least one <match> tag

match

The <match> </match> tag is optional and specifies the acquisition criteria of content objects before they can be acquired by CDS software. Every attribute within a single <match> tag has a Boolean AND relationship (to form a logical conjunction) with the other attributes.

You can specify multiple <match> tags within the <matchRule> tag. The <match> tags have a Boolean OR relationship (to form a logical inclusion) with other <match> tags. You must specify at least one <match> tag per <matchRule> tag.

Attributes

- *mime-type*

  The *mime-type* attribute specifies MIME-types.

- *extension*

  The *extension* attribute specifies file extensions.

- *time-before*

  The *time-before* attribute can provide both an absolute time (modified before yyyy-mm-dd hh:mm:ss) or a relative time (modified within ddd:hh:ss), relative to the present time, to download content. Time parameters should be expressed in GMT time zones. (For GMT offsets, see the "Manifest File Time Zone Tables" section on page B-47.)

- *time-after*

  The *time-after* attribute can provide both an absolute time (modified after yyyy-mm-dd hh:mm:ss) or a relative time (modified within ddd:hh:ss), relative to the present time, to download content. Time parameters should be expressed in GMT time zones. (For GMT offsets, see the "Manifest File Time Zone Tables" section on page B-47.)

  Note Relative time is calculated based on current time. We recommend that you synchronize the the server clock and the Service Engine clock so that relative time calculations are accurate.

- *minFileSizeInB/KB/MB*

  The *minFileSizeInB/KB/MB* attribute specifies that the acquired content size must be larger than this number of bytes, kilobytes, or megabytes. The size attribute can be expressed in bytes (B), kilobytes (KB), or megabytes (MB).

  The *minFileSizeInB/KB/MB* attribute replaces the *size-min-in-B/KB/MB* attribute, which continues to be supported for backward compatibility only.

- *maxFileSizeInB/KB/MB*

  The *maxFileSizeInB/KB/MB* attribute specifies that the acquired content size must be smaller than this number of bytes, kilobytes, or megabytes. This attribute can be expressed in bytes (B), kilobytes (KB), or megabytes (MB).

  The *maxFileSizeInB/KB/MB* attribute replaces the *size-max-in-B/KB/MB* attribute, which continues to be supported for backward compatibility only.

- *prefix*

  The *prefix* attribute is optional and specifies a prefix as a match rule to filter out websites during a crawl job.

- *url-pattern*

  The *url-pattern* attribute is optional and specifies a regular expression as a match rule to filter out certain URLs.

Subelements

None

Examples

```
<! - - crawling item group -- >
<item-group server=”origin-server” type=”prepos”>
       <matchRule>
              <match time-before=”2000-05-05 12:0:0”/>
       </matchRule>
          <crawler start-url=”eng/index.html” depth=”-1”/>
            <crawler start-url=”hr/index.html” depth=”3”>
                       <matchRule>
                              <match minFileSizeIn-KB=”1” extension=”xxx”/>
                       </matchRule>
            </crawler>
</item-group>
```

To download content that was created or modified within the last 90 days, use the relative time format, as shown in the following example:

```
<match time-after=”90:00:00”/>
```

To download content that was not modified within the last 2 weeks, use the relative time format, as shown in the following example:

```
<match time-before=”14:00:00”/>
```

To download content that has been modified after January 30, 2003, 10:30 p.m., use the absolute time format, as shown in the following example:

```
<match time-after=”2003-01-30 10:30:00”/>
```

contains

The <contains /> tag is optional and identifies content objects that are embedded within the content item currently being described. For example, the components of a Synchronized Multimedia Integration Language (SMIL) file request for an item using <contains /> links are only accepted after CDS software determines that dependent content objects are present in the Service Engine.

The <contains /> tag must be enclosed within the <item> </item> tag.

The <contains /> tag is used to include embedded files for some video files, such as .asf or .rp. The CDS software does not serve this item unless every contained item is present.

Attributes

The *cdn-url* attribute is required and is the relative CDS network URL of one of the embedded contents.

Subelements

None

Example

```
<item src=”house/img08.jpb” cdn-url=”img08.jpg” />
<item src=”house/img09.jpb” cdn-url=”img09.jpg” />
<item cdn-url="house.rp"src="house/house.rp">
     <contains cdn-url="img08.jpg"/>
     <contains cdn-url="img09.jpg"/>
</item>
```

XML Schema

In the case of the Manifest file, an XML schema defines the custom markup language of the Manifest file and the appearance of a given set of XML documents. The XML schema specifies which tags or elements you can use in your documents, the attributes those tags can contain, and their arrangement.

