Gluster Filesystem
Unified File and Object Storage – Beta 1

Release: July 2011

Copyright

This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

Table of Contents

1. About this Guide

This guide describes Unified File and Object Storage feature and its deployment and management.

1.1. Disclaimer

Gluster, Inc. has designated English as the official language for all of its product documentation and other documentation, as well as all our customer communications. All documentation prepared or delivered by Gluster will be written, interpreted and applied in English, and English is the official and controlling language for all our documents, agreements, instruments, notices, disclosures and communications, in any form, electronic or otherwise (collectively, the "Gluster Documents").

Any customer, vendor, partner or other party who requires a translation of any of the Gluster Documents is responsible for preparing or obtaining such translation, including associated costs. However, regardless of any such translation, the English language version of any of the Gluster Documents prepared or delivered by Gluster shall control for any interpretation, enforcement, application or resolution.

1.2. Audience

This guide is intended for systems administrators interested in using Unified File and Object Storage feature of GlusterFS.

1.3. Prerequisite

This document assumes that you are familiar with the Linux operating system, concepts of File System, GlusterFS concepts, ReST web services, and HTTP/ 1.1.

1.4. Typographical Conventions

The following table lists the formatting conventions that are used in this guide to make it easier for you to recognize and use specific types of information.

Convention	Description	Example
`Courier` Text	Commands formatted as courier indicate shell commands.	`gluster volume start volname`
Italicized Text	Within a command, italicized text represents variables, which must be substituted with specific values.	`gluster volume start` ***volname***
Square Brackets	Within a command, optional parameters are shown in square brackets.	`gluster volume start volname [force]`
Curly Brackets	Within a command, alternative parameters are grouped within curly brackets and separated by the vertical OR bar.	gluster volume { start | stop | delete } volname

1.5. Feedback

Gluster welcomes your comments and suggestions on the quality and usefulness of its documentation. If you find any errors or have any other suggestions, write to us at docfeedback@gluster.com for clarification and provide the chapter, section, and page number, if available.

Gluster offers a range of resources related to Gluster software:

- Discuss technical problems and solutions on the Discussion Forum (http://community.gluster.org)
- Get hands-on step-by-step tutorials (http://www.gluster.com/community/documentation/index.php/Main_Page)
- Reach Support (http://www.gluster.com/services/)

2. Introducing Unified File and Object Storage

Gluster's Unified File and Object Storage unifies NAS and object storage technology. This provides a system for data storage that enables users to access the same data as an object and as a file, simplifying management and controlling storage costs.

GlusterFS with unified file and object storage technology, is key to enterprise customer's ability to adopt and deploy cloud storage solutions. GlusterFS allows users to access data as objects from an Amazon S3 compatible interface and access files from a NAS interface including NFS and CIFS. In addition to decreasing cost and making it faster and easier to access object data, GlusterFS also delivers massive scalability, high availability and replication of object storage. For infrastructure as a service offerings, GlusterFS enables organizations to build their own Amazon-like storage offering for its customers. Enterprises can use GlusterFS to accelerate the process of preparing file-based applications for the cloud and simplify new application development for cloud computing environments.

3. Preparing to Deploy Unified File and Object Storage

This section provides information on pre-requisites and list of dependencies that will be installed during installation of Unified File and Object Storage of GlusterFS.

3.1. Pre-requisites

Enable 'user_xattr' for the backend used for GlusterFS using the following command:

```
# mount -o user_xattr,rw <device name> <mount point>
```

For example,

```
# mount -o user_xattr,rw /dev/hda1 /data
```

3.2. Dependencies

The following packages will be installed in CentOS when you install Gluster Object Storage:

- curl
- gcc
- memcached
- python2.6
- xattr
- netifaces
- eventlet
- paste
- configobj
- coverage
- setuptools
- webob
- greenlet
- xfsprogs
- simplejson
- nose
- openssl

- pastedeploy

The following packages will be installed in CentOS when you install Gluster Object Storage:

- curl
- gcc
- memcached
- python-software-properties
- python-configobj
- python-coverage
- python-dev
- python-nose
- python-setuptools
- python-simplejson
- python-xattr xfsprogs
- python-webob
- python-eventlet
- python-greenlet
- python-pastedeploy
- python-netifaces

4. Deploying and Configuring Unified File and Object Storage

This section describes how to deploy and configure Gluster Unified File and Object Storage in your storage environment, and verify that it is functioning correctly.

