Table of Contents

Hedvig Object Storage Overview .................................................................................................... 4
Object Storage Supported Features ................................................................................................ 4
Object Storage Topology .................................................................................................................. 5
Configuring Hedvig Object Storage ................................................................................................ 6
   Hedvig Configuration File (config.xml) ....................................................................................... 6
      Bucket Properties <bucket_properties> Element .................................................................... 7
      S3 <s3> Element ......................................................................................................................... 8
   Setting Up an HTTPS-supported S3 Service .................................................................................. 9
Hedvig Daily Log Files .................................................................................................................... 10
Hedvig Daily Activity Log ................................................................................................................ 10
Viewing Object Storage Metrics in the Hedvig WebUI ................................................................... 10
Setting Up S3 Service Users ......................................................................................................... 12
   Setting Up Users in the Hedvig WebUI ....................................................................................... 12
   Setting Up Users in the Hedvig REST API .................................................................................. 13
Tenant-based Quota Management .............................................................................................. 14
   Configuring a Tenant’s Upper Quota Limit ............................................................................... 14
   Bucket Quota Management ......................................................................................................... 15
Setting Up Security Tokens .......................................................................................................... 16
   Security Tokens as AWS Secret Access Keys .......................................................................... 16
   Generating Security Tokens using the Hedvig WebUI ............................................................... 17
   Generating Security Tokens using the Hedvig CLI ................................................................. 18
   Generating Security Tokens using the Hedvig REST API ....................................................... 19
Connecting to the Hedvig S3 Endpoint ........................................................................................ 20
   Using HTTPS with the AWS CLI .............................................................................................. 20
   Setting S3 Server-Side Encryption (SSE) .................................................................................. 21
Specifying Hedvig Attributes during Bucket Creation ................................................................. 22
Request Access Styles ................................................................................................................... 25
   Determining Path Style vs. Virtual-Hosted Style Requests ..................................................... 25
Access Control .............................................................................................................................. 26
   S3 Bucket Policy Support ........................................................................................................... 26
      S3 Bucket Policy Processing .................................................................................................. 26
   S3 Bucket and Object Action Support ....................................................................................... 29
   Bucket Policy Examples ............................................................................................................ 30
Website Hosting ............................................................................................................................ 31
   Website Redirection .................................................................................................................... 31
   Applying the CORS Configuration to Buckets ......................................................................... 31
Bucket and Object Naming Restrictions ................................................................. 33
Hedvig-supported S3 API Calls ............................................................................ 34
Hedvig-supported AWS CLI Reference ................................................................. 38
Operations on Buckets.......................................................................................... 39
  put-bucket-acl ........................................................................................................ 39
  get-bucket-acl ........................................................................................................ 40
  put-bucket-cors ....................................................................................................... 40
  get-bucket-cors ....................................................................................................... 40
  delete-bucket-cors ................................................................................................. 41
  put-bucket-policy ................................................................................................. 41
  get-bucket-policy ................................................................................................. 41
  delete-bucket-policy ............................................................................................. 42
  put-bucket-tagging .............................................................................................. 42
  get-bucket-tagging .............................................................................................. 42
  delete-bucket-tagging ......................................................................................... 43
  put-bucket-versioning ......................................................................................... 43
  get-bucket-versioning ......................................................................................... 43
  create-bucket ....................................................................................................... 44
  delete-bucket ........................................................................................................ 44
  head-bucket ........................................................................................................... 45
  list-buckets ........................................................................................................... 45
  list-objects or list-objects-v2 .............................................................................. 46
  list-object-versions ............................................................................................. 46
Operations on Objects ........................................................................................... 47
  put-object ............................................................................................................ 48
  get-object ............................................................................................................. 49
  delete-object ....................................................................................................... 49
  delete-objects ..................................................................................................... 50
  put-object-acl ...................................................................................................... 50
  get-object-acl ...................................................................................................... 51
  put-object-tagging .............................................................................................. 51
  get-object-tagging .............................................................................................. 52
  delete-object-tagging ......................................................................................... 52
  copy-object .......................................................................................................... 53
  head-object ........................................................................................................... 54
Multipart Operations ............................................................................................. 54
  abort-multipart-upload ................................................................................... 55
  complete-multipart-upload ................................................................................ 55
  create-multipart-upload .................................................................................... 56
  list-multipart-uploads ......................................................................................... 56
  list-parts ............................................................................................................... 57
  upload-part .......................................................................................................... 57
  upload-part-copy ................................................................................................. 58
Glossary .................................................................................................................. 59
Hedvig Object Storage Overview

Hedvig supports an S3 protocol-compatible object storage implementation, which can be managed in many ways, for example:

- Open-source user interface applications, such as Cyberduck and S3 Browser, are compatible with Hedvig Object Storage. *(Hedvig does not currently have an object storage UI.)*
- New S3 applications are developed by leveraging the open-source AWS SDKs, which are available in different languages for application developers.

