Akamai® NetStorage

User Guide

July 25, 2011

Akamai Technologies, Inc.

Akamai Customer Care: **1-877-425-2832** or, for routine requests, e-mail **ccare@akamai.com**

The EdgeControl® portal, for customers and resellers: **http://control.akamai.com**

US Headquarters
8 Cambridge Center
Cambridge, MA 02142

Tel: 617.444.3000
Fax: 617.444.3001

US Toll free 877.4AKAMAI (877.425.2624)

For a list of offices around the world, see:
http://www.akamai.com/en/html/about/locations.html

Akamai® NetStorage User Guide

Contents

PREFACE • 1

CHAPTER 1. INTRODUCING AKAMAI NETSTORAGE • 3

CHAPTER 2. GUIDELINES AND BEST PRACTICES • 9

CHAPTER 3. ACCESSING NETSTORAGE • 13

CHAPTER 4. USING FILE TRANSFER PROTOCOL • 17

CHAPTER 5. USING RSYNC • 21

CHAPTER 6. USING SECURE COPY • 29

CHAPTER 7. USING SSH FILE TRANSFER PROTOCOL • 33

Preface

Welcome to the *Akamai® NetStorage User Guide*. NetStorage is a managed product that provides persistent and replicated storage, and serves stored content such as images, streaming media files, software, documents, and other objects. NetStorage makes the content easily accessible to Akamai Edge servers, and thus complements Akamai's content delivery products.

About This Document

This document discusses file uploading and management on NetStorage over secure and non-secure connections. It contains a brief overview and usage guidelines and best practices. This document *does not discuss*:

- Setup and administration of your NetStorage accounts, a topic covered in the document *Managing Akamai NetStorage Accounts*, which is available from Akamai's EdgeControl® portal (https://control.akamai.com).
- Details of integration with companion Akamai products; these topics are covered in the documentation for the specific companion product in question, which is also available from the EdgeControl portal.

Chapter 1. Introducing Akamai NetStorage

In This Chapter

About NetStorage

Akamai NetStorage is a managed product that provides persistent, replicated storage of Web site content, including images, streaming media files, software, documents, and other digital objects. By mirroring content to a small number of core network locations and thus making it highly available to, and easily accessible by, Akamai Edge servers, NetStorage complements Akamai's content delivery products. NetStorage is built by layering Akamai's proprietary replication technology and Global Traffic Management on top of best-of-breed core storage. The result is a scalable, high-performance, and highly available content storage product.

Storage Site Architecture

NetStorage is built on an infrastructure of geographically diverse storage sites, as well as internally-managed Akamai storage facilities.

In addition to multiple terabytes of storage capacity, each storage site contains:

- Akamai front-end storage servers that provide access for content upload and deletion, as well as access for content retrieval by product offerings such as Akamai HTTP Content Delivery and Akamai Streaming.
- Akamai replication servers that copy files to multiple storage locations.
- Global Traffic Management network agents that gather real-time data on Internet traffic conditions, directing end-user requests for NetStorage content to the optimal storage site.
- File servers that export filesystems to the front-end storage servers.

- Dual Network Transit Provider connectivity, ensuring that NetStorage is resilient to Internet connectivity failures.

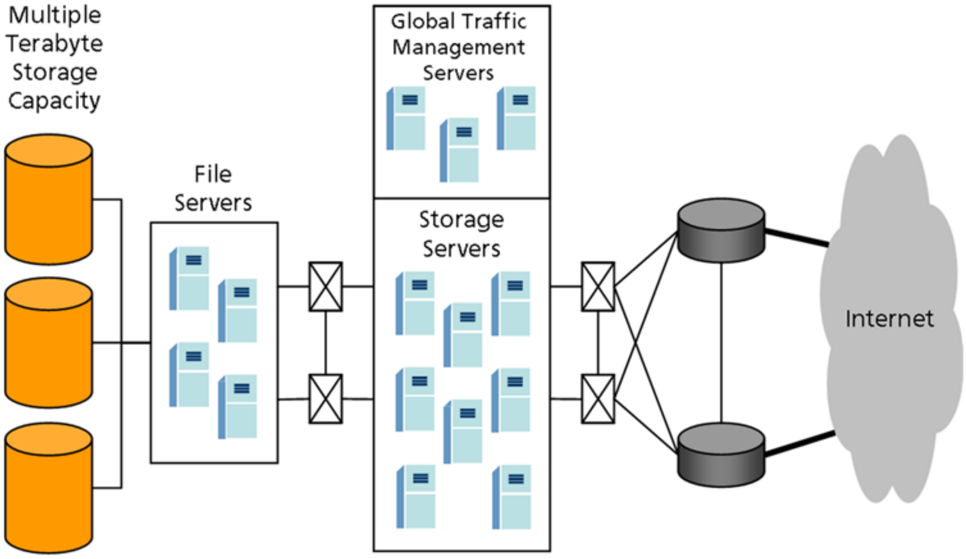
Multiple Terabyte Storage Capacity
File Servers
Global Traffic Management Servers
Storage Servers
Internet

Figure 1-1. NetStorage Site Architecture

How NetStorage Works

The very first step in setting up NetStorage is to use the EdgeControl portal to define your user accounts and passwords. Also, you will upload your SSH keys here if you intend to use one of the secure access methods. Details for performing all aspects of NetStorage configuration can be found in *Managing Akamai NetStorage Accounts*.

Uploading Content

After configuring your NetStorage account, you then upload your content to your storage site using one of the following access options:

- **File Transfer Protocol (FTP).** Akamai NetStorage supports the FTP protocol and command set as outlined in RFC 959, with some logical omissions and proprietary augmentations.
- **rsync.** NetStorage supports most of the normal command set for rsync and rsync/SSH with some exceptions regarding unsupported features such as in-place file modification, hard links, and scratch directory creation.
- **HTTP.** NetStorage supports some HTTP-based features for depositing and managing content. Please contact your Akamai representative for more information.
- **Secure Shell (SSH)-Based Access.**
    - **Secure Copy Program (SCP).** NetStorage supports the normal command set for SCP.

- **SSH File Transfer Protocol (SFTP).** NetStorage supports SFTP access using several clients, including OpenSSH, PuTTY, and WinSCP.
- **Akamai NetStorage Content Management Server (CMS).** CMS is a proprietary shell-like interface that allows content providers to manage NetStorage content in a secure environment.
- **Akamai Site Snapshot Tool (SST).** SST is a proprietary tool primarily used as a flexible failover solution to automate the process of downloading content from an enterprise onto the Akamai platform by pulling static files from a content provider's Web site origin to a failover site.

You are given access to NetStorage via a unique domain name of the form **[customer_name].upload.akamai.com**. The domain name is resolved, via Akamai's Global Traffic Management system, to a live upload server on a NetStorage site that is optimally located (in terms of load, network traffic conditions, and latency) to receive the content. The NetStorage upload server presents the client with one or more top-level directories into which content is deposited. Each directory is customer-specific, and you can be nearly as creative with structures and views within each as you can on a local server.

Although NetStorage content is ultimately stored on two sites, you need only upload content to one. After each file upload (or deletion) operation, the NetStorage replication system automatically copies the file (or propagates the deletion) to the second storage site.

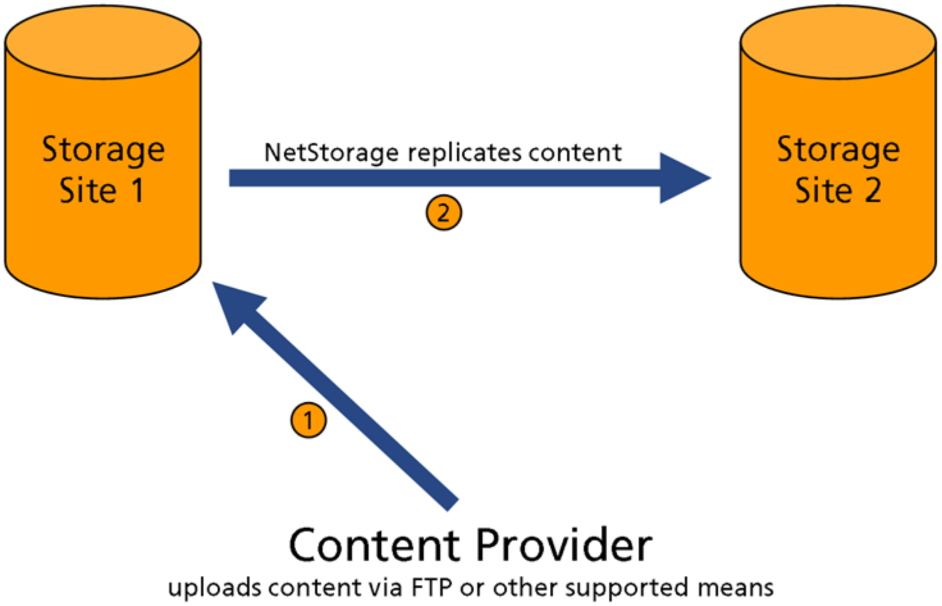
Storage Site 1
NetStorage replicates content
2
Storage Site 2
1
Content Provider
uploads content via FTP or other supported means

Figure 1-2. Upload and Replication

Downloading Content

Storage URLs

NetStorage is designed to store content that is delivered to end users by one of Akamai's content delivery products. This means that end users obtain NetStorage content through Akamai Edge servers, and in the case of a cache miss[1], the Edge servers retrieve the content from NetStorage. The URLs (Uniform Resource Locators) that Edge servers use to retrieve content from NetStorage are called Storage URLs and use the following format:

[*customer_name*]**.download.akamai.com/**[*akamai_directory*]**/**[*path*]**/**[*filename*]

Where:

- [*customer_name*]**.download.akamai.com** is resolved, via Akamai's Global Traffic Management system, to a live server in the NetStorage site that is optimally located (in terms of load, network traffic conditions, and latency) to deliver the content.
- [*akamai_directory*] identifies the customer-specific top-level directory that contains the requested content. For customers with small amounts of content (less than 100GB), [*akamai_directory*] is simply the customer's CP code; for customers with large amounts of content, it takes the form **cpcode/a**, **cpcode/b**, **cpcode/c**, and so on.
- [*path*]**/**[*filename*] represents the directory and file created by the customer via upload (for example, **ads/quicktime/car.mov**).