The XML Schema file describes and dictates the content of the XML file. The CdnManifest.xsd file contains the XML schema. To view or download a copy of the CdnManifest.xsd file, see the "Viewing or Downloading XML Schema Files" section on page 6-22.

PlayServerTable XML Schema

The following XML code defines the PlayServerTable schema (playServerTable.xsd) for the CdnManfiest.xsd:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="playServerTable">
     <xs:complexType>
         <xs:sequence>
             <xs:element ref="playServer" minOccurs="1" maxOccurs="unbounded"/>
         </xs:sequence>
     </xs:complexType>
 </xs:element>
<xs:element name="playServer">
     <xs:complexType>
         <xs:choice minOccurs="1" maxOccurs="unbounded">
             <xs:element ref="contentType"/>
             <xs:element ref="extension"/>
         </xs:choice>
         <xs:attribute name="name" use="required">
             <xs:simpleType>
                 <xs:restriction base="xs:string">
                      <xs:enumeration value="wmt"/>
                      <xs:enumeration value="http"/>
                      <xs:enumeration value="qtss"/>
                 </xs:restriction>
             </xs:simpleType>
         </xs:attribute>
     </xs:complexType>
 </xs:element>
 <xs:element name="contentType">
     <xs:complexType>
         <xs:attribute name="name" type="xs:string" use="required"/>
     </xs:complexType>
 </xs:element>
 <xs:element name="extension">
     <xs:complexType>
         <xs:attribute name="name" type="xs:string" use="required"/>
     </xs:complexType>
 </xs:element>
</xs:schema>
```

Default PlayServerTable Schema

The following XML code defines the default PlayServerTable:

```
<?xml version="1.0"?>
```

```
<playServerTable xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
     xsi:noNamespaceSchemaLocation = "PlayServerTable.xsd">

   <!-- playServer http and https can always play all prepositon
      contents unless users use customized <playServerTable>
      or "playServer" attribute in the manifest file
      -->

  <playServer name="qtss">
    <contentType name="video/quicktime" />
    <extension name="mov" />
    <extension name="qt" />
    <extension name="mp4" />
    <extension name="3gp" />
    <extension name="3g2" />

    <!-- extension avi could also go here -->
  </playServer>

  <playServer name="wmt">
    <!-- MIME types taken from
    http://msdn.microsoft.com/workshop/imedia/windowsmedia/server/mime.asp
    -->
    <contentType name="video/x-ms-asf" />
    <contentType name="audio/x-ms-wma" />
    <contentType name="video/x-ms-wmv" />
    <contentType name="video/x-ms-wm"  />
    <contentType name="application/x-ms-wmz" />
    <contentType name="application/x-ms-wmd" />

    <extension name="wma" /> <!-- audio content -->
    <extension name="wmv" /> <!-- audio/video content -->
    <extension name="asf" /> <!-- audio/video content (legacy) -->
    <extension name="wm"  /> <!-- reserved for future use -->

