DataFabric® Manager

Distributing Data with DFM

Tobin Sears | 1/2004

TECHNICAL REPORT

Network Appliance, a pioneer and industry leader in data storage technology, helps organizations understand and meet complex technical challenges with advanced storage solutions and global data management strategies.

Network Appliance Inc.

Table of Contents

Document Scope

This document focuses on the distribution component of DataFabric Manager (DFM) and does not address DFM's full functionality and feature set. For information beyond the scope of the paper, please refer to the DFM product page:

http://www.netapp.com/products/filer/datafabric.html

Introduction

Device management and data management can be daunting tasks, especially in today's diverse and distributed corporate networks. The introduction of new applications and employees to support growth results in an increase in data production and network traffic—two byproducts that necessitate the need to invest in network infrastructure. Unfortunately, as new equipment and data are introduced into the network, device management and data availability emerge as paramount issues. NetApp DataFabric Manager helps organizations minimize these complexities by centralizing the management and monitoring of NetApp devices as well as providing a platform for controlling data distribution.

What Is DataFabric Manager (DFM)?

DataFabric Manager is a simple, yet powerful application for managing a distributed storage infrastructure consisting of NetApp storage and NetCache content delivery systems. DFM's support for logical group formation enables administrators to effectively manage large numbers of distributed devices through one centralized interface. Common tasks such as device configuration, software upgrades, provisioning, backup, and monitoring are simplified through device and file system consolidation. Group-based management and monitoring not only alleviate the need for a large IT staff but also provide administrators with a global view of how their network and storage infrastructure is performing on a real-time basis. DFM also helps ensure data availability and business continuance by allowing administrators to proactively predict and protect against increased demand for storage and data resources through the use of quotas, threshold settings, and the prepositioning of data.

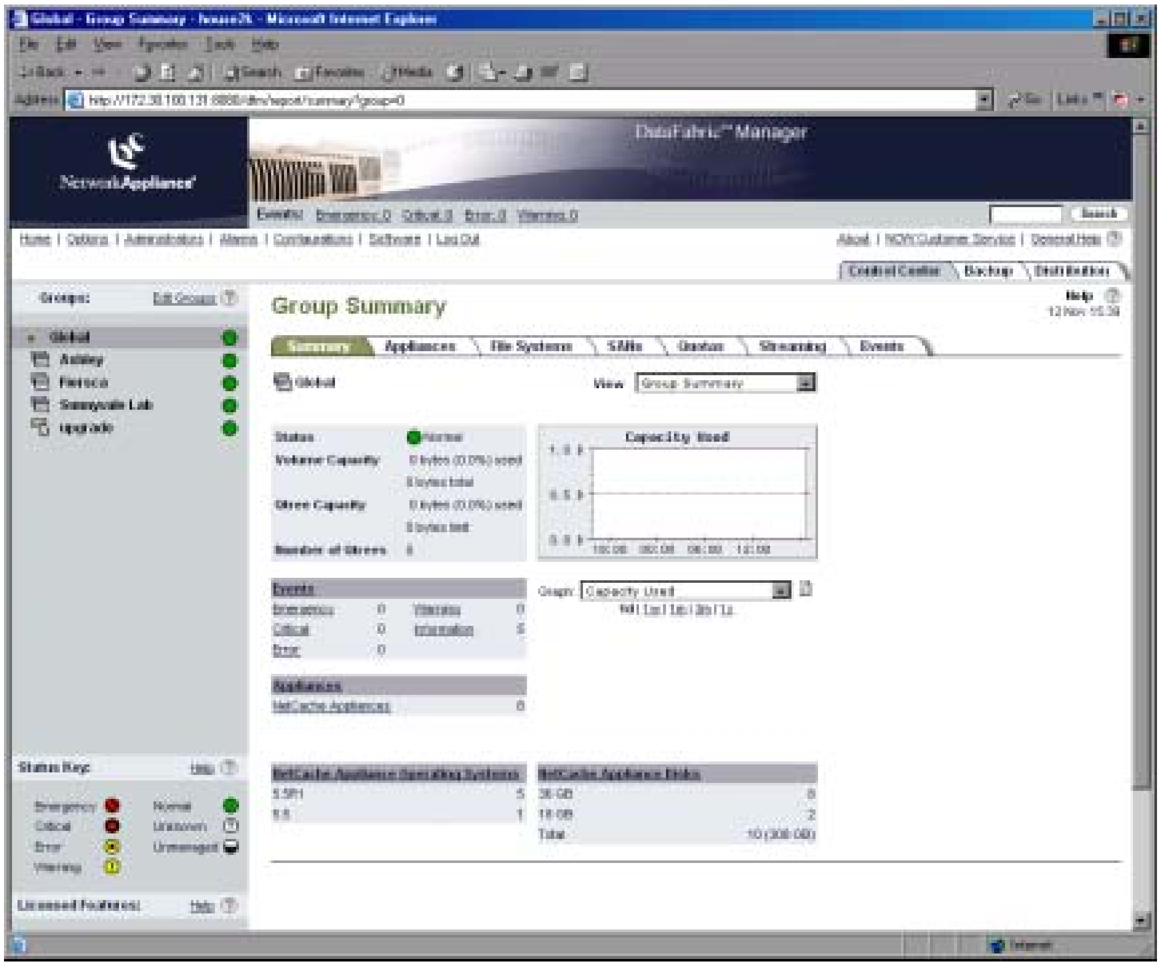


Figure 1) NetApp DataFabric Manager (DFM)

Why Is Data Prepositioning Important?