An example Storage URL is:

example.download.akamai.com/9391/b/ads/quicktime/car.mov

Akamaization and ARLs

To publish NetStorage content to a Web site and enable its delivery via an Akamai content delivery product, you must generate an Akamaized® URL from your Storage URL. This results in an Akamai Resource Locator (ARL)[2], which you then insert into the appropriate HTML files or content publishing system on your Web site.

For example, Akamaizing the sample Storage URL:

example.download.akamai.com/9391/b/ads/quicktime/car.mov

produces the ARL:

http://a9.g.akamai.net/f/9/9391/6h/example.download.akamai.com/9391/b/ads/quicktime/car.mov

The ARL contains a number of fields that aid in Akamai's content delivery process.

[1] Cache hit and cache miss refer to whether or not Edge servers hold the requested content in their cache.

[2] For more details on Akamaization and ARL format, please see the FreeFlow® white paper or the *Akamai On-Demand Streaming Integration Guide*.

Content Delivery to a Browser

After you upload your content to NetStorage and properly Akamaize your Web site, end-user browsers request NetStorage content from Akamai Edge servers, which is then delivered to each browser as shown in Figure 1-3

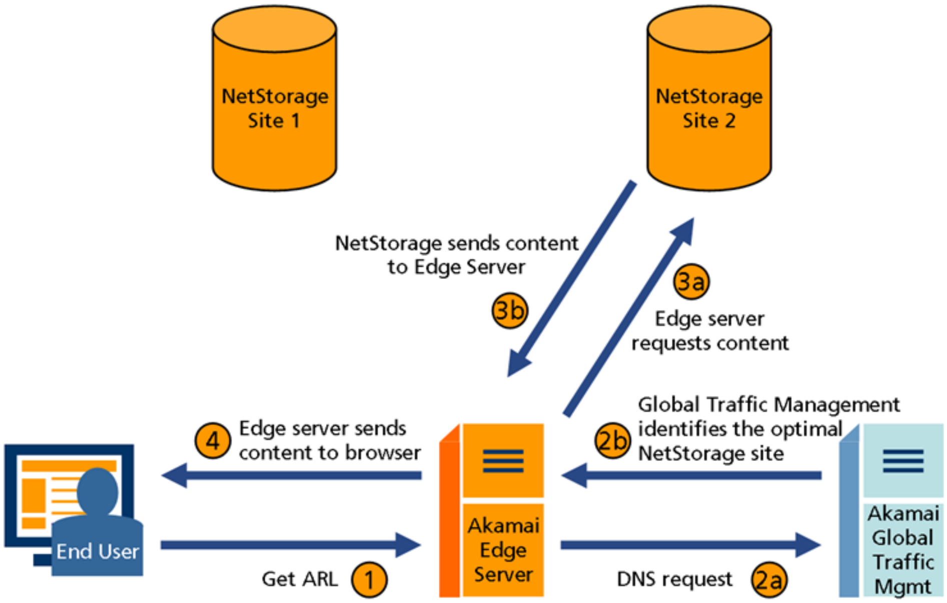
NetStorage Site 1
NetStorage Site 2
NetStorage sends content to Edge Server
3b
3a
Edge server requests content
Global Traffic Management identifies the optimal NetStorage site
2b
4
Edge server sends content to browser
End User
Akamai Edge Server
Akamai Global Traffic Mgmt
Get ARL
1
DNS request
2a

Figure 1-3. Downloading NetStorage Content

1. An end user requests content from an Akamaized Web site, and Akamai's intelligent DNS system directs the browser to the optimal Akamai Edge server. In the event of a cache hit, the Edge server simply returns the content to the browser. In the event of a cache miss, the Edge server must retrieve the content from NetStorage.
2. Global Traffic Management resolves the Storage URL domain name, **`example.download.akamai.com`**, to the optimal NetStorage site based on real-time Internet traffic mapping.
3. The Edge server retrieves the content from the optimal NetStorage site.
4. The Edge server delivers the content to the browser and stores it in cache for future requests.

Security

Access to NetStorage content is limited to customer upload accounts and Akamai Edge servers. In addition, you have the option, using the EdgeControl portal, of designating content as "Streaming Only," which further restricts access by allowing only Akamai streaming Edge servers to retrieve content. This policy prevents end users from downloading an entire streaming file that is meant to be available only via a streaming protocol. NetStorage enforces this policy by accepting or rejecting Edge server requests based on the content being requested and the type of Edge server making the request (streaming or non-streaming).

Reporting

NetStorage provides reporting of operations and usage. This is provided via the reports application available in the EdgeControl portal.

Chapter 2. Guidelines and Best Practices

In This Chapter

During the time that Akamai has provided NetStorage services to its customers, a number of best practices have been discovered. While these do not necessarily reflect the system's absolute maximums, they have been shown to provide the best overall performance.

Before Using NetStorage

Be aware of the storage group provisioning attributes set in the EdgeControl portal

The administrative interface is used to set up your storage groups, replication sets, directory access, and read-write privileges. Other attributes can include:

- The default index file name or names, such as index.htm, index.html, etc.: the file served if the request does not specify a file (i.e., if it ends with a slash "/").
- Whether or not case is forced.

  If case is forced, content names are converted to either upper or lower case as they are uploaded, depending on the option selected. From that point forward, requests for the objects coming through the Akamai network are also forced to the same case, so that there is no case mismatch.

  Be aware, this option should be set *before* uploading files to NetStorage. Files uploaded before the option is set are *not* converted, but requests coming in *after* the option is set *are* converted. If you change the option after files are uploaded, user requests may get errors based on case mismatch.

See *Managing Akamai NetStorage Accounts* for further information.

Akamai's content delivery products and NetStorage can be configured for permanent redirects

When a URL **http://hostname/directory**—with no trailing slash—is presented, the normal behavior of Web servers is to serve a permanent redirect to **http://hostname/directory/**—with a trailing slash—so the client browser correctly handles relative links in the default page (e.g., index.html). Akamai's content delivery products are unable to determine whether the final component in the URL is a directory or an object, however, because they do not receive the URL passed, but rather see a translated version. Under normal circumstances, therefore, proper redirects will not be served.

By making adjustments to your back-end configuration, Akamai can cause Akamai's content delivery products and NetStorage to compensate for this to produce the expected redirect behavior.

Some content management operations, such as renaming directories, are not allowed in NetStorage

Because of replication and certain other features involved in NetStorage, you cannot rename directories. You can, however, create symbolic links between directories as a replacement for renaming them.

Also, most file attributes, such as read/write/ownership/execute—attributes other than the time stamp—are not set using your upload account. Certain attributes (read-only, for example) are controlled through the EdgeControl portal.

NetStorage does not support in-place updates to existing files, including resumption of interrupted file uploads

While some of the client software used with several of the access methods described in this document allow the resumption of interrupted file uploads, NetStorage itself does not support it to eliminate the possibility of inadvertently replicating partially-uploaded content and making it accessible to end users.

NetStorage does not support hard link creation

Symbolic links, however, are fully supported.

User-specified scratch directories are not supported

Akamai retains control of all scratch directory parameters and behavior.

Multiple simultaneous logins can affect replication performance

Replication performance can suffer if simultaneous uploads into the same account are performed in different storage centers.

Be aware, the number of simultaneous logins by a given user account name and password is limited to 10. For more information, please contact your Akamai representative.

The Edge Logging feature has restrictions

POSTs larger than 40 kB in size cannot be written to NetStorage. Also, the expectation for frequency is about one POST per second.

Routine Usage

Directory structure impacts performance

For best performance, the ideal directory structure should be approximately two to three levels deep with each directory containing no more than 2000 files. Having too many files in the same directory may cause slowness or a timeout when trying to list the content.

Consider increasing the time-out setting

If you are trying to access NetStorage and have a very large number of files under certain directories, your client will likely time out. You may want to increase the time-out setting on your client or local firewall.

Note: Using secure protocols to transfer data to NetStorage could affect performance.

File Manager's capabilities are limited

File Manager provides a basic GUI for browsing and managing content stored in NetStorage via the portal. It is not, however, meant to be a substitute for a production file upload system with which you can traverse through complex directory trees and initiate large data transfers or upload large files.

Use rsync with caution, especially if you have a very large number of files on NetStorage

Since rsync tries to synchronize files between local and remote destinations and would peruse your entire directory structure, using it is not efficient if you are trying to sync up a smaller number of files. Here, it would be better to upload them via your CMS system.

Be aware of NetStorage's restrictions and limitations

- The maximum size of a single storage group is 10 TB.
- The maximum size of a file that can be uploaded to NetStorage is 25 GB.
- The maximum number of files or subdirectories in any given directory is 50,000.
- There are no guarantees regarding the number of operations per second that may be provided.

Replication may result in a lag in availability.

There is most often a small lag time between the completion of a file upload to one NetStorage location and its availability from the other locations. As long as the file is available at one NetStorage location, however, all Akamai Edge servers have access to it.

Be aware of NetStorage file path restrictions

- No path components (e.g., domains, directories, and filenames) may begin with a period (.).
- All path characters must be of ASCII values 32–255, inclusive.
- The total path can be no more than 800 characters in length.
- Individual path components can have no more than 200 characters.

Security

For authentication and security purposes, Akamai recommends using secure access to NetStorage, which also includes protecting your content from being infected by certain types of malwares (the Gumblar virus, for example).

Use secure access (SSH) when connecting to NetStorage

Akamai recommends using secure access (SSH) to connect to NetStorage. See "Using Secure Access" on page 13 for procedures to configure secure access. You can also refer to Knowledge Base articles 4701 and 1313.