   </playServer>

</playServerTable>
```

Manifest File Time Zone Tables

To convert to local time, you must know the time difference between Greenwich mean time (GMT) and local time for both standard time and summer time (daylight saving time). Table B-6 through Table B-21 list the time zones supported by the Manifest file. The format for writing the time zone is:

<*zonename*>:[+|-:]hh:mm per line

In this format, <*zonename*> is the name of the time zone or standard time zone abbreviation (see Table B-6) without spaces before or after the colon (":"), and "[+|-:]hh:mm" is the GMT offset in hours and minutes. The GMT offset default is "+."

Table B-6 Standard Time Zones and GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
ACT:+:09:30	Etc/GMT+7:-:07:00	HST:-:10:00
ADT:-:03:00	Etc/GMT+8:-:08:00	IET:-:05:00
AET:+:10:00	Etc/GMT+9:-:09:00	IST:+:05:30

Table B-6 Standard Time Zones and GMT Offsets (continued)

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
AGT:-:03:00	Etc/GMT-0:00:00	JST:+:09:00
ART:+:02:00	Etc/GMT-10:+:10:00	MDT:-:06:00
AST:-:09:00	Etc/GMT-11:+:11:00	MET:+:01:00
BET:-:03:00	Etc/GMT-12:+:12:00	MIT:-:11:00
BST:+:06:00	Etc/GMT-13:+:13:00	MST7MDT:-:07:00
CAT:+:02:00	Etc/GMT-14:+:14:00	MST:-:07:00
CDT:-:05:00	Etc/GMT-1:+:01:00	NET:+:04:00
CET:+:01:00	Etc/GMT-2:+:02:00	NST:+:12:00
CNT:-:03:30	Etc/GMT-3:+:03:00	NZ-CHAT:+:12:45
CST6CDT:-:06:00	Etc/GMT-4:+:04:00	NZ:+:12:00
CST:-:06:00	Etc/GMT-5:+:05:00	Navajo:-:07:00
CTT:+:08:00	Etc/GMT-6:+:06:00	PDT:-:07:00
EAT:+:03:00	Etc/GMT-7:+:07:00	PLT:+:05:00
ECT:+:01:00	Etc/GMT-8:+:08:00	PNT:-:07:00
EDT:-:04:00	Etc/GMT-9:+:09:00	PRC:+:08:00
EET:+:02:00	Etc/GMT0:00:00	PRT:-:04:00
EST5EDT:-:05:00	Etc/GMT:00:00	PST8PDT:-:08:00
EST:-:05:00	Etc/Greenwich:00:00	PST:-:08:00
Etc/GMT+0:00:00	Etc/UCT:00:00	ROK:+:09:00
Etc/GMT+10:-:10:00	Etc/UTC:00:00	SST:+:11:00
Etc/GMT+11:-:11:00	Etc/Universal:00:00	UCT:00:00
Etc/GMT+12:-:12:00	Etc/Zulu:00:00	UTC:00:00
Etc/GMT+1:-:01:00	GB-Eire:00:00	Universal:00:00
Etc/GMT+2:-:02:00	GB:00:00	VST:+:07:00
Etc/GMT+3:-:03:00	GMT0:00:00	W-SU:+:03:00
Etc/GMT+4:-:04:00	GMT:00:00	WET:00:00
Etc/GMT+5:-:05:00	Greenwich:00:00	Zulu:00:00
Etc/GMT+6:-:06:00	HDT:-:09:00	—

Table B-7 Africa GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Africa/Abidjan:00:00	Africa/Djibouti:+:03:00	Africa/Maputo:+:02:00
Africa/Accra:00:00	Africa/Douala:+:01:00	Africa/Maseru:+:02:00
Africa/Addis_Ababa:+:03:00	Africa/El_Aaiun:00:00	Africa/Mbabane:+:02:00
Africa/Algiers:+:01:00	Africa/Freetown:00:00	Africa/Mogadishu:+:03:00

Table B-7 Africa GMT Offsets (continued)

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Africa/Asmera:+:03:00	Africa/Gaborone:+:02:00	Africa/Monrovia:00:00
Africa/Bamako:00:00	Africa/Harare:+:02:00	Africa/Nairobi:+:03:00
Africa/Bangui:+:01:00	Africa/Johannesburg:+:02:00	Africa/Ndjamena:+:01:00
Africa/Banjul:00:00	Africa/Kampala:+:03:00	Africa/Niamey:+:01:00
Africa/Bissau:00:00	Africa/Khartoum:+:03:00	Africa/Nouakchott:00:00