The prepositioning of data allows system administrators to effectively manage their network and server loads by guaranteeing data availability for sites served by NetCache. In order to experience any caching-related benefits, such as decreased bandwidth utilization, response times, and server load, the requested data must be available on the cache before the first client request is received—if the data isn't on the cache, requests are served directly from the origin server.

In a traditional caching environment, caches become populated with data as users make requests. This demand-based method of caching works well in most cases but can be further refined by ensuring the first request for an object is always served from the cache. Consider the following example—a remote office user accesses a multibitrate streaming file over a WAN link. If the file is encoded at 56Kb, 512Kb, and 768Kb, the client will more than likely receive the 56Kb-encoded content because of bandwidth constraints. At this point, only the 56Kb file will be on the cache, since it was the only file requested by the client. The two other files, encoded at 512Kb and 768Kb, will never make it to the cache on a passive basis. In order to provide remote users with high-quality content that they will actually watch, it needs to be prepopulated. In this case, the entire multibitrate file can be prepositioned onto the cache before a user requests it. The prepositioning process can be controlled through protocol-based bandwidth management rules and by virtue of the fact that the file isn't being streamed at a particular bitrate; instead it's transferred via HTTP or HTTPS. More information regarding supported protocols can be found in the Content Management System (CMS) Architecture section.

The Content Management System (CMS) or distribution component of DFM provides a method for controlling the prepopulation of Web-based content onto NetCache appliances. Ensuring data availability at key access points, provides added benefits to both the end user and network infrastructure.

Benefits of ensuring data availability:

- Decreased bandwidth utilization for caches located at the WAN edge
- Faster end user response times
- Offload and scale busy server farms at the data center
- Increased effectiveness of cache hierarchies caused by prepopulating parent caches

Distribution Prerequisites

Currently, CMS is able to manage the prepopulation of HTTP- and streaming (MMS and RTSP)-based objects. Before preloading can take place, however, the following prerequisites must be satisfied on both NetCache and the DFM server:

- A valid NetCache management license must be installed on the DFM server.
- A valid distribution license must be installed on the DFM server.
- The HTTP protocol must be enabled on the participating NetCache appliances. (MMS and RTSP licenses must also be installed and enabled if objects are to be stored using those protocols.)
- NetCache appliances that will participate in the distribution need to be organized into groups on the DFM server. A group consists of one of more appliances. (See the “Edit Groups” page in DFM.)
- The user creating and running the distribution has distribution rights. (See the “Administrators” page in DFM.)
- The content intended to be prepopulated onto the caches is accessible via HTTP or HTTPS.
- The content intended to be prepopulated needs to be cacheable. Both preloaded content and demand-based content adhere to the same cacheability constructs. Content that isn’t cacheable, such as cookies and URLs containing cgi-bin, will not be stored on the cache.

CMS Architecture

Concepts Used in CMS

Table of contents (TOC): An XML file created by DataFabric Manager that contains the list of objects you want to distribute to the NetCache appliances. This is also referred to as a “content list.”

Distribution: The collective data and settings required to distribute content, such as the TOC, NetCache appliance destination groups, and scheduling properties. When a distribution is running, it’s considered a job.

Job: Any action performed with the objective of distributing content, such as creating a TOC, retrieving content, or ejecting content. A job contains all of the information defined in a distribution.

Overview

The content management system is composed of two distinct components—the DFM distribution application and the NetCache distribution agent. The DFM component is used to define and manage an XML-based table of contents (TOC) file, while the distribution agent residing on NetCache is responsible for TOC interpretation and execution. The TOC defines the content to be acquired by the caches and is delivered to them by DFM. When the caches receive the TOC from DFM, content is pulled from the specified Web server as predicated by the schedule. The advantages of using a decentralized architecture can be summarized below:

- Avoids large amounts of I/O processing and disk storage.
- Allows for fan-in of content to caches from multiple applications and Web servers.
- Allows for greater flexibility when adding or deleting caches from content groups.
- Allows for easier error recovery. No retransmission of previously delivered content is needed if a cache loses connection during a content push operation—only the content that hasn't been delivered will be requested during the next push to that cache.

DFM Distribution Application

The distribution option of DFM is a separately licensed component used to define and manage TOC creation and scheduling. The DFM application is responsible for generating the TOC and delivering it to the specified caches but does not store any of the defined content or track which caches have what content.

DFM Distribution Application Features

The following distribution-related items can be managed from the DFM GUI and/or CLI (for CLI usage information, refer to the appendix):

I. Distribution configuration details
II. Job status
III. Events
IV. User rights and quotas

I. Distribution configuration details

The Distributions page of DFM lets the user manage new and existing distributions. The following configurable settings are available when editing or creating a distribution on the “Edit a Distribution” page:

- Distribution name or ID—a unique identifier for the distribution.

- Distribution list properties (note that more or less information may be required depending on what TOC generation method is chosen for a distribution; refer to the TOC Generation Methods in DFM section for more detail):

    - Method—Web CGI, single URL, directory walk, or spider.
    - Source site URL—An HTTP- or HTTPS-based source address used by NetCache to download the content.
    - Target site URL—A target URL used by NetCache to name the imported object.
    - Content availability settings—Content availability settings are optional metadata applied to all of the content defined in the distribution.

      The following settings control content availability on NetCache appliances:

        - Max age: Sets the time after which the NetCache appliance may verify with the origin server that the inserted cached content is current. The appliance may contact the origin server to resolve misses at any time.
        - Min age: Specifies the earliest time that content may be served.
        - Lock time: Specifies the period that the file will be stored in the cache as a “locked” object. If no value for lock time is set in the content list, it is set to the current time when you update the distribution in DataFabric Manager. More information on how a locked object is modified and/or ejected can be found in the NetCache Distribution Agent section.