1. Untar the glusterfs-3.3beta1 tarball on each server in your cluster. You can download the software at http://bits.gluster.com/pub/gluster/glusterfs/src/glusterfs-3.3beta1.tar.gz.

2. Run the configuration utility using the following command:

```
# ./configure
GlusterFS configure summary
==================
FUSE client : yes
Infiniband verbs : yes
epoll IO multiplex : yes
argp-standalone : no
fusermount : no
readline : yes
georeplication : yes
```

   The configuration summary shows the components that will be built with GlusterFS.

3. Build the GlusterFS software using the following commands:

```
# make
# make install
```

4. Verify that 3.3beta1 version of GlusterFS is installed, using the following command:

```
# glusterfs –version
```

   For more information on installing GlusterFS, refer to GlusterFS Installation at http://www.gluster.com/community/documentation/index.php/Gluster_3.2_Filesystem_Installation_Guide

5. Add FUSE module by using the following command:

```
# modprobe fuse
```

6. Untar the gluster-object-storage-3.3beta1 tarball to each server in your cluster. You can download the software at http://download.gluster.com/pub/gluster/glusterfs/qa-releases/3.3-beta/.

7. Navigate to the gluster-object-storage-3.3beta1 directory and the following install scripts will be available:

- `centos_install.sh`
- `ssa_install.sh`
- `ubuntu_install.sh`

8. Run the install script corresponding to your distribution to install gluster object storage and its dependencies.

9. Create volume 'auth' using the following command:

```
# gluster volume create NEW-VOLNAME NEW-BRICK...
```

For example,

```
# gluster volume create auth server1:/exp1
```

10. Start the volume 'auth' using the following command:

```
# gluster volume start VOLNAME
```

For example,

```
# gluster volume start auth
Starting auth volume has been successful
```

11. You can create more volumes or use existing volumes as object storage accounts.

```
# gluster volume create VOL-NAME NEW-BRICK...
```

For example,

```
# gluster volume create v1 server1:/data server2:/data
```

You can verify the existing accounts using `gluster-object-list` as described in Step 13.

12. Initialize the authentication using the following command:

```
# gluster-object-prep -K <super_admin_key> -A <http_url/https_url>
```

For example,

```
# gluster-object-prep -K Password@123 -A https://127.0.0.1:443/auth/
```

Note: All accounts except 'auth' are accessible by username = '.super_admin' and password = <super_admin_key>. You should generate user token on 'auth' server local to that account to access with '.super_admin'.

13. Add users to the account by using the following command:

```
# gluster-object-add-user -a -K <super_admin_key> -A <http_url/https_url>
<account_name> <user_name> <password> [<uid>] [<gid>]
```

For example,

```
# gluster-object-add-user -a -K pasword@124 -A
https://127.0.0.1:443/auth/ v1 john samplepassword 212 444
```

The UID (owner) and GID (group owner) will be the owner for objects/containers created by this user. If UID and GID is not specified, it will default to 'root'.

You can run `# gluster-object-add-user` for help.

14. Verify that the account is successfully created by using the following command:

    `# gluster-object-list -K <super_admin_key> -A <http_url/https_url>`

    For example,

    ```
    # gluster-object-list -K password@123 -A https://127.0.0.1:443/auth/
    ```

    Verify that the output lists all the accounts available in the storage pool.

15. (Optional) Delete the existing users and its tokens using the following command:

    ```
    # gluster-object-delete-user -K <super_admin_key> -A <http_url/https_url>
    <account_name> <user_name>
    ```

    For example,

    ```
    # gluster-object-delete-user -K pasword@124 -A
    https://127.0.0.1:443/auth/ v1 john
    ```

    You can run `# gluster-object-delete-user` for help.