Note: You must use the username "sshacs” when connecting to NetStorage securely. Also, once an account has been converted to secure, you cannot use that account to connect non-securely.

You can use the following methods to securely connect to NetStorage:

- **SFTP**—Secure File Transfer Protocol
- **RSYNC**—Remote Synchronization (if using SSH)
- **SSH**—Secure Shell
- **SCP**—Secure Copy

Restrict IP access to your NetStorage account

Procedures to restrict access (Account Upload IP ACLs) for NetStorage can be found in the document *Managing Akamai NetStorage Accounts* available on the EdgeControl portal.

Set up inbound firewall rules to allow access from NetStorage servers.

To do so, use the following procedures:

1. Log in to the Akamai EdgeControl portal.
2. In the left-hand navigation menu under All Services, click Firewall Rules Notifications.
3. Click Subscribe.
4. Select the **NetStorage** check box.
5. Click .

Provide restricted access to NetStorage users as necessary

For example, you may choose to give read-only access or give access to certain CP code folders only.

Make security recommendations to your NetStorage users

To help prevent your content from being infected by malware or FTP botnets, you can recommend your NetStorage users not store their FTP password on their PCs nor upload content to NetStorage from the same computer they would use to surf the Web.

Note: If you suspect your content on NetStorage is infected please refer to KB article 4611 for next steps. Please refer also to the Gumblar virus article 2722.

Chapter 3. Accessing NetStorage

In This Chapter

When uploading to and managing Akamai NetStorage, you have both secure and non-secure options available to you.

Using Non-Secure Access

For non-secure access via FTP or rsync, use the upload account name and password you configured in the EdgeControl portal. The type of access (Read/Write, directory restrictions, etc.), are also set up using that interface.

Using Secure Access

To access NetStorage through secure means, you must first send the public key for a security certificate to Akamai through the EdgeControl portal (see the document *Managing Akamai NetStorage User Accounts* for details). Each account that will be used for secure access must have one or more SSH keys uploaded; a single key cannot be assigned to more than one user name.

Once an SSH key is associated with an upload account, that account cannot be used for non-secure NetStorage access. If you need to use a combination of secure and non-secure access, you must create separate upload accounts for each.

SSH Keys

NetStorage supports SSH protocol 2 keys of types **RSA** and **DSA**. It does *not* support SSH protocol 1 keys.

When generating SSH keys, you should use ssh-keygen or some other tool that produces keys in the OpenSSH format as in the following examples:

- SSH-2 RSA

```
ssh-rsa
      AAAAB3NzaC1yc2EAAAABJQAAAIEAvLmfw5u9QTr5eBEiTiUPqkNsXGrAfV4yiNSOY
      AXx4xvfuq7rRW0w4J4bkXkb7zK6k1BJqOPPKhDO5Xz6/P7j5fz/
      BGCKrCGDibNM+qKlBUqjbnrwK5csS2cvMla29vnPopq+xYpm1Mt8K5f0npODnrVFy
      Q8ErFH8E0XNyL4Ssmk=
```

- SSH-2 DSA

```
ssh-dss AAAAB3NzaC1kc3MAAACBANnOss/pKL+W5JY1mL1kXslVIdTbFdQN/
      cW90nHRDYeO2UOGSFKYkadean79B5COyk0zO1WrEgFVb62nZuDvthe3NFmcLnemnh
      u4+2b5k7KBPMH+4WYap9287VKAqw0Qelew0gZSfH66MhZpyvRb5XRO3GOWVMVqaMc
      /y/
      9UzPPHAAAAFQCU2RjQvaxo9H5U2fYZMsRpn3jGMQAAAIEAhAWu8zyXObfnCAiEg0T
      FsdZNWYScvqybpvw4WmBnqrJj0CC3eaX8yDYK9SzZnXQRD0ALaV43p57WV2gYXGrX
      1pcw2qwQ1xLfHJDlaJKplau+tXxCMp630ZuYWDNv0QSPvNKt3uNzAoXc7odWMiVUp
      XdfmG54IMKryQmh6LFr1KkAAACAELkvOSBCtPoTA77lyywMwJmZ3oQ5moo9rCEAhr
      o4kzq/
      2gzJpVn2nZMmFQ8h3x+1DYMVGyNByU3S1j2VAvjR+BoxgeR3B1Tg+1+q+zL5k+pLH
      3F2M1ZGMkYlWhmiifmv1cTo5qzsddgbOfnC/i6r37zpLEK0xIIJmX0L1f0ZOZ8=
      jdoe-20060414-example
```

Note: NetStorage does not allow SSH public keys longer than 1000 characters in length. SSH-2 RSA keys must therefore be 2048 bits or less, and SSH-2 DSA keys must be 1024 bits or less.

Be aware, NetStorage does not support other SSH public key formats, such as the ssh.com format shown here:

```
---- BEGIN SSH2 PUBLIC KEY ----
Comment: "jdoe-20060414-example"
AAAAB3NzaC1yc2EAAAABJQAAAIEA6KT2NT7BbhPKfbeU1MgeGEfOuDbxMls7vp7z
to7Ub18CuUQB7mAeEvs3xcXyz38omT6eOubdX8kKBA6nGjsPjSPKNc7OuaTOOYh2
xoXJbA2r6ktcdi7E7PEY+nOlUgCUWxOYHQYj/204WHlXaZL+qQ1SPDhZgTv5n+bM
lQAqUjc=
---- END SSH2 PUBLIC KEY ----
```

If your tool generates the ssh.com format, you must convert it to OpenSSH format using the OpenSSH suite of tools, which is freely available for most platforms based on The Open Group's UNIX® system (a port for the Microsoft® Windows® operating system is also freely available from Cygwin). The following command performs the public key conversion:

```
ssh-keygen -f [ssh.com_format_key].pub -i > [new_openssh_file].pub
```

All other non-OpenSSH formats, including that generated by ssh-keygen2, are also unsupported.

SSH Clients

NetStorage supports the use of OpenSSH clients and PuTTY. To use protocol 2 keys with an OpenSSH client or PuTTY, simply specify the key type using the SSH-2 option.

NetStorage does not support SSH Communications Security Corporation's SSH2 client.

SSH Tips

If you have trouble using secure access with NetStorage, consider the following:

- Verify the key you uploaded to NetStorage is your public key, not your private key.
- Ensure your public key lengths are 2048 bits or less for SSH-2 RSA and 1024 bits or less for SSH-2 DSA.

Logging in to NetStorage with Secure Access

Regardless of the type of secure access you use, you must always use the user name **sshacs**, *not* the user name associated with the upload account. The SSH key uniquely assigned to the account is used to associate the login session with the directory view and attributes set for the correct NetStorage user name.

Using SSH (Secure Shell) Tunnels

Accessing SSH tunnels is not the preferred way of using SSH with NetStorage and is not permitted. You must use one of the secure-access methods discussed in this document's later chapters.

Upload and Download CNAMEs

When you access your storage site, always use your upload account hostname unless your Akamai representative specifically advises you to do otherwise.

- The upload hostname is in the form, [**customer_name**]**.upload.akamai.com**, where [**customer_name**] is the name used for your account. When you access this host name, you are automatically routed to the nearest NetStorage site.
- The download hostname, [**customer_name**]**.download.akamai.com**, is used to serve content to Akamai Edge servers. The servers are directed to download content from this hostname using either an ARL (Akamai Resource Locator) or an Edge server configuration. Unless your Akamai representative instructs you to do so, you should never use the download hostname for file upload or content management.

*Note: Do not use IPs or region-specific DNS names—names that end in **akadns.net**—unless your Akamai representative specifically instructs you to do so. Using these can result in significant error conditions.*

Your directory access is set in the EdgeControl portal. The default is often named **/nnnnn** (or is a subdirectory of **/nnnnn**), where **nnnnn** is a Content Provider (CP) code assigned to you by contract.

Chapter 4. Using File Transfer Protocol

In This Chapter

File Transfer Protocol (FTP) is a non-secure means of transferring files between networked computers using TCP (Transmission Control Protocol). Akamai NetStorage supports the FTP command set as set out in RFC 959, which you can review at **http://www.w3.org/Protocols/rfc959/Overview.html**. You can also read about FTP commands at **http://cr.yp.to/ftp.html**.

FTP Commands

Following is the complete list of RFC 959 FTP commands with their definitions. Be aware, your FTP client may list the command under a different name or not at all.

Unsupported Commands

There are some FTP tasks that do not make sense for, and are not used in, the NetStorage environment:

Command	Description
Access Control Commands	
`acct`	**Account**—Specify a user account
`rein`	**Reinitialize**—Close the user, but leave the connection open
`smnt`	**Structure Mount**—Mount a new file system data structure without changing the user or account
Transfer Parameter Commands	

`mode`	**Transfer Mode**—Designate the mode for data transfer
	S = Stream
	B = Block
	C = Compressed

`stru`	**File Structure**—Designate the file structure
	F = File (no record structure)
	R = Record structure
	P = Page structure

FTP Service Commands	
`allo`	**Allocate**—Hold space to store a file
`appe`	**Append (with create)**—Store a file on the server, appending the data to the end of an existing file, or creating a new file if one does not exist

Supported Commands

NetStorage fully supports the following FTP commands.