    - TOC and content update scheduling

      There are two tasks that can be scheduled for each distribution:

        1. Create the content list (TOC): Schedule when and how often to update the content list defined in the distribution.
        2. Update the content on appliances: Schedule when and how often NetCache appliances fetch updated content from the origin Web servers.

           Schedules can be set for a monthly, weekly, daily, or hourly recurrence. Scheduling is required only as a means to automate recurring updates. If scheduling isn’t desired, manual updates can be performed from the main distribution page. Note that to enable or activate a schedule, it’s necessary to select the checkbox immediately above the schedule settings. If the checkbox is not selected, the schedule is not enabled and will not take effect. Table 1 summarizes the available scheduling options for the TOC and content.

Table 1) TOC and Content Scheduling Options

Type	Function	Use When ...
Update content on appliances	Updates the content list (TOC) and content on appliances	You have new or updated content, have updated the content list, and are ready for the appliances to retrieve the content.
Create content list	Generates or updates the content list in DataFabric Manager	You have new or updated content and want to update the content list in preparation for updating content on appliances.
Update content using current content list	Updates the content only on appliances, not the content list	You have new or updated content that can be retrieved using the current content list.
Update schedule on appliances	Updates the schedule but not content on appliances	You have changed the schedule for updating appliances and want the appliances to retrieve the new schedule but not retrieve content.
Remove schedule from appliances	Deletes the schedule, but not the content list or content, from appliances	You want to stop performing automatically recurring updates and instead perform only manual updates.
Eject content from appliances	Ejects all content associated with the distribution from appliances in the participating groups	You want to replace all the content with new content, or you want to discontinue the site.

- Participating group selection (groups that will receive the TOC defined in this distribution)
  - Group selections for the distribution

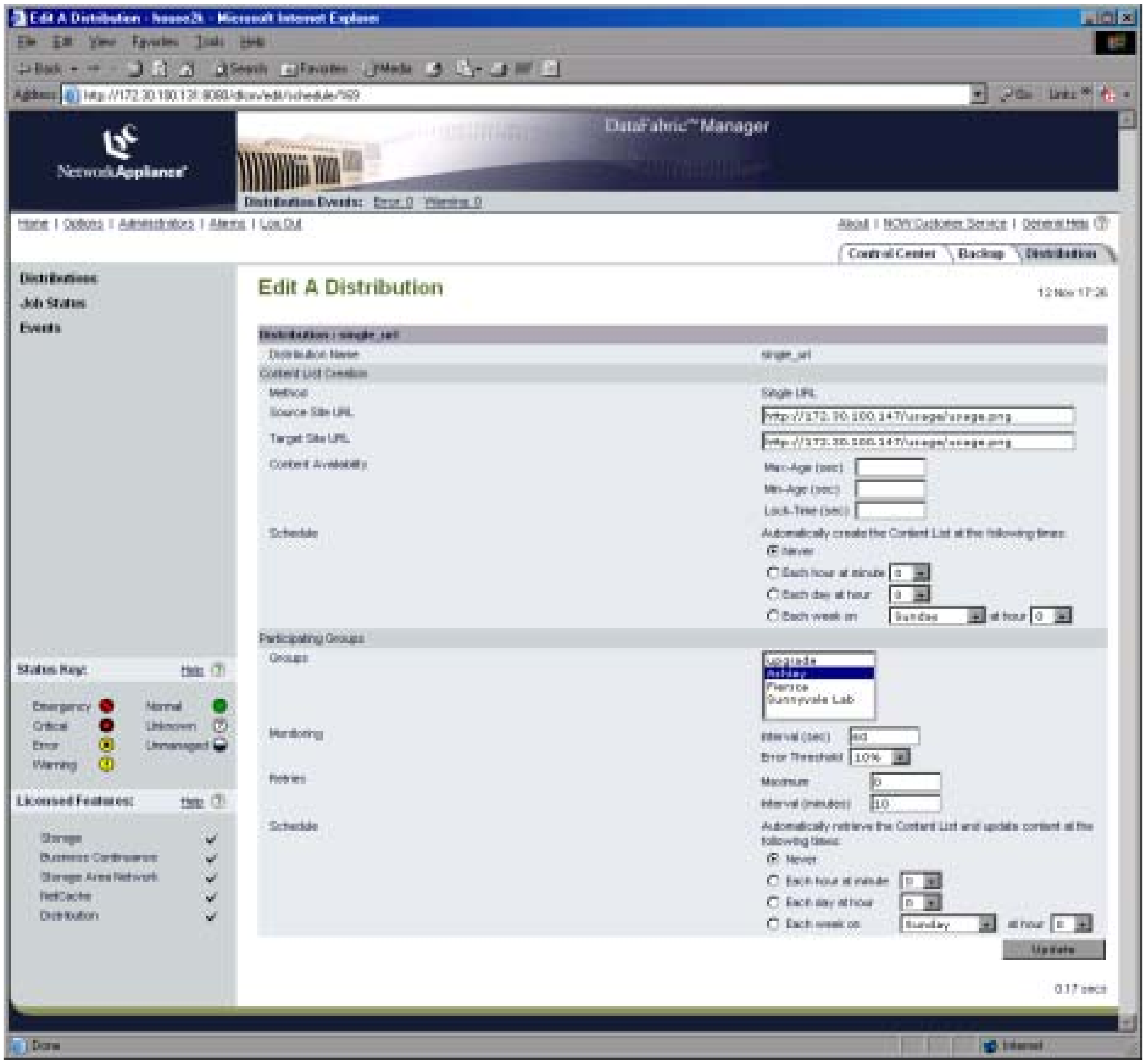
    The Groups field is the only field in this section that you must set for the distribution to take effect. You must select at least one group to act as destination for the content. If you do not select a group, DataFabric Manager generates an error when you attempt to update the distribution. Other fields in the Participating Groups section can be left at the default values.
  - Group monitoring settings

    The monitoring options allow the user to control the status reports sent to DFM while the NetCache appliances are fetching content.
  - Error threshold and retry definitions

    The retry options determine how NetCache will behave when an attempt to retrieve content fails. Note that NetCache will attempt a retry any time it fails to retrieve one or more files. It is not necessary for the failure rate to reach the error threshold specified in the monitoring options before another attempt is made. More information can be found in the Error Handling section.