16. (Optional) Set the service URL for the existing accounts using the following command:

    ```
    # gluster-object-set-account-service -K <super_admin_key> <account>
    <service> <name> <value>
    ```

    For example,

    ```
    # gluster-object-set-account-service -K pasword@124
    https://127.0.0.1:443/auth/ test storage local
    https://127.0.0.1:443/v1/auth_test
    ```

    You can run `# gluster-object-set-account-service` for help.

17. (Optional) Stop the service using the following command:

    ```
    # gluster-object-stop
    ```

18. Use cURL or any other application to access object storage.

5. Working with Unified File and Object Storage

This section describes the ReST API for administering and managing Object Storage. All requests will be directed to the host and URL described in the X-Storage-URL HTTP header obtained during successful authentication.

The ReST interface is a data service that allows you to construct HTTP requests to query data in GlusterFS.

5.1. Configuring Authenticated Access

Authentication is the process of proving identity to the system. To use the ReST interface, you must obtain an authorization token using GET method and supply it with v1.0 as the path.

Each ReST request against the Object Storage system requires the addition of a specific authorization token HTTP x-header, defined as X-Auth-Token. The storage URL and authentication token are returned in the headers of the response.

- To authenticate, use the following command:

```
GET /v1.0 HTTP/1.1
Host: <storage URL>
X-Auth-User: <user name>
X-Auth-Key: <authentication-token-key>
```

For example,

```
GET /auth/v1.0 HTTP/1.1
Host: auth.example.com
X-Auth-User: john
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89

HTTP/1.1 204 No Content
Date: Wed, 13 Jul 2011 15:32:21 GMT
Server: Apache
X-Storage-Url: https://example.storage.com/v1
X-Auth-Token: abbefd18-0fac-3b3a-72b4-321c44ec1cbb
Content-Length: 0
Content-Type: text/plain; charset=UTF-8
```

The X-Storage-Url has to be parsed and used in the connection and request line of all subsequent requests to the server. In the example output, users connecting to server will send most container/object requests with a host header of example.storage.com and the request line's version and account as /v1.

Note: The authentication tokens are valid for a 24 hour period.

5.2. Working with Accounts

This section describes the list of operations you can perform at the account level of the URL.

5.2.1. Displaying Container Information

You can list the objects of a specific container, or all containers, as needed using GET command. You can use the following optional parameters with GET request to refine the results:

Parameter	Description
limit	Limits the number of results to at most n value.
marker	Returns object names greater in value than the specified marker.
format	Specify either json or xml to return the respective serialized response.

To display container information

- List objects of a specific container using the following command:

```
GET / <account>/<container> HTTP/1.1
Host: <storage URL>
X-Auth-Key: <authentication-token-key>
```

For example,

```
GET / v1 photos HTTP/1.1
Host: auth.example.com
X-Auth-User: john
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89

HTTP/1.1 200 Ok
Date: Wed, 13 Jul 2011 15:32:21 GMT
Server: Apache
Content-Type: text/plain; charset=UTF-8
Content-Length: 139

sample file.jpg
test-file.pdf
You and Me.pdf
Puddle of Mudd.mp3
Test Reports.doc
```

- List all the containers of an account using the following command:

```
GET / <account> HTTP/1.1
Host: <storage URL>
X-Auth-Key: <authentication-token-key>
```

For example,

-
```
GET / v1 HTTP/1.1
Host: auth.example.com
X-Auth-User: john
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89


HTTP/1.1 200 Ok
```

```
Date: Wed, 13 Jul 2011 16:32:21 GMT
Server: Apache
Content-Type: text/plain; charset=UTF-8
Content-Length: 39

songs
movies
documents
reports
```

5.2.2. Displaying Account Metadata Information

You can issue HEAD command to the storage service to view the number of containers and the total bytes stored in the account.