Command	Description
Access Control Commands	
`cdup`	**Change to Parent Directory**—Change to current directory's parent directory
`cwd`	**Change Working Directory**—Change to a different remote directory
`pass`	**Password**—Specify a user password (the **`USER`** command must be used first)
`quit`	**Logout**—End the session
`user`	**User Name**—Specify the user
Transfer Parameter Commands	
`pasv`	**Passive**—Enter or exit passive mode
`port`	**Data Port**—Designate the data connection port

`type`	**Representation Type**—Specify the type of transfer
	A = ASCII (N = Non-print, T = Telnet format effectors, C = Carriage Control (ASA))
	E=EBCDIC (N = Non-print, T = Telnet format effectors, C = Carriage Control (ASA))
	I=Image
	L <byte size> = Local byte Byte size

FTP Service Commands	
`abor`	**Abort**—Cancel the preceding service command
`dele`	**Delete**—Delete the designated file
`help`	**Help**—Display help information
`list`	**List**—Display the contents of a directory or information about a file
`mkd`	**Make Directory**—Create a new directory
`nlst`	**Name List**—Display the contents of a directory or send them to a local file
`noop`	**Noop**—Ask the server to reply with an **`OK`** message
`pwd`	**Print Working Directory**—Display the path of the current directory
`rmd`	**Remove Directory**—Delete an existing directory
`rest`	**Restart**—Restart copying the file at the specified point (must be followed by an FTP service command **for download only**; NetStorage does not yet support this command for upload) *Note: You can restart a transfer (`rest`) only before a file retrieval (`retr`). If you restart a transfer prior to a file store (`stor`), the store will return an error.*
`retr`	**Retrieve**—Copy a file from the server to the designated location
`rnfr`	**Rename From**—Designate an existing file to be renamed (must be followed by the **`RNTO`** command)
`rnto`	**Rename To**—Designate the new path and name of the file specified with the **RNFR** command

Command	Description
`site`	**Site Parameters**—Execute a command particular to the server in question (see “NetStorage FTP Extensions” on page 19)
`stat`	**Status**—Display a server status response
`stor`	**Store**—Store a file on the server, replacing an existing file of the same name, or creating a new file if one does not exist
`stou`	**Store Unique**—Store a file in the server’s current directory with a unique name
`syst`	**System**—Display the server’s operating system

NetStorage FTP Extensions

Akamai provides several FTP extensions specifically for use with NetStorage:

- Register an expected byte count
- Register an MD5 hash digest
- Retrieve an MD5 hash digest
- Register a date-time setting
- Create a symbolic link
- Copy a file to a new file name or to a directory
- Index a zip file

As you read the descriptions that follow, please note that some FTP clients do not accept the **SITE** command as an interactive input. In those cases, the client usually accepts the **QUOTE** command in some form, which allows the command to be sent.

site chksize [bytecount]

```
ftp> quote "site chksize 400000"
200 Size check set to 400000
```

This command allows an FTP client to register an expected byte count with the server so the size of the next uploaded file can be checked against that count before being committed to the NetStorage area. If the counts do not match, the transfer attempt returns an error. If there is some other error during transfer, the byte count setting is discarded and must be re-registered before retrying the upload.

site chkhash [md5digest]

```
ftp> quote "site chkhash 9524eb021939b163162f6ed6426c79d3"
200 Hash check set to 9524eb021939b163162f6ed6426c79d3
```

This command allows an FTP client to register an MD5 hash digest with the server so that the content-hash of the next uploaded file can be checked before the file is committed to the NetStorage area. If the hash values do not match, the transfer attempt returns an error. If there is some other error during transfer, the hash string is discarded and must be re-registered before retrying the upload.

Note: If the file in question is an archive file, do not run this command with the **`site az2z`** *command. Doing so will cause an error, as the* **`site az2z`** *command adds information to the archive, which will cause the MD5 checksum to fail.*

site shohash [filename]

```
ftp> quote "site shohash test.0a.base"
200 test.0a.base MD5= 9524eb021939b163162f6ed6426c79d3
```

This command allows an FTP client to retrieve the MD5 hash digest of the content in a file held by NetStorage.

site settime [yyyymmddhhmmss]

```
ftp> quote "site settime 20060430040506"
200 Time value set to 20060430040506 (1146369906)
```

This command allows an FTP client to register a date-time setting to be applied to the next uploaded file as it is committed to NetStorage. Normal FTP practice is to assign newly-uploaded files the current date-time, with some servers permitting a "modtime" operation after the fact. Since NetStorage files are moved and replicated quickly after being committed, it is not possible for them to be modified after being uploaded.

Note: The date-time setting is only good for one upload attempt. If an error occurs during transmission, the setting is discarded to avoid erroneously applying that value to a subsequent upload event.

site symlink [symlink] [target]

```
ftp> quote "site symlink linkto-dir dir"
200 linkto-dir linked to dir
```

This command allows an FTP client to create a symbolic link, **`[symlink]`**, to a target directory, **`[target]`**. A file listing then shows the resulting linkage. For example:

```
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
drwxr-xr-x 2 f1334 storage 0 Apr 29 22:30 dir
lrwxrwxrwx 1 f1334 storage 0 Apr 29 22:30 linkto-dir -> dir
226 Transfer complete.
```

site copy [src] [dest]

```
ftp> quote "site copy file1 file2"
200 file1 copied to file2
```

This command allows an FTP client to copy a source file **`[src]`** to a destination **`[dest]`**, where **`[dest]`** can be a new file name or a directory.

site az2z

```
ftp> quote "site az2z"
200 Az2z flag is set for your next upload
```

This command creates a Serve from Zip hash for the next uploaded file. If the next upload is not an archive file an error is generated.

Note: Do not run this command with **`site chkhash [MD5Digest]`**. *Doing so will cause an error, as the* **`site az2z`** *command adds information to the archive, which will cause the MD5 checksum to fail.*

Chapter 5. Using rsync

In This Chapter

The rsync program is a UNIX system-based tool that copies content from a local machine to a remote machine or between two local machines (but not between two remote machines). The tool is particularly useful because it quickly synchronizes content by transferring only the differences between changed files, not overwriting complete files. This generally saves time over other file transfer methods.

Before rsync can be used to access NetStorage, it must be enabled in the EdgeControl portal for the upload account or accounts that will be using it. Details for doing so can be found in the document *Managing Akamai NetStorage Accounts*, also available from the EdgeControl portal.

Guidelines and Best Practices

- **If possible, use the same rsync version as NetStorage.** NetStorage currently uses rsync-2.6.6, and it is strongly recommended you use the same version rsync client, as older versions can be problematic. For example, an rsync-2.6.3 client cannot use the `--delete` command with an rsync-2.6.6 server. These problems are typically generic rsync issues not specific to NetStorage.

- **Using the `--delete` command option with NetStorage's force-case feature enabled may produce undesired behavior.** NetStorage's force-case feature only affects file lookup and creation, not file enumeration. That is, end users' file requests are converted to the desired case, as are uploaded file names. But force-case will likely behave unexpectedly if used with rsync's `--delete` option (the option removes files from the destination directories being synchronized if they are no longer present on the source).

  For example, if you enable force case to force file names to lowercase, your initial upload of a local file called **TextFile.txt** will create a file called **textfile.txt** at the destination. When you attempt a new rsync operation using the `--delete` option, your client notes that it does not have a local file named **textfile.txt** and directs the server to delete it at the destination. Your client then uploads the local file **TextFile.txt**, which the server subsequently renames to **textfile.txt**.

  Two work-around options are currently available:

  - Do not use the `--delete` option. Rather, use rsync to first update any changed files. Then perform a local scan for deleted files and use CMS or another mechanism to delete those files from NetStorage.

  - Rename all local files to match your force-case settings. In this case, **`--delete`** will work as expected.

- **Using rsync on the Sun Microsystems® Solaris™ operating system.** The Solaris operating system truncates any rsync password greater than eight (8) characters—its **`getpass()`** function call in rsync takes the first eight characters only. If your password is longer than this, you can work around the problem by using the **`--password-file`** option, or by setting the environment variable, **`RSYNC_PASSWORD`**.

Caution: Do not use rsync or any sync-type tool for directory trees containing more than 100,000 files total, or containing any subdirectories with more than 5,000 files (including the root of the sync request). If your directories exceed either of these limitations in a production environment, you risk incurring possibly debilitating operational inefficiencies. In these cases you should break the rsync operation into mulitple, non-overlapping jobs that run sequentially.

Additionally, when using rsync, be certain your source environment has rigorous controls in place for both its configuration and directory image. This will help to avoid sudden large target changes as a result of accidental source changes. A common example is when accidental configuration changes occur that select a new, empty source directory to which to synchronize the target. Since the new source is empty, rsync will faithfully attempt to delete everything at the target to make it conform with it.

rsync Command Options

Following is the rsync tool's usage and a complete list of rsync command options with their definitions. You can find additional rsync information at **http://rsync.samba.org/ftp/rsync/rsync.html**.

```
rsync [option]... src [src]... dest
rsync [option]... src [src]... [user@]host:dest
rsync [option]... src [src]... [user@]host::dest
rsync [option]... src [src]... rsync://[user@]host[:port]/dest
rsync [option]... src
rsync [option]... [user@]host:src [dest]
rsync [option]... [user@]host::src [dest]
rsync [option]... rsync://[user@]host[:port]/src [dest]
```

Unsupported Command Options

NetStorage supports the normal command set for rsync and rsync/SSH, with the exception of the following options:

Option	Description
`--inplace`	Perform an in-place file overwrite rather than taking advantage of the rsync algorithm
`-H --hard-links`	Duplicate any hard links present at the source on the destination

Option	Description
--partial	If a transfer interruption results in partial file, do not automatically delete the file
--partial-dir=*DIR*	If a transfer interruption results in partial file, do not automatically delete the file and deposit the file in directory ***DIR***
--delay-updates	Wait until the transfer operation has ended to put the temporary files in place
-T --temp-dir=*DIR*	Specify a scratch directory for temporary file generation
--link-dest=*DIR*	If the source files differ from those inside ***DIR*** copy them to the destination, and also create hard links from any unchanged files in ***DIR*** to the destination

Supported Command Options

Following are supported rsync command options and their descriptions:

Option	Description
-v --verbose	Display the operation's execution step by step
-q --quiet	Display only error messages
-c --checksum	Use a 128-bit MD4 checksum to determine which files to update, rather than modification time and file size
-a --archive	Use archive mode, which executes the **-rlptgoD** options
-r --recursive	Copy recursively
-R --relative	Include the source file's relative path when copying the file to the destination
--no-relative	Disable the **--relative** option; for use with the **--files-from** option
--no-implied-dirs	Use with the **--relative** option; cause a copy operation to assume the source file's relative path already exists at the destination
-b --backup	Retain existing destination files with new file names suffixed with "~"
--backup-dir=*DIR*	Use with the **--backup** option; specify a directory destination for backup files
--suffix=*SUFFIX*	Use with the **--backup** option; replace the default suffix, "~", with ***SUFFIX***
-u --update	Do not copy files on the destination with newer modification times than the source files
-d --dirs	Copy encountered directories, but not their contents
-l --links	Re-create source symlinks at the destination