Table 2) Participating Group Monitor and Retry Settings

Setting	Function	Default Value
Monitoring—interval	Determines how frequently the appliance sends status reports to DataFabric Manager.	60 sec.
Monitoring—threshold	Determines the minimum value for assigning a status of “failed” to the distribution job. This value measures the number of files listed in the content list that were not successfully fetched by the appliance, as a percentage of the total number of files in the content list.	10%
Retries—maximum	Determines the maximum number of times the appliance attempts retries to fetch content.	5
Retries—interval	Determines how frequently the appliance attempts a retry to fetch content.	30 sec.


Edit A Distribution

12 Nov 17:26

Distribution : single_url	
Distribution Name	single_url
Content List Creation	
Method	Single URL
Source Site URL	http://172.30.100.147/usage/usage.png
Target Site URL	http://172.30.100.147/usage/usage.png

Content Availability	Max-Age (sec):
	 Min-Age (sec):
	 Lock-Time (sec):

Schedule	Automatically create the Content List at the following times: [✓ Never] / Each hour at minute 0 / Each day at hour 0 / Each week on Sunday at hour 0
Participating Groups	
Groups	Upgrade, [✓ Ashley], Pierce, Sunnyvale Lab
Monitoring	Interval (sec): 60 Error Threshold: 10%
Retries	Maximum: 0 Interval (minutes): 10
Schedule	Automatically retrieve the Content List and update content at the following times: [✓ Never] / Each hour at minute 0 / Each day at hour 0 / Each week on Sunday at hour 0

0.17 secs

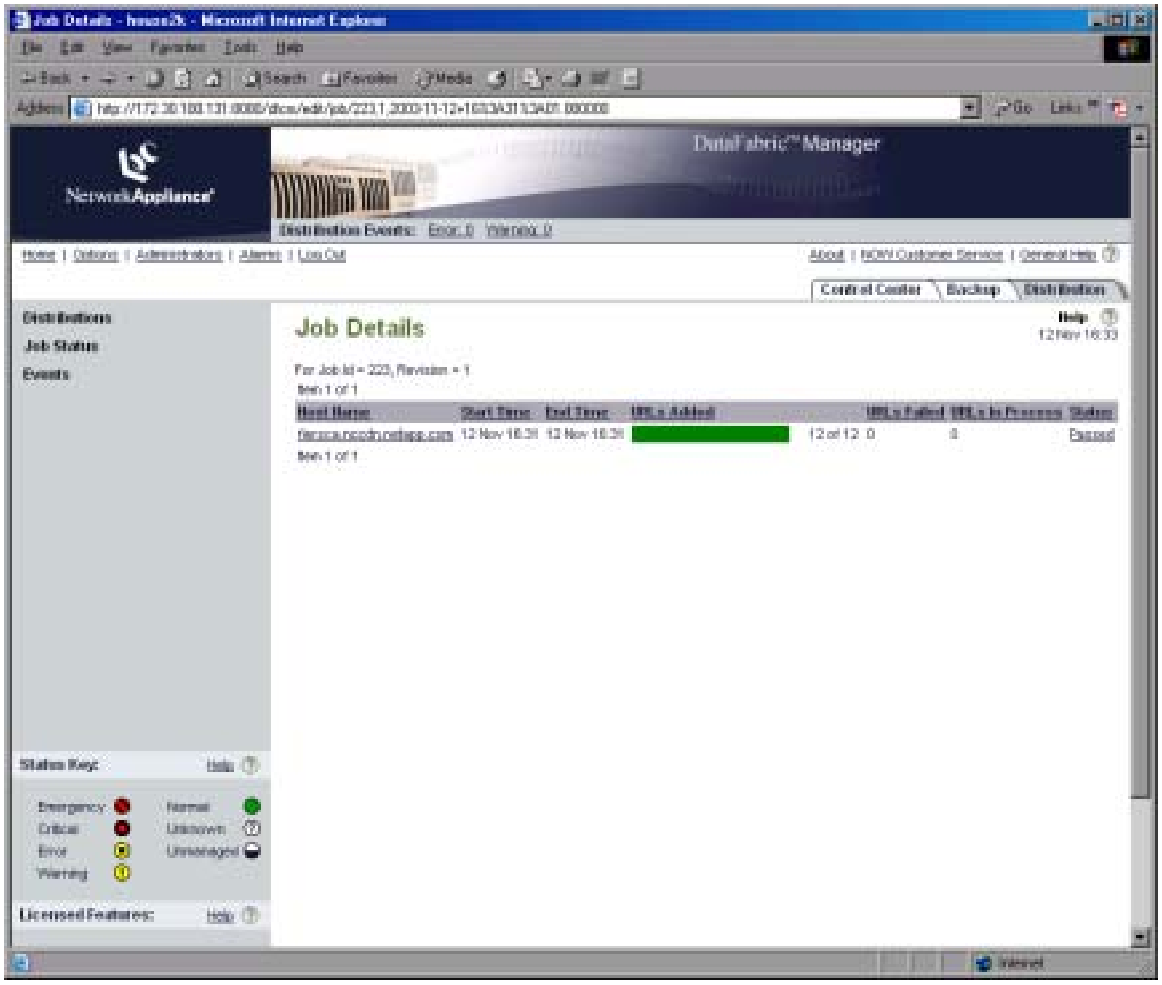
Figure 2) The “Edit a Distribution” Page

II. Job Status

The job status page displays the current status of all completed or running jobs, as well as any that may have failed to complete. A job is defined as any action performed with the objective of distributing content, such as creating a TOC, retrieving content, or ejecting content. Users can access detailed information about a specific job by clicking the job’s Job ID link (Table 3 and Figure 3).