- To display containers and storage used, use the following command:

```
HEAD <account> HTTP/1.1
Host: <storage URL>
X-Auth-Key: <authentication-token-key>
```

For example,

```
GET / v1 HTTP/1.1
Host: auth.example.com
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89

HTTP/1.1 204 No Content
Date: Wed, 13 Jul 2011 16:52:21 GMT
Server: Apache
X-Account-Container-Count: 4
X-Account-Total-Bytes-Used: 394792
```

5.3. Working with Containers

This section describes the list of operations you can perform at the container level of the URL.

5.3.1. Creating Container

You can use PUT command to create containers. Containers are the storage folders for your data. The URL encoded name must be less than 256 bytes and cannot contain a forward slash '/' character.

- To create a container, use the following command:

```
PUT / <account>/<container>/HTTP/1.1
Host: <storage URL>
X-Auth-Key: <authentication-token-key>
```

For example,

```
PUT / v1/pictures/HTTP/1.1
Host: auth.example.com
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89
```

```
HTTP/1.1 201 Created
Date: Wed, 13 Jul 2011 17:32:21 GMT
Server: Apache
Content-Type: text/plain; charset=UTF-8
```

The status code of 201 (Created) indicates that you have successfully created the container. If a container with same is already existed, the status code of 202 is displayed.

5.3.2. Displaying Objects of a Container

You can list the objects of a container using GET command. You can use the following optional parameters with GET request to refine the results:

Parameter	Description
limit	Limits the number of results to at most n value.
marker	Returns object names greater in value than the specified marker.
prefix	Displays the results limited to object names beginning with the substring x. beginning with the substring x.
path	Returns the object names nested in the pseudo path.
format	Specify either json or xml to return the respective serialized response.
delimiter	Returns all the object names nested in the container

To display container information

- List objects of a specific container using the following command:

```
GET / <account>/<container>[parm=value] HTTP/1.1
Host: <storage URL>
X-Auth-Key: <authentication-token-key>
```

For example,

```
GET / v1/images HTTP/1.1
Host: auth.example.com
X-Auth-User: john
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89

HTTP/1.1 200 Ok
Date: Wed, 13 Jul 2011 15:42:21 GMT
Server: Apache
Content-Type: text/plain; charset=UTF-8
Content-Length: 139
sample file.jpg
test-file.pdf
You and Me.pdf
Puddle of Mudd.mp3
Test Reports.doc
```

5.3.3. Displaying Container Metadata Information

You can issue HEAD command to the storage service to view the number of objects in a container and the total bytes of all the objects stored in the container.

- To display list of objects and storage used, use the following command:

```
HEAD <account>/<container> HTTP/1.1
Host: <storage URL>
X-Auth-Key: <authentication-token-key>
```

For example,

```
GET / v1/images HTTP/1.1
Host: auth.example.com
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89

HTTP/1.1 204 No Content
Date: Wed, 13 Jul 2011 19:52:21 GMT
Server: Apache
X-Account-Object-Count: 8
X-Container-Bytes-Used: 472
```

5.3.4. Deleting Container

You can use DELETE command to permanently delete containers. The container must be empty before it can be deleted.

You can issue HEAD command to determine if it contains any objects.

- To delete a container, use the following command:

```
DELETE / <account>/<container>/HTTP/1.1
Host: <storage URL>
X-Auth-Key: <authentication-token-key>
```

For example,

```
DELETE / v1/pictures HTTP/1.1
Host: auth.example.com
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89

HTTP/1.1 204 No Content
Date: Wed, 13 Jul 2011 17:52:21 GMT
Server: Apache
Content-Length: 0
Content-Type: text/plain; charset=UTF-8
```

The status code of 204 (No Content) indicates that you have successfully deleted the container. If that container does not exist, the status code 404 (Not Found) is displayed, and if the container is not empty, the status code 409 (Conflict) is displayed.

5.4. Working with Objects

An object represents the data and any metadata for the files stored in the system. Through the ReST interface, metadata for an object can be included by adding custom HTTP headers to the request and the data payload as the request body. Objects name should not exceed 1024 bytes after URL encoding.

This section describes the list of operations you can perform at the object level of the URL.

5.4.1. Creating or Updating Object

You can use PUT command to write or update an object's content and metadata.