Africa/Blantyre:+:02:00	Africa/Kigali:+:02:00	Africa/Ouagadougou:00:00
Africa/Brazzaville:+:01:00	Africa/Kinshasa:+:01:00	Africa/Porto-Novo:+:01:00
Africa/Bujumbura:+:02:00	Africa/Lagos:+:01:00	Africa/Sao_Tome:00:00
Africa/Cairo:+:02:00	Africa/Libreville:+:01:00	Africa/Timbuktu:00:00
Africa/Casablanca:00:00	Africa/Lome:00:00	Africa/Tripoli:+:02:00
Africa/Ceuta:+:01:00	Africa/Luanda:+:01:00	Africa/Tunis:+:01:00
Africa/Conakry:00:00	Africa/Lubumbashi:+:02:00	Africa/Windhoek:+:01:00
Africa/Dakar:00:00	Africa/Lusaka:+:02:00	—
Africa/Dar_es_Salaam:+:03:00	Africa/Malabo:+:01:00	—

Table B-8 America GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
America/Adak:-:10:00	America/Grenada:-:04:00	America/Noronha:-:02:00
America/Anchorage:-:09:00	America/Guadeloupe:-:04:00	America/North_Dak/Ctr:-:06:00
America/Anguilla:-:04:00	America/Guatemala:-:06:00	America/Panama:-:05:00
America/Antigua:-:04:00	America/Guayaquil:-:05:00	America/Pangnirtung:-:05:00
America/Araguaina:-:03:00	America/Guyana:-:04:00	America/Paramaribo:-:03:00
America/Aruba:-:04:00	America/Halifax:-:04:00	America/Phoenix:-:07:00
America/Asuncion:-:04:00	America/Havana:-:05:00	America/Port-au-Prince:-:05:00
America/Atka:-:10:00	America/Hermosillo:-:07:00	America/Port_of_Spain:-:04:00
America/Barbados:-:04:00	America/Ind/Indian:-:05:00	America/Porto_Acre:-:05:00
America/Belem:-:03:00	America/Ind/Knox:-:05:00	America/Porto_Velho:-:04:00
America/Belize:-:06:00	America/Ind/Marengo:-:05:00	America/Puerto_Rico:-:04:00
America/Boa_Vista:-:04:00	America/Ind/Vevay:-:05:00	America/Rainy_River:-:06:00
America/Bogota:-:05:00	America/Indianapolis:-:05:00	America/Rankin_Inlet:-:06:00
America/Bogota:-:05:00	America/Inuvik:-:07:00	America/Recife:-:03:00
America/Buenos_Aires:-:03:00	America/Iqaluit:-:05:00	America/Regina:-:06:00
America/Cambridge_Bay:-:07:0	America/Jamaica:-:05:00	America/Rio_Branco:-:05:00
America/Cancun:-:06:00	America/Jujuy:-:03:00	America/Rosario:-:03:00
America/Caracas:-:04:00	America/Juneau:-:09:00	America/Santiago:-:04:00

Table B-8 America GMT Offsets (continued)

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
America/Catamarca:-:03:00	America/Ken/Louisville:-:05:00	America/Santo_Domingo:-:04:0
America/Cayenne:-:03:00	America/Ken/Monticello:-:05:0	America/Sao_Paulo:-:03:00
America/Cayman:-:05:00	America/Knox_IN:-:05:00	America/Scoresbysund:-:01:00
America/Chicago:-:06:00	America/La_Paz:-:04:00	America/Shiprock:-:07:00
America/Chihuahua:-:07:00	America/Lima:-:05:00	America/St_Johns:-:03:30
America/Cordoba:-:03:00	America/Los_Angeles:-:08:00	America/St_Lucia:-:04:00
America/Costa_Rica:-:06:00	America/Louisville:-:05:00	America/St_Thomas:-:04:00
America/Cuiaba:-:04:00	America/Maceio:-:03:00	America/St_Vincent:-:04:00
America/Curacao:-:04:00	America/Managua:-:06:00	America/Swift_Current:-:06:00
America/Danmarkshavn:00:00	America/Manaus:-:04:00	America/Tegucigalpa:-:06:00
America/Dawson:-:08:00	America/Martinique:-:04:00	America/Thule:-:04:00
America/Dawson_Creek:-:07:00	America/Mazatlan:-:07:00	America/Thunder_Bay:-:05:00