Table 3) Information Available for a Specific Job in the Job Details Section

Data Type	Description
Host name	The name of the NetCache appliance
Host ID	The numeric identifier of the NetCache appliance
Start time	Time the job started
End time	Time the job finished
URLs added	Graphic illustration of the percentage of URLs successfully retrieved by this appliance
URLs failed	Number of URLs not successfully retrieved by the appliance
URLs in process	Number of URLs currently being retrieved by the appliance
Status	Status of the job: in process, passed, or failed


Job Details - hsuas2k - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address
Go Links
NetworkAppliance
DataFabric™ Manager
Distribution Events: Error: 0 Warning: 0
Home | Options | Administrators | Alarms | Log Out
About | NOW Customer Service | General Help
Control Center Backup Distribution
Distributions
Job Status
Events
Help
Status Key:
Emergency
Critical
Error
Warning
Normal
Unknown
Unmanaged
Licensed Features:
Internet

Job Details

12 Nov 16:33

For Job Id = 223, Revision = 1

Item 1 of 1

Host Name	Start Time	End Time	URLs Added		URLs Failed	URLs In Process	Status
[illegible]	12 Nov 16:31	12 Nov 16:31	🟩	12 of 12	0	0	Passed

Item 1 of 1

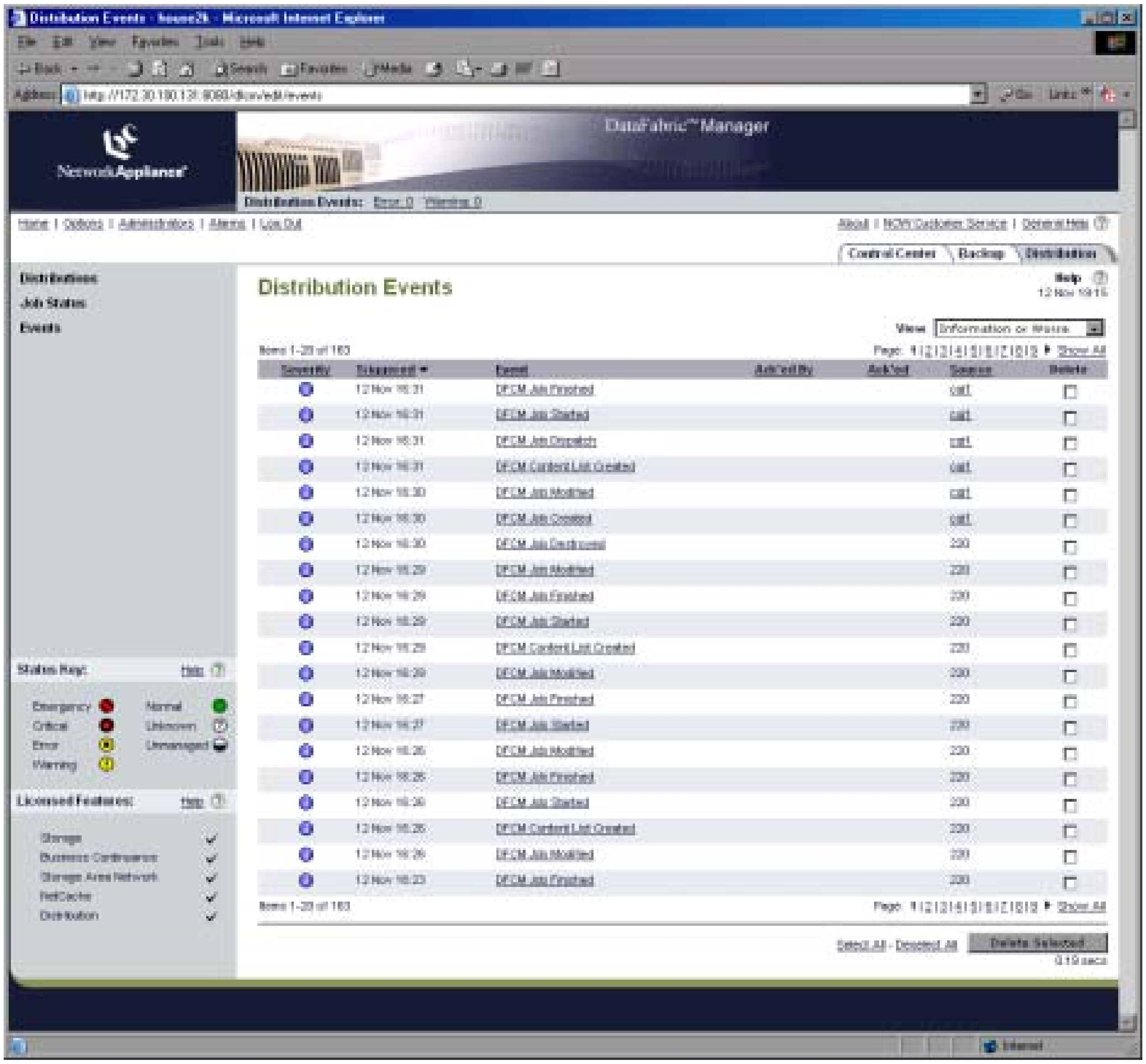
Figure 3) The Job Details page

III. Events

The events page provides a chronological listing of any distribution-related event such as a job modification or a content list creation. For administrative purposes, each individual event contains informational flags or triggers indicating the status of the logged event. Administrators can use these flags to ensure that the distribution is operating efficiently.