Option	Description
-L --copy-links	When encountering symlinks, copy the referenced file instead
--copy-unsafe-links	When encountering symlinks that reference files outside the operation’s copy path hierarchy, copy the referenced file instead
--safe-links	Disregard symlinks referencing files outside the operation’s copy path hierarchy
-K --keep-dirlinks	If the source directory is real, but the destination is a symlink referencing another directory, copy the source directory’s contents to the symlink target
-p --perms	Match destination permissions to those of the source
-o --owner	Change destination file ownership to that of the source file (reserved for the super-user)
-g --group	Change destination file group to that of the source file if the destination is running as the super-user, else preserve only the groups of which the destination is a member
--devices	Copy character and block device files to the destination (reserved for the super-user)
-t --times	Include files’ modification times in transfer
-O --omit-dir-times	Used with **--times**; omit directories
-S --sparse	Deal with sparse files in an efficient manner
-n --dry-run	Display the actions that would take place during the operation without actually performing it
-W --whole-file	Override the rsync algorithm and copy the entire file
--no-whole-file	Disable the **--whole-file** option
-x --one-file-system	Do not recurse beyond the boundary of the current filesystem
-B --block-size=*SIZE*	Set the block size to a static value rather than basing it on the individual file size
-e --rsh=*CMD*	Use an alternative remote shell for the rsync operation
--rsync-path=*PRGRM*	Choose the program that starts rsync on the remote machine
--existing	Do not copy files that do not already exist at the destination
--ignore-existing	Do not copy files that already exist at the destination
--remove-sent-files	Remove files and symlinks from the source that are being newly created or the content of which has been updated on the destination
--del	The same as the **--delete-during** option

Option		Description
	--delete	Remove files from the destination directories being synchronized if they are no longer present on the source
	--delete-before	Remove files from the destination directories being synchronized before the transfer occurs (this is the default)
	--delete-during	Remove files from the destination directories being synchronized during the transfer
	--delete-after	Remove files from the destination directories being synchronized after the transfer
	--delete-excluded	Remove files from the destination directories being synchronized if they are no longer present on the source and also removed files defined with the **--exclude** option
	--ignore-errors	Used with the **--delete** option to tell the destination to perform deletions despite any I/O errors
	--force	Used with the **--recursive** or **--archive** options; delete empty and non-empty destination directories that are being replaced by non-directories
	--max-delete=*NUM*	Do not remove more than ***NUM*** files or directories
	--max-size=*SIZE*	Do not copy files larger than ***SIZE***; suffix with **k**, **m**, or **g** to indicate kilobytes, megabytes, and gigabytes, respectively
	--numeric-ids	Map numeric user IDs and group IDs rather than their user and group names
	--timeout=TIME	Set the maximum timeout for I/O (in seconds)
-I	**--ignore-times**	Copy files that match in size and modification time between source and destination (normally passed over)
	--size-only	Do not copy files with matching sizes even if their time-stamps differ
	--modify-window=*NUM*	Consider files' timestamps to be identical if they fall within ***NUM*** seconds of each other (default is 0 seconds)
-y	**--fuzzy**	If the destination file is missing, look in the destination directory for a file of the exact same size and modification time or with a similar file name to use as a basis
	--compare-dest=*DIR*	If the source files differ from those inside ***DIR*** copy them to the destination
	--copy-dest=*DIR*	If the source files differ from those inside ***DIR*** copy them to the destination, and also perform a local copy of any unchanged files from ***DIR*** to the destination

Option	Description
`-z --compress`	Use compression on the data during the operation
`-C --cvs-exclude`	Use the CVS algorithm to exclude multiple files from the operation
`-f--filter=`***`RULE`***	Include a ***`RULE`*** to filter out particular files

`-F`	The first occurrence sets the filtering rule:
	`--filter='dir-merge /.rsync-filter'`
	The second occurrence sets the rule:
	`--filter='- .rsync-filter'`

`--exclude=`***`PATTERN`***	Do not include files that match ***`PATTERN`***
`--exclude-from=`***`FILE`***	Do not include files that match patterns defined in the ***`FILE`*** file
`--include=`***`PATTERN`***	Include files that match ***`PATTERN`***
`--include-from=`***`FILE`***	Include files that match patterns defined in the ***`FILE`*** file
`--files-from=`***`FILE`***	Transfer only source files included in the ***`FILE`*** file
`-0 --from0`	End each entry read from a file with a '0' (pertains to **`--exclude-from`**, **`--include-from`**, **`--files-from`**, and **`--filter`**; **`--cvs-exclude`** is not affected)
`--address=`***`ADDRESS`***	Bind rsync to a specific IP address or hostname
`--port=`***`PORT`***	Use ***`PORT`*** instead of the default, port 873
`--blocking-io`	Use blocking I/O when using a remote shell
`--stats`	Display the file transfer's statistics
`--progress`	Display information on the file transfer's progress
`-P`	Identical to **`--partial --progress`**
`-i --itemize-changes`	Display information on file changes made during the transfer
`--log-format=`***`FORMAT`***	Use ***`FORMAT`*** for per-file output
`--password-file=`***`FILE`***	Get the password for remote rsync daemon access from ***`FILE`***
`--list-only`	Display a list of source files in the operation rather than copying them to the destination
`--bwlimit=`***`KBPS`***	Set the maximum rate of transfer (in kilobytes per second)
`--write-batch=`***`FILE`***	Create a file for later use with the **`--read-batch`** option
`--only-write-batch=`***`FILE`***	Create a file for later use with the **`--read-batch`** option, but do not update the destination

Option	Description
--read-batch=*FILE*	Use the file created with the **--write-batch** or **--only-write-batch** options
--protocol=*NUM*	Use an older protocol version (***NUM***)
--version	Display the rsync version information
-h --help	Display the help information

Chapter 6. Using Secure Copy

In This Chapter

Secure Copy (SCP) provides a secure means of copying files to and from NetStorage using SSH for authentication and security. It is, however, limited to file transfers and cannot be used for other management tasks. Because of this limitation, you may wish to use SFTP (SSH File Transfer Protocol), which has largely superseded SCP and is a more capable SSH-based tool. Please read “Using Secure Access” on page 13 before using any of the secure NetStorage access methods.

SCP Command Options

NetStorage supports the normal command set for SCP (except as noted below), which is provided below. You can find more details at **http://www.openbsd.org/cgi-bin/man.cgi?query=scp**.

```
scp [-1234BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
      [-l limit] [-o ssh_option] [-P port] [-S program]
      [[user@]host1:]file1 [...] [[user@]host2:]file2
```

Option	Description
-1	Use SSH protocol 1 *Note: NetStorage does not support SSH protocol 1*
-2	Use SSH protocol 2
-4	Use only IPv4 addresses
-6	Use only IPv6 addresses
-B	Use batch mode
-C	Use compression
-c ***cipher***	Specify the cipher for encrypting the data transfer
-F ***ssh_config***	Designate a substitute per-user SSH configuration file
-i ***identity_file***	Specify the private SSH key file
-l ***limit***	Confine the bandwidth used to ***limit*** (given in kilobits per second)

Option	Description
`-o` *ssh_option*	Send the specified option to SSH in the ssh_config format (for details on these options, refer to **http://www.openbsd.org/cgi-bin/man.cgi?query=ssh_config**): AddressFamily BatchMode BindAddress ChallengeResponseAuthentication CheckHostIP Cipher Ciphers Compression CompressionLevel ConnectionAttempts ConnectTimeout ControlMaster ControlPath GlobalKnownHostsFile GSSAPIAuthentication GSSAPIDelegateCredentials HashKnownHosts Host HostbasedAuthentication HostKeyAlgorithm HostKeyAlias HostName IdentityFile IdentitiesOnly KbdInteractiveDevices LogLevel MACs NoHostAuthenticationForLocalHost NumberOfPasswordPrompts PasswordAuthentication Port PreferredAuthentications Protocol ProxyCommand PubKeyAuthentication RekeyLimit RhostsRSAAuthentication RSAAuthentication SendEnv ServerAliveInterval ServerAliveCountMax SmartcardDevice StrictHostKeyChecking TCPKeepAlive UsePrivilegedPort User UserKnownHostsFile VerifyHostKeyDNS
`-P` *port*	Connect to ***port*** on the remote host
`-p`	Save the original file's modifications times, access times, and modes

Option	Description
`-q`	Turn off the progress meter
`-r`	Copy recursively
`-S` *`program`*	Specify the program to use for connecting to the destination
`-v`	Display the operation's execution step by step

Using SCP Clients

NetStorage fully supports the OpenSSH SCP client, as well as the PuTTY PSCP client. The WinSCP client allows you to choose between SCP and SFTP backend protocols, but it only works with NetStorage when it is configured to exclusively use the SFTP backend protocol.

Chapter 7. Using SSH File Transfer Protocol

In This Chapter

SSH File Transfer Protocol (SFTP) is an OpenSSH-based remote file system protocol similar to FTP, but which allows for encrypted transfers and lower-level file I/O calls against the server (for example, open/read/write/close, opendir/readdir/closedir, unlink, rename, and symlink). SFTP is also far more comprehensive than SCP (Secure Copy)—which it has largely superseded—in that you can manage your content beyond simple data transfers.

Despite its name, do not confuse SFTP with the following unrelated protocols:

- **Simple File Transfer Protocol** (RFC 913).
- **File Transfer Protocol** (RFC 959).
- **FTP over SSH.** This is FTP encapsulated inside an SSH session.
- **FTPS.** This is FTP encapsulated inside an SSL/TLS session.