You can verify the data integrity by including an MD5checksum for the object's data in the ETag header. ETag header is optional and can be used to ensure that the object's contents are stored successfully in the storage system.

You can assign custom metadata to objects by including additional HTTP headers on the PUT request. The objects created with custom metadata via HTTP headers are identified with the `X-Object-Meta-` prefix.

- To create or update an object, use the following command:

```
PUT / <account>/<container>/<object> HTTP/1.1
Host: <storage URL>
X-Auth-Key: <authentication-token-key>
ETag: da1e100dc9e7becc810986e37875ae38
Content-Length: 342909
X-Object-Meta-PIN: 2343
```

For example,

```
GET / v1/pictures/dog HTTP/1.1
Host: auth.example.com
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89
ETag: da1e100dc9e7becc810986e37875ae38


HTTP/1.1 201 Created
Date: Wed, 13 Jul 2011 18:32:21 GMT
Server: Apache
ETag: da1e100dc9e7becc810986e37875ae38
Content-Length: 0
Content-Type: text/plain; charset=UTF-8
```

The status code of 201 (Created) indicates that you have successfully created or updated the object. If there is a missing content-Length or Content-Type header in the request, the status code of 412 (Length Required) is displayed. (Optionally) If the MD5 checksum of the data written to the storage system does not match the ETag value, the status code of 422 (Unprocessable Entity) is displayed.

5.4.1.1. Chunked Transfer Encoding

You can upload data without knowing the size of the data to be uploaded. You can do this by specifying an HTTP header of Transfer-Encoding: chunked and without using a Content-Length header.

You can use this feature while doing a DB dump, piping the output through gzip, and then piping the data directly into Object Storage without having to buffer the data to disk to compute the file size.

- To create or update an object, use the following command:

```
PUT / <account>/<container>/<object> HTTP/1.1
Host: <storage URL>
X-Auth-Key: <authentication-token-key>
Transfer-Encoding: chunked
X-Object-Meta-PIN: 2343
```

For example,

```
GET / v1/pictures/cat HTTP/1.1
Host: auth.example.com
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89
Transfer-Encoding: chunked
X-Object-Meta-PIN: 2343

19
A bunch of data broken up
D
into chunks.
0
```

5.4.2. Copying Object

You can copy object from one container to another or add a new object and then add reference to designate the source of the data from another container.

To copy object from one container to another

- To add a new object and designate the source of the data from another container, use the following command:

```
COPY /<account>/<container>/<sourceobject> HTTP/1.1
Host: <storage URL>
X-Auth-Token: < authentication-token-key>
Destination: /<container>/<destinationobject>
```

For example,

```
COPY /v1/images/dogs HTTP/1.1
Host: auth.example.com
X-Auth-Token: a87620deb2742ee3cd41341e26ab2e89
Destination: /photos/cats

HTTP/1.1 201 Created
Date: Wed, 13 Jul 2011 18:32:21 GMT
Server: Apache
Content-Length: 0
Content-Type: text/plain; charset=UTF-8
```

The status code of 201 (Created) indicates that you have successfully copied the object. If there is a missing content-Length or Content-Type header in the request, the status code of 412 (Length Required) is displayed.

- To add a new object and designate the source of the data from another container, use the following command:

```
COPY /<account>/<container>/<sourceobject> HTTP/1.1
Host: <storage URL>
X-Auth-Token: < authentication-token-key>
Destination: /<container>/<destinationobject>
```

For example,

```
COPY /v1/images/dogs HTTP/1.1
Host: auth.example.com
X-Auth-Token: a87620deb2742ee3cd41341e26ab2e89
Destination: /photos/cats

HTTP/1.1 201 Created
Date: Wed, 13 Jul 2011 18:32:21 GMT
Server: Apache
Content-Length: 0
Content-Type: text/plain; charset=UTF-8
```

The status code of 201 (Created) indicates that you have successfully copied the object. If there is a missing content-Length or Content-Type header in the request, the status code of 412 (Length Required) is displayed.

5.4.3. Displaying Object Information

You can issue GET command on an object to view the object data of the object.