Table 4) Information Available in the Event Details Section

Information Type	Description
Event	Displays the name of the event—for example, NVRAM Battery: Missing, Disks: Some Failed, and so on.
Source	Displays the source of the event, either an appliance or a volume.
Severity	Displays the icon and value (for example, Error) corresponding to the severity of the event.
About	Displays the reason event occurred.
Condition	Displays the actual condition of the source because of the event—for example, Volume 98.3% full.
Triggered	Displays the date and time that this event was triggered.
Acknowledged	Displays the date and time that an administrator acknowledged the receipt of the notification of this event. If there was no need to acknowledge the event, this value is the same as the event timestamp.
Deleted	Displays whether the event has been deleted from the event log or not.



Distribution Events

Items 1-20 of 183

Severity	Triggered ▾	Event	Ack'ed By	Ack'ed	Source	Delete
●	12 Nov 16:31	DFCM Job Finished			cat1	☐
●	12 Nov 16:31	DFCM Job Started			cat1	☐
●	12 Nov 16:31	DFCM Job Dispatch			cat1	☐
●	12 Nov 16:31	DFCM Content List Created			cat1	☐
●	12 Nov 16:30	DFCM Job Modified			cat1	☐
●	12 Nov 16:30	DFCM Job Created			cat1	☐
●	12 Nov 16:30	DFCM Job Destroyed			220	☐
●	12 Nov 16:29	DFCM Job Modified			220	☐
●	12 Nov 16:29	DFCM Job Finished			220	☐
●	12 Nov 16:29	DFCM Job Started			220	☐
●	12 Nov 16:29	DFCM Content List Created			220	☐
●	12 Nov 16:29	DFCM Job Modified			220	☐
●	12 Nov 16:27	DFCM Job Finished			220	☐
●	12 Nov 16:27	DFCM Job Started			220	☐
●	12 Nov 16:26	DFCM Job Modified			220	☐
●	12 Nov 16:26	DFCM Job Finished			220	☐
●	12 Nov 16:26	DFCM Job Started			220	☐
●	12 Nov 16:26	DFCM Content List Created			220	☐
●	12 Nov 16:26	DFCM Job Modified			220	☐
●	12 Nov 16:23	DFCM Job Finished			220	☐

Items 1-20 of 183

Figure 4) The Distribution Events Page

IV. User rights and quotas

The DFM application supports global and group-based access control. Global access privileges allow the user to view and perform actions on any group within DFM. Group access privileges limit user access to specified groups only. For a user who has been granted Distribution rights, the following additional group, distribution, and quota privileges can be specified:

- Group access controls—determine what groups a user can push content to.
- Distribution access controls—determine which URLs a user can push to the caches. For each target URL in the TOC, the NetCache distribution agent evaluates the configured regex-based rules in determining whether or not the user has permission to fill the specified content.
- Quotas—Distribution disk quotas set a limit on the amount of NetCache appliance disk space each administrator can use for content distribution. An administrator can specify a disk quota for each group of NetCache appliances they have access to.

- Additional quota information
    - Quotas are configured on a per-cache basis. In other words, the configured disk quota is applied to each cache in the group. For example, if the quota is set at 100MB for a group and there are 10 NetCache appliances in the group, the administrator can use 100 MB per appliance. The total usable space for the group is 1000MB.Disk quotas are optional. An administrator without a quota for a group is allowed unlimited disk usage in that group.
    - Note that the DFM UI does not display statistics on current disk quota usage. To view the current disk quota usage for an administrator, use the CLI command “dfcm quota status.”

TOC Definition and Structure

The TOC, or content list, is used to describe the content that is being managed. Each item in the TOC is defined by four fields that tell the caches about the objects to be filled:

1. Source URL—The source URL is an HTTP- or HTTPS-based source address used by the caches to download the content.
2. Target URL—The target URL is used by the cache to name the imported objects. For HTTP- or HTTPS-based objects, this can be the same as the source URL (or left blank) if the imported object is to be stored identically to the source URL. If, however, the target URL is to be remapped to an RTSP- or MMS-based URL, a URL prefix mapping must be entered. Note that remapping streaming content pulled from an HTTP or HTTPS site to a target URL of RTSP or MMS will trigger packetization of the content on the cache as it is stored in the appropriate form.
3. Metadata—Content-associated attributes such as min/max-age and cache-lock time (maxlock-age).
4. Modification Time—A modification time associated with each object. Used to determine if it has changed since the last update.

TOC Generation Methods in DFM

DFM currently supports four methods of generating a TOC: directory walk, spider, Web cgi, and single URL. All of these methods require that the content to be filled is accessible via HTTP or HTTPS. Note that support for the acquisition of the source URL using HTTPS is supported in DFM versions 3.0 and later. Previous versions of DFM supported HTTP acquisition only.

TOC generation methods:

1. Directory walk
2. Spider
3. Web CGI
4. Single URL

---

1. Directory walk

Directory walk uses the NFS or CIFS protocol to create a list of URLs based on a recursive file system walk of a DFM-accessible, exported directory. NFS and CIFS shares that will be used to create the content’s metafile information must offer read privileges to user “nobody” in UNIX® and to the user that creates the directory walk job under Windows®. The configuration for a directory walk requires the source and target URLs as well as a directory of file mapping or the “RootPath” entry. The RootPath, which is equivalent to the absolute path in UNIX and the UNC or local path in Windows, needs to be accessible from the DFM machine. While this method quickly creates a simple mapping of static files to URLs, it cannot deal with dynamically derived data. Data generated from a database or URLs that don’t mirror the static filenames are not captured.