Object Storage Supported Features

- Access over HTTP and HTTPS
- S3 versions 2 and 4 authentication
- ACLs (access control lists), for bucket and object granularity, and access control, using the bucket policy
- Hierarchical object namespaces
- Object versioning
- Large objects through multipart uploads
- Compression at the bucket level
- Encryption at the object level
- Tagging and customized metadata (at both the bucket and object level)
- Efficient reading (parts and whole), writing, and listing of object data and metadata, and object versions in a bucket
- Deletion of multiple objects in one request, reducing per request overhead
- Dual data center replication policy environments
- Data protection policies:
  - Erasure Coding (using Reed-Solomon Erasure Codes)
  - Replication
- Static website hosting
- Website redirection
- Tenant-based quota management
Object Storage Topology

A Hedvig Storage Cluster deployed to run object storage has the flexibility only to run a Hedvig Storage Proxy as a service on the cluster storage node.

**Note:** The load balancer shown in *Figure 1* is not mandatory.

*Figure 1: Hedvig Object Storage topology*
Configuring Hedvig Object Storage

- **Hedvig Configuration File (config.xml)**
- **Setting Up an HTTPS-supported S3 Service**
- **Hedvig Daily Log Files**
- **Hedvig Daily Activity Log**
- **Viewing Object Storage Metrics in the Hedvig WebUI**

**Hedvig Configuration File (config.xml)**

Hedvig Storage Cluster parameters are identified in a configuration file (config.xml), located in the /var/log/hedvig/ directory of each Hedvig Storage Proxy. The `<objectstore>` element contains the parameters for configuring object storage on a storage proxy.

**Note:** Any time you change the Hedvig configuration file, you must restart the Hedvig service for these changes to take effect.

```xml
<objectstore>
  <bucket_properties>
    <bucket_blk_size_in_kb>4</bucket_blk_size_in_kb>
    <bucket_rf>3</bucket_rf>
    <bucket_replication_policy>Agnostic</bucket_replication_policy>
    <bucket_versioning>unversioned</bucket_versioning>
    <bucket_compressed>true</bucket_compressed>
    <bucket_protection_policy>ERASURE</bucket_protection_policy>
    <bucket_erasure_config>RS_4_2</bucket_erasure_config>
    <bucket_disk_residence>HDD</bucket_disk_residence>
  </bucket_properties>
  <s3>
    <metaclients>4</metaclients>
    <readclients>64</readclients>
    <writeclients>64</writeclients>
    <port>16000</port>
    <certfile>/var/log/hedvig/hedvigs3.crt</certfile>
    <keyfile>/var/log/hedvig/hedvigs3.key</keyfile>
    <rsakeyfile>/var/log/hedvig/privateKey</rsakeyfile>
    <https>false</https>
    <hostendpoints>nginxvm.hedviginc.com</hostendpoints>
  </s3>
</objectstore>
```

The `<objectstore>` element has two sections:

- **Bucket properties `<bucket_properties>` element**
- **S3 `<s3>` element**
Bucket Properties <bucket_properties> Element

The following properties are *global* bucket properties and can be overridden by bucket *location* constraints when creating a bucket.

**Note:** After bucket creation is complete, none of these properties can be modified.

- **bucket_blk_size_in_kb** [values: 4, 64 (default)] — the default bucket block size for all buckets created through this storage proxy. It is recommended that you use a default of 64k. For buckets using Erasure Coding, only 64k is supported.

- **bucket_rf** [values: 1 to 6 (default)] — the default bucket replication factor for all buckets created through this storage proxy.

- **bucket_replication_policy** [values: Agnostic, RackAware, DataCenterAware] — the default bucket replication policy for all buckets created through this storage proxy.

- **bucket_versioning** [values: unversioned, versioningenabled] — the default bucket versioning status for all buckets created through this storage proxy.