SFTP Commands and Options

NetStorage supports the full SFTP command set (except as noted below), details for which you can obtain at **http://www.openbsd.org/cgi-bin/man.cgi?query=sftp**.

Note: The following list of commands and options are specific to the OpenSSH SFTP client and can vary depending on the client used. For example, the PuTTY PSFTP client adds options (e.g., recursive directory traversal) not available to the OpenSSH client.

```
sftp [-1Cv] [-B buffer_size] [-b batchfile] [-F ssh_config]
      [-o ssh_option] [-P sftp_server_path] [-R num_requests]
      [-S program] [-s subsystem | sftp_server] host
sftp [[user@]host[:file [file]]]
sftp [[user@]host[:dir[/]]]
sftp -b batchfile [user@]host
```

Command or Option	Description
`-1`	Use SSH protocol 1 *Note: NetStorage does not support SSH protocol 1*
`-B buffer_size`	Set the file transfer buffer size (32768 bytes is the default)
`-b batchfile`	Specify the command file to use in batch mode
`-C`	Use compression

Command or Option	Description
-F *ssh_config*	Designate a substitute per-user SSH configuration file
-o *ssh_option*	Send the specified option to SSH in the format for ssh_config (for details on these options, refer to **http://www.openbsd.org/cgi-bin/man.cgi?query=ssh_config**): AddressFamily BatchMode BindAddress ChallengeResponseAuthentication CheckHostIP Cipher Ciphers Compression CompressionLevel ConnectionAttempts ConnectTimeout ControlMaster ControlPath GlobalKnownHostsFile GSSAPIAuthentication GSSAPIDelegateCredentials HashKnownHosts Host HostbasedAuthentication HostKeyAlgorithm HostKeyAlias HostName IdentityFile IdentitiesOnly KbdInteractiveDevices LogLevel MACs NoHostAuthenticationForLocalHost NumberOfPasswordPrompts PasswordAuthentication Port PreferredAuthentications Protocol ProxyCommand PubKeyAuthentication RekeyLimit RhostsRSAAuthentication RSAAuthentication SendEnv ServerAliveInterval ServerAliveCountMax SmartcardDevice StrictHostKeyChecking TCPKeepAlive UsePrivilegedPort User UserKnownHostsFile VerifyHostKeyDNS

Command or Option	Description
-P *sftp_server_path*	Connect to a local SFTP server without using SSH
-R *num_requests*	Specify the number of allowed concurrent requests
-S *program*	Specify the program to use for connecting to the destination
-s *subsystem* | *sftp_server*	Designate the subsystem for the SSH2 protocol or an SFTP server's path
-v	Increase the amount of logging
Interactive Commands	
bye	End the SFTP session
cd *path*	Change to a new remote directory
chgrp *grp path*	Change a file's group to ***grp***
chmod *mode path*	Change a file's permissions to ***mode***
chown *own path*	Change a file's owner to ***own***
exit	End the SFTP session
get [-P] ***remote-path***	Copy content from the remote machine to the local machine (using **-P** includes full file permissions and access times)
help	Display help information
lcd *path*	Change to a new directory on the local machine
lls [*ls-options* [*path*]]	Display the contents of a directory on the local machine (***ls-options*** is any option available to the local machine's **ls** command)
lmkdir *path*	Create a new directory on the local machine
ln *oldpath newpath*	Generate a symbolic link
lpwd	Display the local machine's working directory
ls [-1aflnrSt] [*path*]	Display the contents of a directory **-1** List the output in a single column **-a** Include files that start with a dot (".") **-f** Keep the list unsorted **-l** Include additional file information such as permissions and ownership **-n** Display a numerically-presented long listing that includes user and group information **-r** Sort in reverse order **-S** Use file size to sort the list **-t** Use last modification time to sort the list
lumask *umask*	Set the umask on the local machine
mkdir *path*	Create a new directory on the remote machine

Command or Option	Description
`progress`	Enable and disable the progress meter
`put [-P]` ***`local-path`*** **`[`*****`remote-path`*****`]`**	Copy content from the local machine to the remote machine (using **`-P`** includes full file permissions and access times)
`pwd`	Display the path of the current remote directory
`quit`	End the SFTP session
`rename` ***`oldpath newpath`***	Change a file's name
`rm` ***`path`***	Remove an existing file
`rmdir` ***`path`***	Delete an existing directory
`symlink` ***`oldpath newpath`***	Generate a symbolic link
`version`	Show the version of the SFTP protocol
`!` ***`command`***	Use the local shell to perform a command
`!`	Temporarily return to the local shell
`?`	Display help information

Using SFTP Clients

Several available SFTP clients are usable with NetStorage, including OpenSSH, PuTTY, and WinSCP. There are several points worthy of note when using these.

Using OpenSSH

The OpenSSH SFTP client is fully interoperable with NetStorage. Following are notes about the directory listing (**`ls`**) command when using OpenSSH versions 3.4p1 and earlier.

- Output is unsorted (version 3.5p1 introduced sorting).
- The "`.`" and "`..`" entries are always displayed (version 3.5p1 introduced the filtering of "hidden" dotfiles, as well as a new **`-a`** option that displays these entries).
- The command's argument must be a directory (version 3.5p1 removed this restriction so the command can also be used on other object types such as files and symbolic links).

Using PuTTY

The PuTTY PSFTP clients is fully interoperable with NetStorage, as is the PuTTY PSCP client's **`-sftp`** backend mode of operation. Following are some user-interface differences between the PuTTY and OpenSSH SFTP clients:

- The PuTTY PSFTP client does not support globbing (wildcard characters) as fully as the OpenSSH client; this also affects PSCP in the **`-sftp`** backend mode.

- The PuTTY PSFTP client does not support the following constructs or commands in either interactive usage or “batch” mode:
  - Leading **#** characters are not treated as comments as they are with the OpenSSH client; they are treated as syntax errors.
  - Globbing expressions (wildcard characters) are not supported.
  - The **get** and **put** commands have no option to preserve transferred files’ permission and access time attributes (i.e., there is no **-P** option as with the OpenSSH client).
  - There is no **symlink** command.
- The PuTTY PSFTP client has two commands not present in the OpenSSH SFTP client: **reget** and **reput**. These exist to allow the user to resume an interrupted upload or download in which the client or server has only some prefix portion of the file in question.
  - **reget (resume download).** The client tells the server to seek to the position corresponding to the end of the portion of the file on the client, and to transfer the remaining bytes.
  - **reput (resume upload).** The client queries to see how much of the file exists on the server, and then only uploads the remaining portion of the file.

Note: When performed on NetStorage, **reput** *will always result in a full upload.*

Using WinSCP

As previously stated, SFTP provides opendir/readdir/closedir interfaces, which are suitable for browsing support. The popular OpenSSH and PuTTY SFTP clients provide rudimentary filesystem browsing support, but WinSCP (**www.winscp.net**), an open-source SCP/SFTP client, offers a graphical user interface.

Like PuTTY PSCP, WinSCP allows you to choose between SCP and SFTP backend protocols. Unlike PuTTY, WinSCP only works with NetStorage when it is configured to exclusively use the SFTP backend protocol (the option to prefer SFTP with fallback to SCP will not work).

WinSCP requires a very specific configuration in order to work with NetStorage:

1. Start WinSCP.

   The **WinSCP Login** dialog box appears.

2. Set the session parameters.

   a. Click **Session** in the left-hand option tree window.

   b. In the **Host name** text box, enter your NetStorage host name (e.g., example.upload.akamai.com).

   c. In the **User name** text box, enter **sshacs**.

   d. If you are not using Pageant, the PuTTY SSH agent, use the **Private key file** text box’s browse button navigate to your SSH private key.

  e. In the **Protocol** area, select the **SFTP** radio button.

3. Set the environment parameters.

   a. In the left-hand option tree window under **Environment**, click **Directories**.

   b. In the **Remote directory** text box, enter your NetStorage working directory.

   c. In the **Local directory** text box, enter the directory on your local machine in which you would like to begin your SFTP session.

4. Set the SSH parameters.

   a. In the left-hand option tree window, click **SSH**.

   b. In the **Protocol** options area, under **Preferred SSH protocol version**, select the **2** or **2 only** radio button.

5. Click the **Login** button.

   a. If you used the **Private key file** text box to enter your SSH private key, enter your key file password when prompted.

   WinSCP starts and your SFTP session begins.

Your NetStorage connection will time out after approximately 15 minutes of inactivity (for example, no data transfers or file browsing). If, for some reason, you must leave WinSCP idle and connected for longer than 15 minutes, you can override the timeout by configuring **Keepalives** in the **WinSCP Login** dialog box prior to logging in. To do so, click **Connection** in the left-hand option tree window, and in the **Keepalives** area, select the **Executing dummy protocol commands** radio button and enter **99** in the **Seconds between keepalives** spin box.

Note: The SFTP timeout is intended to help reduce load on the NetStorage server. Please do not enable the Keepalive mechanism unless absolutely necessary.

Using Proprietary SFTP Clients

Other proprietary, third-party SFTP clients—South River Technologies' WebDrive® client, SSH Communications Security's SSH Tectia® Client, and GlobalSCAPE's CuteFTP® Professional client—are also usable with NetStorage.

WebDrive Client

Like WinSCP, WebDrive (**www.southrivertechnologies.com**) provides a graphical user interface to access NetStorage using SFTP. It operates at a lower level than WinSCP, however, and provides a remote filesystem so you can access files on NetStorage as if they were on a local or network drive. You can manipulate files using Windows Explorer or a command line.

SSH Tectia Client

SSH Tectia Client (**www.ssh.com**) generates a commercial implementation of SSH keys commonly known as the "ssh.com format." If you generate your keys with SSH Tectia Client, you must convert the public key to OpenSSH format as described in Chapter 3 before uploading it in the EdgeControl portal.

Note: NetStorage imposes a 1000-character limit on public keys, so SSH-2 RSA public keys must be 2048 bits or less, and SSH-2 DSA keys must be 1024 bits or less.