America/Denver:-:07:00	America/Mendoza:-:03:00	America/Tijuana:-:08:00
America/Detroit:-:05:00	America/Menominee:-:06:00	America/Tortola:-:04:00
America/Dominica:-:04:00	America/Merida:-:06:00	America/Vancouver:-:08:00
America/Edmonton:-:07:00	America/Mexico_City:-:06:00	America/St_Lucia:-:04:00
America/Eirunepe:-:05:00	America/Miquelon:-:03:00	America/Virgin:-:04:00
America/El_Salvador:-:06:00	America/Monterrey:-:06:00	America/Whitehorse:-:08:00
America/Ensenada:-:08:00	America/Montevideo:-:03:00	America/Winnipeg:-:06:00
America/Fort_Wayne:-:05:00	America/Montreal:-:05:00	America/Yakutat:-:09:00
America/Fortaleza:-:03:00	America/Montserrat:-:04:00	America/Yellowknife:-:07:00
America/Glace_Bay:-:04:00	America/Nassau:-:05:00	America/Virgin:-:04:00
America/Godthab:-:03:00	America/New_York:-:05:00	America/Whitehorse:-:08:00
America/Goose_Bay:-:04:00	America/Nipigon:-:05:00	America/Winnipeg:-:06:00
America/Grand_Turk:-:05:00	America/Nome:-:09:00	America/Tortola:-:04:00

Table B-9 Antarctica/Arctic GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Antarctica/Casey:+:08:00	Antarctica/McMurdo:+:12:00	Antarctica/Vostok:+:06:00
Antarctica/Davis:+:07:00	Antarctica/Palmer:-:04:00	Arctic/Longyearbyen:+:01:00
Antarctica/DtDUrville:+:10:00	Antarctica/South_Pole:+:12:00	—
Antarctica/Mawson:+:06:00	Antarctica/Syowa:+:03:00	—

Table B-10 Asia GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Asia/Aden:+:03:00	Asia/Hong_Kong:+:08:00	Asia/Riyadh87:+:03:07
Asia/Almaty:+:06:00	Asia/Hovd:+:07:00	Asia/Riyadh88:+:03:07
Asia/Amman:+:02:00	Asia/Irkutsk:+:08:00	Asia/Riyadh89:+:03:07
Asia/Anadyr:+:12:00	Asia/Istanbul:+:02:00	Asia/Riyadh:+:03:00
Asia/Aqtau:+:04:00	Asia/Jakarta:+:07:00	Asia/Saigon:+:07:00
Asia/Aqtobe:+:05:00	Asia/Jayapura:+:09:00	Asia/Sakhalin:+:10:00
Asia/Ashgabat:+:05:00	Asia/Jerusalem:+:02:00	Asia/Samarkand:+:05:00
Asia/Ashkhabad:+:05:00	Asia/Kabul:+:04:30	Asia/Seoul:+:09:00
Asia/Baghdad:+:03:00	Asia/Kamchatka:+:12:00	Asia/Shanghai:+:08:00
Asia/Bahrain:+:03:00	Asia/Karachi:+:05:00	Asia/Singapore:+:08:00
Asia/Baku:+:04:00	Asia/Kashgar:+:08:00	Asia/Taipei:+:08:00
Asia/Bangkok:+:07:00	Asia/Katmandu:+:05:45	Asia/Tashkent:+:05:00
Asia/Beirut:+:02:00	Asia/Krasnoyarsk:+:07:00	Asia/Tbilisi:+:04:00
Asia/Bishkek:+:05:00	Asia/Kuala_Lumpur:+:08:00	Asia/Tehran:+:03:30
Asia/Brunei:+:08:00	Asia/Kuching:+:08:00	Asia/Tel_Aviv:+:02:00
Asia/Calcutta:+:05:30	Asia/Kuwait:+:03:00	Asia/Thimbu:+:06:00
Asia/Choibalsan:+:09:00	Asia/Macao:+:08:00	Asia/Thimphu:+:06:00
Asia/Chongqing:+:08:00	Asia/Magadan:+:11:00	Asia/Tokyo:+:09:00
Asia/Chungking:+:08:00	Asia/Manila:+:08:00	Asia/Ujung_Pandang:+:08:00
Asia/Colombo:+:06:00	Asia/Muscat:+:04:00	Asia/Ulaanbaatar:+:08:00
Asia/Dacca:+:06:00	Asia/Nicosia:+:02:00	Asia/Ulan_Bator:+:08:00
Asia/Damascus:+:02:00	Asia/Novosibirsk:+:06:00	Asia/Urumqi:+:08:00
Asia/Dhaka:+:06:00	Asia/Omsk:+:06:00	Asia/Vientiane:+:07:00
Asia/Dili:+:09:00	Asia/Phnom_Penh:+:07:00	Asia/Vladivostok:+:10:00