- **bucket_compressed** [values: true, false] — the default instruction for whether a bucket is to be compressed for all buckets created through this storage proxy.

- **bucket_protection_policy** [values: MIRROR, ERASURE] — the default data protection policy for all buckets created through this storage proxy.

- **bucket_erasure_config** [values: RS_4_2, RS_8_2, RS_8_3, RS_8_4 (N/A for MIRROR)] — the default Erasure Coding configuration for all buckets created through this storage proxy. (See also the section on Erasure Coding in *Table 1: Hedvig attribute keys and valid values.*)

- **bucket_disk_residence** [values: HDD, Flash] — the default bucket residence for all buckets created through this storage proxy.

- **bucket_quota_in_gb** — an optional tag that can be specified to associate a bucket quota size with the bucket upon creation through this storage proxy. By default, this value is set to zero. Note that if bucket quota exceeds the tenant quota, an insufficient space error may be returned.
S3 <s3> Element

- `<metaclients>` – the number of S3 Service threads that will be processing requests other than IO (all other lightweight requests) within each S3 Service. *Contact Hedvig for help with setting this value.*

- `<readclients>` – the number of S3 Service threads that will be processing Read IO requests (read/download object) within each S3 Service. *Contact Hedvig for help with setting this value.*

- `<writeclients>` – the number of S3 Service threads that process Write IO requests (create object) within each S3 Service. *Contact Hedvig for help with setting this value.*

- `<port>` [default: 16000] – the port on which a Hedvig S3 Storage Proxy is to listen for incoming S3 Service client requests.

- `<certfile>` and `<keyfile>` – the paths to the certification and key files, which are necessary only if `<https>` is set to `true`. Otherwise, these elements are not required.

- `<rsakeyfile>` – the path to the RSA private key file for decryption on the Hedvig Storage Proxy, which must be the same file that is present on the Hedvig Storage Cluster Nodes under `/usr/local/hedvig/pages/privateKey`.

**IMPORTANT**: To start an S3 Service on a Hedvig Storage Proxy, you must copy the RSA private key file to `/var/log/hedvig/privateKey` and set the restrictions properly.

- `<https>` [values: true, false] – enables or disables an HTTPS-supported S3 Service.

- `<hostendpoints>` – If the Hedvig Storage Cluster setup uses a load balancer with the storage proxies, then the load balancer hostname endpoints must be included in the storage proxy configuration file. This information identifies path vs. virtual-hosted style requests made through the load balancer and extracts the bucket name from the host header for virtual-hosted style requests. Use comma-separated load balancer fully qualified hostnames, such as `nginxvm1.hedviginc.com,nginxvm2.hedviginc.com`. 
Setting Up an HTTPS-supported S3 Service

To enable an HTTPS-supported S3 Service, follow these steps:

**Note:** If you have existing key and certificate files, you can skip step 1.

1. Generate three files: Key, Certificate Signing Request (CSR), and Digital Certificate (CRT).
   
   **Key:** `openssl genrsa -out ~/hedvigs3.key 2048`
   
   **CSR:** `openssl req -new -sha256 -key ~/hedvigs3.key -out ~/hedvigs3.csr`
   
   **Important:** Put the storage proxy hostname or * to match all hostnames in this field, for example:
   
   Common Name (your name or your server's hostname): s3proxyhost.hedviginc.com
   
   **CRT:** `openssl x509 -req -days 365 -in ~/hedvigs3.csr -signkey ~/hedvigs3.key -out hedvigs3.crt`

2. Deploy the Hedvig Storage Proxy.
   
   a. Copy both the `hedvigs3.crt` and `hedvigs3.key` files to the `/var/log/hedvig` directory on the Hedvig Storage Proxy.
   
   b. Update the `<objectstore><s3>` nested tag section of the Hedvig Storage Proxy configuration file (`/var/log/hedvig/config.xml`), as follows:
      
      i. `<certfile>` — Insert the path to the certificate file.
      ii. `<keyfile>` — Insert the path to the key file.
      iii. `<https>` — Set to true to enable HTTPS.

```
<objectstore>
  <bucket_properties>
    <bucket_blk_size_in_kb>64</bucket_blk_size_in_kb>
  </bucket_properties>
  <s3>
    <metaclients>4</metaclients>
    <writeclients>64</writeclients>
    <readclients>64</readclients>
    <port>16000</port>
    <certfile>/var/log/hedvig/hedvigs3.crt</certfile>
    <keyfile>/var/log/hedvig/hedvigs3.key</keyfile>
    <rsakeyfile>/var/log/hedvig/privateKey</rsakeyfile>
    <https>true</https>
    <accesslog>False</accesslog>
  </s3>
</objectstore>
```