Directory walk example:

A user wants the cache to retrieve two Windows Media files that are located on a network-mapped drive (Z:) on the DFM machine and store them under this URL prefix:
mms://movies.mycompany.com/windows.

Z:\ = (\\10.1.1.1\share\movies\windows\)
File 1 = Z:\test1.wmv
File 2 = Z:\test2.wmv

This directory is also served from a staging Web server at http://int-stream.testserver.net/windows:8001.

Deployment note: running a Web server and a streaming server on the same machine, using nonstandard HTTP ports to serve source content:

> It's often convenient to run a Web server and streaming media server on the same machine. To avoid port conflicts between the two applications, run the HTTP server on a nonstandard HTTP port if it's enabled for HTTP-based streaming. The source URL field for all TOC generation methods supports port designations in the URL string. For added security, the HTTP directory on the origin server can be configured to only allow access to the acquiring NetCache appliance's IP address.

To create a directory walk–based distribution job, the following information needs to be entered into the distribution configuration page:

<Source URL> http://int-stream.testserver.net/windows:8001
<Target URL> mms://movies.mycompany.com/windows
<RootPath or Directory of File Mapping> Z:\

When the distribution runs, the cache will download the two files using the following URLs:

http://int-stream.testserver.net/windows/test1.wmv:8001

http://int-stream.testserver.net/windows/test2.wmv:8001

and store them as:

mms://movies.mycompany.com/windows/test1.wmv

mms://movies.mycompany.com/windows/test2.wmv

Any subsequent streaming requests for the rewritten URLs will be served as cache hits (assuming the content hasn't been modified since the content fill).

Deployment note: validating cached objects:

> After a distribution job has completed successfully, users can quickly validate the state of an individual object stored on the cache by using the "object" command from the NetCache CLI. The referenced object should be that of the target URL defined in the TOC.

```
Usage:          object <-h> <-b bitrate> <-n object_no> <-t track_no> <URL>

Example(s):     object -b 1250kbps rtsp://myserver.com/myfile
                     object ftp://user:passwd@myserver.com/path
                     object 'xover my.group 11264'
                     object 'article message_id_23423@myserver.com'
                     object 'icap ad_insertion virus_scanning http://myserver.com'
```

2. Spider

The spider method is able to generate a TOC by recursively harvesting links from a Web server. Since the crawler mimics a user request, it is able to capture content stored in databases and URLs containing query strings by following embedded URLs. DFM starts crawling from the base page, as specified in the source URL, and attempts to harvest all of the URLs. However, since spiders typically understand only a limited number of parseable content types, such as HTML, it is often difficult to generate a complete URL listing. URLs created via JavaScript, active server pages, and database-driven sites, for example, will not be parsed.

Spider example:

A user wants to populate the cache with data from http://www.somespidersite.com (three levels deep) and store the objects under the URL prefix http://web.internal-site.com/cdn.

<Source URL> or <Crawling Root URL>	http://www.somespidersite.com
<Crawling Depth>	3
<Target URL>	http://web.internal-site.com/cdn

The format for the target URLs will be composed of the target URL prefix and any subdirectories after the main source URL prefix. For example, if the crawler finds the URL http://www.somespidersite.com/pics/index.html, it will be rewritten as http://web.internal-site.com/cdn/pics/index.html.

3. Web CGI

For complex environments where static files or URLs are not readily available, a CGI script or simple URL list can be maintained on the Web server to generate the TOC. Web CGI offers the greatest level of customization but requires installation on each of the source Web servers.

- CGI example

  The following script performs a recursive scan of all directory and subdirectory contents, converting files in the Web-enabled directories to URLs. Because the script is placed in the cgi-bin directory of the Web server, the contents of the Web directory will be mapped to URLs each time the script is executed. The resulting TOC is then sent to the caches so that they can process it and download the specified content.

  Usage:

  This script creates URLs from static objects on the server from which the participating caches can download the content. The output for the script is:

  URL mod-time(Unix TimeStamp format)

  For example, items residing in the base Web directory will have this form:

  http://server/filename mod-time

  Items located in any subdirectories will have this form:

  http://server/sub/filename mod-time

  It should be noted that this will only be beneficial to the user when the cache is deployed as a forward-proxy for the hosting Web server that executes the cgi script. Since the objects are stored with the same target URL as the source URL, requests for those objects will only be served as hits if the requested URLs match the target URL. For example, if the cache was deployed as an accelerator for the source Web server and this CGI script was used, user requests in the form of http://cache/filename would be misses.

→*Begin Script*

```
#!/usr/bin/perl -w
# Return a complete listing of a web sites contents
# Usage: http://wwww.foo.com/cgi-bin/toc/pathame_to_be_listed
#
print "Content-type: text/plain";
print "\n\n";
#
# Get information about the content from the PATH_INFO variable
#
$path_info = $ENV{'PATH_INFO'};
```

```
$path_trans = $ENV{'PATH_TRANSLATED'};
$server = $ENV{'HTTP_HOST'};

&dirlist($path_trans);

sub dirlist {
        my $path = shift;
        my $FH;
        my $name = "";
        opendir($FH, $path);

        while ($name = readdir($FH)) {

                if ($name eq "." || $name eq "..") {
                        next;
                }

                my $pathname = "$path/$name";
                if (-d $pathname) {
                        &dirlist($pathname);
                } else {
                        @stuff = stat $pathname;
                        #                       #
                        # Convert pathname back into URL now
                        #
                        $pathname =~ s/$path_trans\//$path_info/;
                        print "http://$server$pathname\t$stuff[9]\n";
                }
        }
        closedir($FH);
}
```

→ *End Script*

- Simple object list example

The format for a simple object list is:

<source URL> <mod-time (Unix TimeStamp format) > <target URL> (with spaces in between)

One of the advantages to using a simple object list to generate the TOC is that multiple target URLs can be specified in the same distribution job for a given source URL. For instance, if the root URL for the object list is http://server/my_object_list.txt, this text file can contain any number of target URLs and protocols.