CuteFTP Professional Client

Like SSH Tectia Client, CuteFTP Professional (**www.globalscape.com**) generates SSH keys using the "ssh.com format." If you generate your keys with CuteFTP Professional, you must convert the public key to OpenSSH format as described in Chapter 3 before uploading it in the EdgeControl portal.

Note: NetStorage imposes a 1000-character limit on public keys, so SSH-2 RSA public keys must be 2048 bits or less, and SSH-2 DSA keys must be 1024 bits or less.

Chapter 8. Using Akamai's Content Management Server

In This Chapter

Akamai's Content Management Server (CMS) allows you to log in to a shell-like interface to manage content in NetStorage using a secure environment (see "Using Secure Access" on page 13). This server supports both interactive login and single command execution.

When you log in or access by single command, your user name is always **sshacs**, the name used for all secure access to NetStorage.

Interactive Login:

```
ssh -t sshacs@company_name.upload.akamai.com cms
```

Single Command:

```
ssh sshacs@company_name.upload.akamai.com [command line]
```

Note: CMS supports globbing (e.g., `ls *.txt` *and* `cat *.txt > output.txt`*), but be aware the sequence of the resulting output is not guaranteed to be in an expected (e.g., alphanumeric) order.*

CMS Commands

ls — Display the contents of a directory

```
ls [OPTION]... FILE...
```

`-l --list`	Include additional file information such as permissions and ownership
`-Q --quote-name`	Place entry names in double quotes
`-r --reverse`	Sort in reverse order
`-S`	Use file size to sort the list
`-t`	Use modification time to sort the list
`-U`	Leave the list unsorted and display the contents in directory order
`-X`	Use entry extension to sort alphabetically
`-h --help`	Display help information

mv — Rename a single file, or move a single or multiple files

```
mv [OPTION]... SOURCE DEST
mv [OPTION]... SOURCE... DIRECTORY
```

Note: `SOURCE` *cannot be a directory, as directories cannot be moved or renamed.*

`-i --interactive`	Ask before overwriting existing files
`-v --verbose`	Display the operation's execution step by step
`-h --help`	Display help information

cp — Copy single or multiple files or directories

```
cp [OPTION]... SOURCE DEST
cp [OPTION]... SOURCE... DIRECTORY
```

`-i --interactive`	Ask before overwriting existing files
`-p --preserve`	Copy the attributes for each file, as well
`-r`	Use recursion, creating files from non-directories
`-v --verbose`	Display the operation's execution step by step
`-h --help`	Display help information

rm — Delete single or multiple files or directories

```
rm [OPTION]... FILE...
```

Note: If using interactive mode and your confirmation response does not begin with `y` *or* `Y`*, the file is skipped.*

`-i --interactive`	Ask before removing files
`-r --recursive`	Use recursion in deleting contents
`-v --verbose`	Display the operation's execution step by step
`-h --help`	Display help information

mkdir — Create one or more new directories

```
mkdir [OPTION] DIRECTORY...
```

When using this command with CMS, the `-p` option is implied. That is, no error is generated if the directory already exists, and parent directories will be created if needed.

`-v, --verbose`	Display the operation's execution step by step
`-h, --help`	Display help information

rmdir — Delete single or multiple empty directories

```
rmdir [OPTION]... DIRECTORY...
```

The directories must be empty for this command to work.

`-v, --verbose`	Display information for each deleted directory
`-h, --help`	Display help information

ln — Create a symbolic link to a file or a directory

```
ln [OPTION]... TARGET LINKNAME
ln [OPTION]... TARGET... DIRECTORY
```

Generate a symbolic link called **`LINKNAME`** that points to file or directory **`TARGET`**., or generate symbolic links to multiple targets and place them in another **`DIRECTORY`**.

Note: NetStorage does not support the creation of hard links. When using the **`ln`** *command in CMS, the* **`-s`** *(create a symlink) option is implied.*

`-v`	**`--verbose`**	Display the operation's execution step by step
`-h`	**`--help`**	Display help information

cd — Change to a different remote directory

```
cd DIRECTORY
```

Options: none.

pwd — Display the path of the current remote directory

```
pwd [OPTION]
```

`-h`	**`--help`**	Display help information

cat — Concatenate a file or files and display the result, or concatenate multiple files into a single output file

```
cat [OPTION] FILE...
cat [OPTION] FILE1 FILE2 FILE3... > OUTPUTFILE
```

`-h`	**`--help`**	Display help information

stat — Display information on one or more files, or a file system

```
stat [OPTION] FILENAME [FILENAMES...]
```

Be aware that output is not always the same. Here is a sample output from **stat**:

```
File: /
Size: 1024
Filetype: Directory
Mode: (0755/drwxr-xr-x)
Uid: cpcode9391
Gid: storage
Modify: Wed May 17 09:32:09 2006(00021.03:08:08)
Md5sum: -
```

`-h`	**`--help`**	Display help information

du — Estimate how much disk space each file or subdirectory uses

```
du [OPTION]... FILE or SUBDIRECTORY...
```

Summarize the disk usage in kilobytes, (1000, not 1024) of each **FILE** or **SUBDIRECTORY**. Note that this command may give a different result than some other disk usage utilities, because (1) it divides total bytes by 1000, not 1024, to arrive at kilobytes, and (2) it calculates the total file size by adding up the bytes of all files, then divides by 1000 to give a kilobyte result. It does not round up for block usage (for example, it does not assign a 525-byte file to a size of 1024 based on 525 using two 512-byte blocks), and it does not divide by 1024.

	--block-size=*SIZE*	Use a block size other than the default
-f		Include the number of files inside subdirectories
-S	**--separate-dirs**	Omit subdirectory sizes
-s	**--summarize**	Display totals for the specified file(s) and/or subdirectory(ies) only
	--max-depth=*N*	Display totals for the specified file(s) and/or subdirectory(ies), and also for any subdirectories if they are within ***N*** sublevels of any specified directory.
-h	**--help**	Display help information

hostname — Display the IP address of the remote machine

```
hostname [OPTION]
```

-h	**--help**	Display help information

help — Display the list of valid CMS commands

```
help [OPTION] [COMMAND]
```

-h	**--help**	Display help information

unzip — Perform various operations on a ZIP archive file's contents

```
unzip [-flptuvz] [MODIFIER] ZIPFILE [-dx] FILE_or_DIRECTORY
```

Options:

-f	Uncompress only files that pre-exist on the disk and that are updated, use them to refresh the existing files, and if the files do not pre-exist, do not uncompress them and create them on the disk
-l	Display the ZIP archive file's contents using a shortened format
-d	Uncompress files to **directory**
-p	Display the contents of all files contained in the ZIP archive file using binary format
-t	Check the ZIP archive file's contents for errors
-u	Uncompress all files, updating existing ones, and creating new ones as necessary
-v	Display the ZIP archive file's contents using the verbose (long) format
-x	Do not extract the file or files that follow

-z	Show the comment in the ZIP archive file, if present

Modifiers:

-a	If files are recognized as text, extract them as such, not as binary
-aa	Extract **all** files in the archive as text files, regardless of whether they actually are
-b	Extract all files as binary, regardless of whether they are text
-C	Do not apply case-sensitivity when extracting files
-j	Do not re-create the ZIP archive file's directory structure when extracting files; extract all files to the working directory
-L	If a file was created on an uppercase-only system, convert its file name to lowercase
-LL	Convert all uppercase file names to lowercase
-n	If files already exist, do not overwrite them
-o	Do not prompt before overwriting files
-q	Perform the operation quietly; do not display the operation's step-by-step execution
-qq	Perform the operation even more quietly than with **-q**
-v	Display diagnostic information (used as **unzip -v**)
-X	Preserve user and group IDs

md5sum — Display the MD5 digest for one or more files

```
md5sum [OPTION] FILE...
```

-r	Execute the command recursively
-h --help	Display help information

*Note: If you run this command after the **az2z** command, its hash will differ from that of your local version.*

az2z — Rewrite the ZIP archive file to Akamai ZIP file format (verify the file's integrity and create a hash table for it)

```
az2z [OPTION] FILE
```

-h --help	Display help information

exit - Terminate the CMS session

```
exit
```

Options: none.

Using Akamai’s Content Management Server Site Snapshot Tool (SST)

CMS provides support for Akamai’s Site Snapshot Tool (SST), which may be used for reliable transfer of large amounts of data. This option is available to some customers under contracts with Akamai Professional Services. For more information, please contact your Akamai representative.

Note: By default, SST is limited to 50,000 files per operation. If necessary, however, you may override this limit with the **`--upload-quota`** *option.*

CMS SST Command Options

```
sst [OPTION]... [URL]...
```

Examples:

```
sst -r -N -nH http://www.example.com
```

Download www.example.com, and recurse through links to download those files as well. Download only those files that are newer than the ones already on the failover site, and do not create a host directory www.example.com/.

```
sst -p --load-cookies=cookies.txt --header="Referer: http://
      www.example.com/index.jsp” http://www.example.com
```

Load the cookies file and download based on its contents; include page requisites, and add this Referer header.

```
sst -p -N --input-file=clipart.txt
```

Download the files listed in clipart.txt and the page requisites needed to display the pages, but download files only if they are newer than the ones on the failover site.