*Figure 2: Sample `<objectstore>` element in Hedvig config.xml file*

3. Restart the S3 Service.

   `service hedvigs3 restart`
Hedvig Daily Log Files

Hedvig log files are archived daily on each storage proxy and are located in the
/var/log/hedvig/logs/hos directory.

Hedvig Daily Activity Log

Daily activity logs are generated as a CSV-formatted file, named access.csv, in the
/var/log/hedvig/logs/ARCHIVE directory under the respective date.

The logs list all activities and provide details for each storage proxy, including:

- the HTTP verb
- the user who issued each request
- the response code and status of the request

Viewing Object Storage Metrics in the Hedvig WebUI

You can view read and write metrics, on a per storage proxy level, in the Hedvig WebUI.

1. Login to the Hedvig WebUI with the appropriate credentials (Hedvig or LDAP user).
2. Click on the Cluster Watch tab to see the cluster layout.
3. Click on the HSP box.

   A list of available Storage Proxies is displayed. Hedvig S3 Storage Proxies are identified with a bucket icon ( ).

![Cluster Watch showing available Storage Proxies](image-url)
4. Click on the desired storage proxy to open the Metrics page.

![Metrics page showing Hedvig S3 Storage Proxy metrics](image)

The Metrics page lets you view read/write IOPS, min/max latencies, and throughput for every bucket that has been accessed through that storage proxy.

The left side of the Metrics page shows available metrics options. It is also displayed by clicking the ellipsis at the top right corner of an individual bucket graph. You can toggle these options to see the desired metrics on the graph.

To interpret the metrics displayed in Figure 4: Metrics page showing Hedvig S3 Storage Proxy metrics:

- One IO corresponds to one PUT or GET object.
- All calculations are made based on the size and latency of PUT or GET IOs that occurred during that sampling interval.
- The latency is calculated as the time taken for one PUT or GET IO to complete.
Setting Up S3 Service Users

For S3 purposes, each Hedvig user is treated as an independent user. Permissions and access control are done through S3 APIs and the AWS CLI. (Groups for Hedvig S3 users are not yet supported.)

Buckets that are visible to Hedvig users depend on the tenancy and the user’s role, as set in the Hedvig WebUI. RBAC applies only to this point with respect to S3, as everything is controlled in the granularity of users for S3 in Hedvig.

- Setting Up Users in the Hedvig WebUI
- Setting Up Users in the Hedvig REST API

For more information about tenants and user roles, see the Hedvig RBAC User Guide.

IMPORTANT: To use the AWS CLI, you must add the appropriate Hedvig users and passwords to the AWS credentials file. See also Security Token as AWS Secret Access Key.

Setting Up Users in the Hedvig WebUI

To set up a user within the Hedvig WebUI:

1. Login to the Hedvig WebUI with the appropriate credentials (Hedvig or LDAP user).
2. Select the User icon ( ) at the top right, and in the pulldown menu, select User Administration -> User Management.
3. In the User Management dialog, select Add User.
4. In the Add User dialog, as shown in Figure 5: Hedvig WebUI Add User dialog, create users, as needed, and select Run.

Note: For more information about the Hedvig WebUI, see the Hedvig User Guide.
Setting Up Users in the Hedvig REST API

To set up a user within the Hedvig REST API:

curl -X PUT hedvig1.hedviginc.com/rest/ --data-urlencode
"request={type:AddUser, category:UserManagement,
params:{displayName:'hedviguser', userName:'hedviguser', tenant:'Hedvig',
role:'SuperUser', email:'hedviguser@hedviginc.com', mobile:'1234567890',
password:'hedvig', isGeneratedPassword:'false', isLdapUser:'false'},
sessionId:'d93ca275a6e4dd1e9ec10696e923aded'}}" | python -m json.tool

Here is an example of a returned result:

```
{
    "requestId": "CFC8BBA3-BB06-12D9-771A-E5F830BE72B2",
    "result": {
        "isLdapUser": false,
        "tenant": "Hedvig",
        "userName": "hedviguser"
    },
    "status": "ok",
    "type": "AddUser"
}
```

**Note:** For more information about the Hedvig REST API, see the *Hedvig REST API Guide.*
Tenant-based Quota Management

In object storage, each bucket is pinned to one tenant. That tenant is specified during bucket creation using the LocationConstraint param, x-amz-meta-hedvig-bucket-tenant, which is described in Specifying Hedvig Attributes during Bucket Creation.