Asia/Dubai:+:04:00	Asia/Pontianak:+:07:00	Asia/Yakutsk:+:09:00
Asia/Dushanbe:+:05:00	Asia/Pyongyang:+:09:00	Asia/Yekaterinburg:+:05:00
Asia/Gaza:+:02:00	Asia/Qatar:+:03:00	Asia/Yerevan:+:04:00
Asia/Harbin:+:08:00	Asia/Rangoon:+:06:30	—

Table B-11 Atlantic GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Atlantic/Azores:-:01:00	Atlantic/Faeroe:00:00	Atlantic/South_Georgia:-:02:00
Atlantic/Bermuda:-:04:00	Atlantic/Jan_Mayen:+:01:00	Atlantic/St_Helena:00:00
Atlantic/Canary:00:00	Atlantic/Madeira:00:00	Atlantic/Stanley:-:04:00
Atlantic/Cape_Verde:-:01:00	Atlantic/Reykjavik:00:00	—

Table B-12 Australia GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Australia/ACT:+:10:00	Australia/LHI:+:10:30	Australia/Queensland:+:10:00
Australia/Adelaide:+:09:30	Australia/Lindeman:+:10:00	Australia/South:+:09:30
Australia/Brisbane:+:10:00	Australia/Lord_Howe:+:10:30	Australia/Sydney:+:10:00
Australia/Broken_Hill:+:09:30	Australia/Melbourne:+:10:00	Australia/Tasmania:+:10:00
Australia/Canberra:+:10:00	Australia/NSW:+:10:00	Australia/Victoria:+:10:00
Australia/Darwin:+:09:30	Australia/North:+:09:30	Australia/West:+:08:00
Australia/Hobart:+:10:00	Australia/Perth:+:08:00	Australia/Yancowinna:+:09:30

Table B-13 Brazil GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Brazil/Acre:-:05:00	Brazil/East:-:03:00	Brazil/West:-:04:00
Brazil/DeNoronha:-:02:00	—	—

Table B-14 Canada/Chile/Cuba GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Canada/Atlantic:-:04:00	Canada/Mountain:-:07:00	Canada/Yukon:-:08:00
Canada/Central:-:06:00	Canada/Newfoundland:-:03:30	Chile/Continental:-:04:00
Canada/East-Ssktchwan:-:06:00	Canada/Pacific:-:08:00	Chile/EasterIsland:-:06:00
Canada/Eastern:-:05:00	Canada/Saskatchewan:-:06:00	Cuba:-:05:00

Table B-15 Egypt/Eire/Europe GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Egypt:+:02:00	Europe/Kiev:+:02:00	Europe/Simferopol:+:02:00
Eire:00:00	Europe/Lisbon:00:00	Europe/Skopje:+:01:00
Europe/Amsterdam:+:01:00	Europe/Ljubljana:+:01:00	Europe/Sofia:+:02:00
Europe/Andorra:+:01:00	Europe/London:00:00	Europe/Stockholm:+:01:00
Europe/Athens:+:02:00	Europe/Luxembourg:+:01:00	Europe/Tallinn:+:02:00
Europe/Belfast:00:00	Europe/Madrid:+:01:00	Europe/Tirane:+:01:00
Europe/Belgrade:+:01:00	Europe/Malta:+:01:00	Europe/Tiraspol:+:02:00
Europe/Berlin:+:01:00	Europe/Minsk:+:02:00	Europe/Uzhgorod:+:02:00
Europe/Bratislava:+:01:00	Europe/Monaco:+:01:00	Europe/Vaduz:+:01:00
Europe/Brussels:+:01:00	Europe/Moscow:+:03:00	Europe/Vatican:+:01:00
Europe/Bucharest:+:02:00	Europe/Nicosia:+:02:00	Europe/Vienna:+:01:00
Europe/Budapest:+:01:00	Europe/Oslo:+:01:00	Europe/Vilnius:+:02:00

Table B-15 Egypt/Eire/Europe GMT Offsets (continued)

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Europe/Chisinau:+:02:00	Europe/Paris:+:01:00	Europe/Warsaw:+:01:00
Europe/Copenhagen:+:01:00	Europe/Prague:+:01:00	Europe/Zagreb:+:01:00
Europe/Dublin:00:00	Europe/Riga:+:02:00	Europe/Zaporozhye:+:02:00
Europe/Gibraltar:+:01:00	Europe/Rome:+:01:00	Europe/Zurich:+:01:00