Following is the full list of command options

`-A --accept=`*`LIST`*	Download only files with the extensions or patterns specified in ***LIST***
`-B --base=`*`URL`*	Prepend any relative links in an input file with ***URL***
`-D --domains=`*`LIST`*	Follow only domains specified in ***LIST***
`-F --force-html`	Regard an input file to be HTML
`-g --glob=`*`on/off`*	Enable or disable globbing to allow or disallow the special wildcard characters
`-G --ignore-tags=`*`LIST`*	Follow all of the default HTML tags that are normally followed during recursion (see the “`-r --recursive`” option below) except those specified in ***LIST***
`-h --help`	Display help information
`-H --span-hosts`	Allow recursion to move to other hosts (must be used with the **`--domains=`*`LIST`*** option)
`-i --input-file=`*`FILE`*	Get the list of URLs to download from ***FILE***

-I --include-directories=*LIST*	Follow only directories specified in ***LIST***
-k --convert-links	Change absolute hyperlinks to relative
-l --level=*NUMBER*	Limit recursion depth to ***NUMBER*** levels
-L --relative	Do not follow any links except relative ones
-m --mirror	Enable options necessary to perform mirroring
-nd --no-directories	When performing recursive downloads, do not re-create the site’s directory hierarchy structure; instead, copy all files to the working directory
-nH --no-host-directories	Do not include a hostname directory in the hierarchy
-N --timestamping	Only download files if they are newer than the existing ones
-o --output-file=*FILE*	Send operation information to ***FILE*** instead of the standard output
-O --output-document=*FILE*	Do not download files, but concatenate their contents and write them to ***FILE***
-q --quiet	Do not display the operation’s step-by-step execution
-Q --quota=*NUMBER*	Set a byte-count limit for downloading multiple files recursively or from an input file (suffix with “k” for kilobytes or “m” for megabytes)
-p --page-requisites	In addition to the specified HTML page, also download any other files required to display the page correctly
-P --directory-prefix=*PREFIX*	Download all files and subdirectories to a directory called ***PREFIX***
-r --recursive	Download with recursion (**use this option with caution**)

By default, if you use recursion the following tags/attributes will be followed:

- a/href
- applet/code
- area/href
- bgsound/src
- body/background
- embed/href, src
- fig/src
- frame/src
- iframe/src
- img/href, lowsrc, src
- input/src
- layer/src
- overlay/src
- script/src
- table/background
- td/background
- th/background
- base/href
- link/href

-R --reject=*LIST*	Download all files except those with the extensions or patterns specified in ***LIST***
-S --server-response	Display sent HTTP server headers and FTP server responses
-t --tries=*NUMBER*	Make ***NUMBER*** attempts to download each URL (20 is the default; use 0 to make unlimited retries)
-T --timeout=*SECONDS*	Do not allow DNS lookups, connections attempts, and read idle times to exceed ***SECONDS***

-v --verbose	Display the operation's execution step by step (this is implied when using the sst command)
-w --wait=*SECONDS*	At the end of a file retrieval, wait ***SECONDS*** before retrieving the next file
-x --force-directories	Re-create the directory hierarchy, regardless of whether one normally would be created
-X --exclude-directories=*LIST*	Follow all directories except those specified in ***LIST***
-z --convert-absolute	Change relative hyperlinks to absolute
--exclude-domains=*LIST*	Follow all domains except those specified in ***LIST***
--follow-ftp	Do not ignore FTP links within HTML pages
--follow-tags=LIST	Follow only a subset ***LIST*** of the default HTML tags that are followed when recursing (see the "-r --recursive" option above).
--header=*STRING*	Include ***STRING*** with HTTP requests' headers
--http-passwd=*PASS*	Specify the HTTP server's password
--http-user=*USER*	Specify the HTTP server's user
--ignore-robots	Do not honor the robot.txt file or the robots metatag
--limit-rate=*RATE*	Do not download faster than ***RATE*** (suffix with "k" for kilobytes/second or "m" for megabytes/second)
--load-cookies=*FILE*	Prior to the first download, load the cookies contained in ***FILE*** *The cookie file format is:* *domain ignore path secure expires name*
--no-clobber	Do not download a file if it already exists in the working directory
--no-http-keep-alive	Disable the persistent connection feature
--no-parent	When using recursion, never ascend to the starting point's parent directory
--passive-ftp	Use passive mode to require the client to start communications with the server
--random-wait	If a file fails to download, wait either 0xWAIT, 1xWAIT, and 2xWAIT, determined randomly, before reattempting the download (WAIT is the ***SECONDS*** value set with the **--wait=*SECONDS*** option)
--retr-symlinks	Ignore symbolic links when performing recursive download, and download the link targets instead, unless the target is a directory
--save-cookies=*FILE*	Before quitting the session, save all of the valid cookies to ***FILE*** *The cookie file format is:* *domain ignore path secure expires name*

`--spider`	Check for the presence of files without actually downloading them
`--upload-quota=`*`QUOTA`*	Override the default 50,000 file limit for this operation only, and set the new limit to ***`QUOTA`*** (e.g., **`--upload-quota=100000`**)
`--waitretry=`*`SECONDS`*	If a file fails to download, reattempt after 1 second; if it again fails to download, wait 2 seconds and try again, and so on until ***`SECONDS`*** between attempts is reached and then stop

Chapter 9. Serving Content from Archive Files

In This Chapter

NetStorage offers a “Serve from Zip” feature that allows it to dynamically examine archive files and serve individual files from within them directly. Without this feature, an archive file would have to be completely unzipped to serve its contents, which could be a lengthy process if it contained a large number of files. Serve from Zip replaces the unzip process with an indexing command that verifies the file’s integrity and adds a hash table to facilitate locating specific files within.

Serve from Zip offers the following benefits:

- Eliminating the unzip process makes individual files available almost immediately.
- Since many individual files are compressed into a single archive, file installation bottlenecks are bypassed.
- Contents that are updated and deleted in groups can be managed as one file rather than many separate files.

Indexing an Archive File

To use the Serve from Zip feature, you must first index the archive file to create a hash table of its contents. This table is then added to the archive as a comment.

Note: If you unzip an indexed archive file with a tool that displays comments (CMS’s **`unzip`** *command, for example), the table hash will be listed among the archive’s contents, appearing as garbled text.*

FTP (Chapter 4) and CMS (Chapter 8) both use the **`az2z`** command to create the hash table. The command’s use differs slightly for each interface, however.

To index an archive file with FTP, enter **`site az2z`** to set the command and then upload the file. Be aware that an error will be generated if the file uploaded after entering the command is not an archive file. Also, the command must be reentered prior to each archive file upload.

If using CMS, upload the archive file to NetStorage using whatever means you prefer. When finished, log in to CMS and run the `az2z [archive_file]` command for each archive.

How NetStorage Serves Content from Archive Files

When serving from an archive file, NetStorage treats it as a directory within the hierarchy of your NetStorage content.

Example 1 You might upload an archive file to `/cp_code/a/b/c.zip`. If an Edge server then sends an end-user request for `company_name.download.akamai.com/cp_code/a/b/c/d/e.html`:

1. NetStorage begins by looking for the full path.
2. If the file is not found, it begins searching backward up the path, starting with `.../cp_code/a/b/c/d.zip`.
3. If that file also is not found, it looks for `.../cp_code/a/b/c.zip`.
4. Finding that file, NetStorage looks inside the archive's index for `/d/e.html`, which it retrieves and serves to the end user.

Example 2 In a more complex scenario, assume there are two zip files along the same path called `.../cp_code/a/b.zip` and `.../cp_code/a/b/c.zip`. Once again, an end user requests `.../cp_code/a/b/c/d/e.html`:

1. NetStorage limits its search to the archive furthest along a given path, so in this case NetStorage will search `c.zip`.
2. If it contains `/d/e.html` the file is served.
3. If it does not contain the file, a 302 error is returned, even if `b.zip` contains `/c/d/e.html`.

This is something to take into account when structuring your content.

Lastly, you may override files inside an archive file by uploading them individually external to the archive. Keep in mind, however, that the upload path must match that of the file in the archive. For example, if the archive `.../cp_code/a/b/c.zip` contains a file at `d/e.html`, then the overriding file must be uploaded to `.../cp_code/a/b/c/d/e.html`.

Serve from Zip Considerations

Some issues to bear in mind when using Serve from Zip:

- When indexing archive files, NetStorage translates backward slashes “\” to forward slashes “/”. This can be problematic if the archive contains any UNIX system-based files having that character in their name. You can overcome this problem by uploading the file external to the archive.

- Archive files are limited to 2 gigabytes in size and/or may contain no more than 65,535 entries (use of multiple archive files easily overcomes these restrictions).
- Serve from Zip does not support multiple-disk archive files.
- Be aware, Akamai's FTP Download product does not support the Serve from Zip feature.
- HTTP byte-range requests are not supported since doing so requires the actual unzipping of the archive file, which could impede performance (for example, if end users repeatedly asked for byte 299999 of a 300 megabyte file).
- Serve from Zip's indexing operation adds information to the archive file, so if CMS's **`md5sum`** (MD5 checksum) command (see Chapter 8) is run after indexing, the hash will differ from that of your local archive version. For the same reason, running FTP's **`site az2z`** command in tandem with its **`site chkhash [MD5Digest]`** command will produce an error.
- Serve from Zip can be problematic for NetStorage's **Send MD5 Sum** feature. When enabled, the feature sends an MD5 hash in the response header of each file served from NetStorage. In the case of Serve from Zip content, however, the MD5 hash is that of the archive file, not of the content file itself, so if the end user's client (including Akamai's Download Manager) checks the response header, the hash will not match that of the file received, and the file will be rejected.

  This behavior also prevents content caching at Akamai's Edge servers if you have asked Akamai to enable the Edge's hash-checking feature. In this case, it does not interfere with the actual serving of content to end users, but it does require Edge servers to retrieve content from NetStorage each time content is requested.

  In the future, Serve from Zip will override the **Send MD5 Sum** feature and NetStorage will not send an MD5 hash with Serve from Zip content.
- Capabilities beyond the basic UNIX system's unzip are not supported (bzip compression, encryption, and zip64 format, for example).
- Zip logic does not support emergency directories or stale content protection.

Using the Range Fetch Alternative

An alternative to Serve from Zip that may be preferred in some circumstances is serving content from a flat file using a range fetch. Obviously, this will not be helpful in situations where preserving a directory structure is required; Serve from Zip remains the best solution in those cases. If you have data that does not fit well into individual files, however, a flat file might work best. For example, if you have a large number of random one-line sentences you want to display (e.g., the UNIX system's fortune program), you could create a file for each and give them random names, or you could create one flat file and use range fetches to gather the individual sentences.