- To display the content of object use the following command:

```
GET / <account>/<container>/<object>  HTTP/1.1
Host: <storage URL>
X-Auth-Key: <Authentication-token-key>
```

For example,

```
GET / v1/images/cat HTTP/1.1
Host: auth.example.com
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89

HTTP/1.1 200 Ok
Date: Wed, 13 Jul 2011 23:52:21 GMT
Server: Apache
Last-Modified: Thu, 14 Jul 2011 13:40:18 GMT
ETag: 8a964ee2a5e88be344f36c22562a6486
Content-Length: 534210
[.....]
```

The status code of 200 (Ok) indicates the object's data is displayed successfully. If that object does not exist, the status code 404 (Not Found) is displayed.

5.4.4. Displaying Object Metadata

You can issue HEAD command on an object to view the object metadata and other standard HTTP headers. You must send only authorization token as header.

- To display the metadata of the object, use the following command:

```
HEAD / <account>/<container>/<object> HTTP/1.1
Host: <storage URL>
X-Auth-Key: <Authentication-token-key>
```

For example,

```
HEAD / v1/images/cat HTTP/1.1
Host: auth.example.com
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89

HTTP/1.1 204 No Content
Date: Wed, 13 Jul 2011 21:52:21 GMT
Server: Apache
Last-Modified: Thu, 14 Jul 2011 13:40:18 GMT
ETag: 8a964ee2a5e88be344f36c22562a6486
Content-Length: 512000
Content-Type: text/plain; charset=UTF-8
X-Object-Meta-House: Cat
X-Object-Meta-Zoo: Cat
X-Object-Meta-Home: Cat
X-Object-Meta-Park: Cat
```

The status code of 204 (No Content) indicates the object's metadata is displayed successfully. If that object does not exist, the status code 404 (Not Found) is displayed.

5.4.5. Updating Object Metadata

You can issue POST command on an object name only to set or overwrite arbitrary key metadata. You cannot change the object's other headers such as Content-Type, ETag and others using POST operation. The POST command will delete all the existing metadata and replace it with the new arbitrary key metadata.

You must prefix X-Object-Meta- to the key names.

- To update the metadata of an object, use the following command:

```
POST / <account>/<container>/<object> HTTP/1.1
Host: <storage URL>
X-Auth-Key: <Authentication-token-key>
X-Object-Meta-<key>: <new value>
X-Object-Meta-<key>: <new value>
```

For example,

```
POST / v1/images/cat HTTP/1.1
Host: auth.example.com
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89
X-Object-Meta-Zoo: Lion
X-Object-Meta-Home: Dog
```

```
HTTP/1.1 202 Accepted
Date: Wed, 13 Jul 2011 22:52:21 GMT
Server: Apache
Content-Length: 0
Content-Type: text/plain; charset=UTF-8
```

The status code of 202 (Accepted) indicates that you have successfully updated the object's metadata. If that object does not exist, the status code 404 (Not Found) is displayed.

5.4.6. Deleting Object

You can use DELETE command to permanently delete the object.

The DELETE command on an object will be processed immediately and any subsequent operations like GET, HEAD, POST, or DELETE on the object will display 404 (Not Found) error.

- To delete an object, use the following command:

```
DELETE / <account>/<container>/<object> HTTP/1.1
Host: <storage URL>
X-Auth-Key: <Authentication-token-key>
```

For example,

```
DELETE / v1/pictures/cat HTTP/1.1
Host: auth.example.com
X-Auth-Key: a87620deb2742ee3cd41341e26ab2e89

HTTP/1.1 204 No Content
Date: Wed, 13 Jul 2011 20:52:21 GMT
Server: Apache
Content-Type: text/plain; charset=UTF-8
```

The status code of 204 (No Content) indicates that you have successfully deleted the object. If that object does not exist, the status code 404 (Not Found) is displayed.

6. Managing Your Gluster Filesystem

The GlusterFS Administration Guide is available at:
http://www.gluster.com/community/documentation/index.php/Gluster_3.2_Filesystem_Administration_Guide.