Europe/Helsinki:+:02:00	Europe/Samara:+:04:00	Europe/Simferopol:+:02:00
Europe/Istanbul:+:02:00	Europe/San_Marino:+:01:00	Europe/Skopje:+:01:00
Europe/Kaliningrad:+:02:00	Europe/Sarajevo:+:01:00	Europe/Sofia:+:02:00

Table B-16 Hong Kong/Iceland/India/Iran/Israel GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Hongkong:+:08:00	Indian/Cocos:+:06:30	Indian/Mauritius:+:04:00
Iceland:00:00	Indian/Comoro:+:03:00	Indian/Mayotte:+:03:00
Indian/Antananarivo:+:03:00	Indian/Kerguelen:+:05:00	Indian/Reunion:+:04:00
Indian/Chagos:+:06:00	Indian/Mahe:+:04:00	Iran:+:03:30
Indian/Christmas:+:07:00	Indian/Maldives:+:05:00	Israel:+:02:00

Table B-17 Jamaica/Japan/Kwajalein/Libya GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Jamaica:-:05:00	Kwajalein:+:12:00	Libya:+:02:00
Japan:+:09:00	—	—

Table B-18 Mexico/Mideast GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Mexico/BajaNorte:-:08:00	Mexico/General:-:06:00	Mideast/Riyadh88:+:03:07
Mexico/BajaSur:-:07:00	Mideast/Riyadh87:+:03:07	Mideast/Riyadh89:+:03:07

Table B-19 Pacific/Poland/Portugal GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Pacific/Apia:-:11:00	Pacific/Johnston:-:10:00	Pacific/Ponape:+:11:00
Pacific/Auckland:+:12:00	Pacific/Kiritimati:+:14:00	Pacific/Port_Moresby:+:10:00
Pacific/Chatham:+:12:45	Pacific/Kosrae:+:11:00	Pacific/Rarotonga:-:10:00
Pacific/Easter:-:06:00	Pacific/Kwajalein:+:12:00	Pacific/Saipan:+:10:00
Pacific/Efate:+:11:00	Pacific/Majuro:+:12:00	Pacific/Samoa:-:11:00

Table B-19 Pacific/Poland/Portugal GMT Offsets (continued)

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Pacific/Enderbury:+:13:00	Pacific/Marquesas:-:09:30	Pacific/Tahiti:-:10:00
Pacific/Fakaofo:-:10:00	Pacific/Midway:-:11:00	Pacific/Tarawa:+:12:00
Pacific/Fiji:+:12:00	Pacific/Nauru:+:12:00	Pacific/Tongatapu:+:13:00
Pacific/Funafuti:+:12:00	Pacific/Niue:-:11:00	Pacific/Truk:+:10:00
Pacific/Galapagos:-:06:00	Pacific/Norfolk:+:11:30	Pacific/Wake:+:12:00
Pacific/Gambier:-:09:00	Pacific/Noumea:+:11:00	Pacific/Wallis:+:12:00
Pacific/Guadalcanal:+:11:00	Pacific/Pago_Pago:-:11:00	Pacific/Yap:+:10:00
Pacific/Guam:+:10:00	Pacific/Palau:+:09:00	Poland:+:01:00
Pacific/Honolulu:-:10:00	Pacific/Pitcairn:-:08:00	Portugal:00:00

Table B-20 Singapore/System V/Turkey GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
Singapore:+:08:00	SystemV/EST5:-:05:00	SystemV/PST8PDT:-:08:00
SystemV/AST4:-:04:00	SystemV/EST5EDT:-:05:00	SystemV/YST9:-:09:00
SystemV/AST4ADT:-:04:00	SystemV/MST7:-:07:00	SystemV/YST9YDT:-:09:00
SystemV/CST6:-:06:00	SystemV/MST7MDT:-:07:00	Turkey:+:02:00
SystemV/CST6CDT:-:06:00	SystemV/PST8:-:08:00	—

Table B-21 U.S. GMT Offsets

Time Zone: GMT Offset	Time Zone: GMT Offset	Time Zone: GMT Offset
US/Alaska:-:09:00	US/Eastern:-:05:00	US/Pacific-New:-:08:00
US/Aleutian:-:10:00	US/Hawaii:-:10:00	US/Pacific:-:08:00
US/Arizona:-:07:00	US/Indiana-Starke:-:05:00	US/Samoa:-:11:00
US/Central:-:06:00	US/Michigan:-:05:00	—
US/East-Indiana:-:05:00	US/Mountain:-:07:00	—