Sample content of “my_object_list.txt”:

<Source URL>	<mod-time>	<Target URL>
http://real.int-web.com/test1.rm	1051573150	rtsp://www.web-a.com/real/video.rm
http://real.int-web.com/test1.rm	1051573150	http://www.web-a.com/real/video.rm
http://wm.int-web.com/test1.wmv	1051573136	mms://www.web-a.com/wms/video.wmv
https://hr.int-web.com/myfile.pdf	1051573178	http://personal.web-a.com/myfile.pdf

4. Single URL

The single URL method of generating a TOC is used to map a single object. To populate the cache with the same URL as the source, leave the target URL blank or copy the same string.

<Source Site URL> http://http.someserver.net/object.pdf
<Target Site URL> blank or http://http.someserver.net/object.pdf

To populate the cache with a target URL *different* than the source URL, choose the protocol and new URL string to be stored on the cache.

Example:

<Source Site URL>	http://http.someserver.net/movies/myfile.wmv
<Target Site URL>	mms://some.streami2ngserver.com/myfile.wmv

NetCache Distribution Agent

The second component in the CMS architecture is the NetCache distribution agent. The distribution agent is the interface between DFM and NetCache and is invoked by DFM whenever a user runs a job.

The process of running a job can be divided into four steps:

Step 1: Configure caches via the XML-based API
Step 2: Pull the TOC
Step 3: Parse the TOC
Step 4: Process the TOC entries

Step 1: Configure caches via the XML-based API

The process begins with the DFM application communicating all of the configuration details to an XML-based API on the NetCache appliance. Subsequently, these settings are written and stored in the NetCache registry. Commands such as processing the TOC, deleting the TOC, aborting a process, ejecting content, and status-related queries are all supported by the API.

Steps 2 and 3: Pulling and parsing the TOC

After the cache has been configured, the TOC is pulled from the DFM server. As the TOC is being imported into the cache, the NC agent starts parsing and processing the entries. The agent does not wait for the entire TOC to be downloaded before beginning the interpretation process. In an effort to maximize efficiency, multiple entries are parsed and processed concurrently to avoid any queuing.

Step 4: Processing the TOC entries

The processing step performs one of three possible actions for each entry in the TOC: a content add, modify, or eject.

- Content additions and modifications

  The following sequence takes place after parsing each entry in the TOC:

a. The agent checks to see if the specified object is already present on the cache.
b. If the object is already on the cache, its last modified time is checked against the origin Web server to ensure freshness.
c. Objects that require updating are ejected from the cache and refetched from the origin server. The metadata that will be associated with the object will be that of the TOC rather than what is specified by the Web server.
d. If the object is still current, step 3 is skipped, and the object’s header information is updated with the metadata settings specified in the TOC.

- Ejecting content

  For each TOC, it's possible to specify a lock time that will be associated with the data to be stored on the cache. A lock time specifies the period that the file will be stored on the cache as a nonejectable object. The process of actually ejecting or deleting "locked" content is initiated by its absence from the current TOC. Objects that have an associated lock time from a previous TOC will get refreshed with the new time if they appear in the latest TOC. If an entry is absent from the most recently processed TOC, the NetCache distribution agent will perform a scan of the entire NetCache inode file and delete those objects.

Error Handling

As the TOC is processed, the NetCache agent logs any errors it encounters into an XML-based log file similar in format to the TOC. As such, the log can be processed as a retry according to the configured settings for the retry interval and maximum number of retries specified for the distribution. For each retry, only the error log is processed. The entire TOC does not need to be downloaded or parsed again for the retry to take place.

The error log contains the following information:

- Size of content list in KB
- Start and end time of job
- Total elapsed time of job
- Number of entries parsed
- Number of entries processed
- Number of entries confilled
- Number of entries failed
- Total size of job in KB
- Number of entries processed per second
- Volume of data processed per second in KB
- Average size of entries in KB
- Percentage of errors

Appendix

- Administering CMS via the DFM CLI

  The content management module of DFM can be managed from either the DFM user interface or the command line. For information on CLI usage, type “dfcm help” at the DFM server command prompt. Alternatively, use “dfcm <command> help” for specific information about a command.

  C:\>dfcm <command> help

  Available commands:

dfcm help	get help for the command
dfcm version	versions of the cms agents on the cache(s)
dfcm acl	manage the user rights for CMS
dfcm toc	TOC creation for the job
dfcm job	manage the jobs
dfcm event	acknowledge, delete, and query details of events
dfcm quota	manage the quotas

- Securing data management

  SecureAdmin™ allows administrators to secure communications between DFM and NetCache by offering support for both the SSH (Secure Shell) and SSL (Secure Sockets Layer) protocols. When SecureAdmin is configured on both DFM and NetCache, SSL will be used between the two products, including the transfer of the TOC between DFM and NetCache appliances. To enable the secure transfer of the TOC between DFM and NetCache, users must be running DFM version 3.0 and NetCache version 5.6.

Network Appliance, Inc.
495 East Java Drive
Sunnyvale, CA 94089
www.netapp.com