This section has the following subsections:

- Configuring a Tenant's Upper Quota Limit
- Bucket Quota Management

Important Notes:

- Object storage buckets cannot share the same tenancy as that of virtual disks with differing protocols.
- For more information about user creation, tenant creation, and tenant quotas, see the Hedvig User Guide and the Hedvig RBAC User Guide.

Configuring a Tenant’s Upper Quota Limit

A tenant can be configured with an upper quota limit, that is, how much of a cluster that tenant can own.

1. In the Hedvig WebUI, at the upper right, click User Administration and select Tenant Management.

You can see the size (quota/maximum capacity) and the current capacity of each tenant.
2. Click the ... button to the left of each tenant name, and resize (increase only, no shrinking) the quota for this tenant.

![Figure 8: Changing quota for tenants](image)

As contrasted with other protocols, bucket usage can vary when an object is created and deleted. Current capacity is updated based on a bucket usage summary every 24 hours and can be adjusted for specific use cases.

### Bucket Quota Management

Based on whether a bucket is created with or without a predetermined quota size, there are two ways to manage the bucket’s quota.

- If a bucket is created *with* a predetermined quota size, then the size of the bucket is reflected immediately at creation in the tenant quota current capacity. Object creation in a bucket is allowed only until this quota threshold is reached. The bucket quota can be resized using the `resizebucketquota` command.

```
alias> resizebucketquota
usage: resizebucketquota -n <arg> -s <arg> -t <arg> -u <arg> -n,--bucketname <arg> Bucket Name
-s,--newsize <arg> New Bucket quota in GB
-t,--objectstoretype <arg> Object store type : S3 or OpenStack
-u,--accountid <arg> Account Id : username
alias> resizebucketquota -t S3 -u HedvigAdmin -s 10GB -n bucket1 Bucket resized successfully.
```

**Important:** The `resizebucketquota` command can only *increase* bucket quota, that is, it cannot shrink it.

- If a bucket is created *without* a predetermined quota size, then object creation consumes capacity directly from the tenant quota. As contrasted with other protocols, bucket usage can vary when an object is created and deleted. In this case, current capacity is updated based on a bucket usage summary every 24 hours, and the update frequency can be adjusted for specific use cases.
Setting Up Security Tokens

To communicate with Hedvig’s S3 Service, you must generate a security token.

- **Security Token as AWS Secret Access Key**

To generate this token, you can use one of three methods:

- **Generating Security Tokens using the Hedvig WebUI**
- **Generating Security Tokens using the Hedvig CLI**
- **Generating Security Tokens using the Hedvig REST API**

**Note:**

- Currently, the generated security token cannot be changed at will.
- The security token email can be sent to the user any number of times.
- Only the actual user, or the superuser, can send the security token email.

**Security Tokens as AWS Secret Access Keys**

A security token, which serves as the `aws_secret_access_key`, is either:

- specified in the AWS credentials file for the AWS CLI on the client
- supplied to your S3 application in the format required for that application

The AWS credentials file is found on the S3 client machine at `~/.aws/credentials`.

In the example in Figure 5: AWS credentials file with security token:

- The `aws_access_key_id` is the Hedvig user, for example, `hedvigdemo`.
- The `aws_secret_access_key` is the security token, for example, `49bd9f92-ab5f-4dc2-bc52-d403bd7c93a1`.

![Figure 9: AWS credentials file with security token](image-url)
Generating Security Tokens using the Hedvig WebUI

1. Login to the Hedvig WebUI with the appropriate credentials (Hedvig or LDAP user).

2. This step varies slightly, depending on whether you are logged in with Hedvig SuperUser privileges or as the actual user.

   • If you are logged in with Hedvig SuperUser privileges, select the User icon ( ) at the top right. In the pulldown menu, select User Administration -> User Management.

     ![User Management Menu](image1)

     **Figure 10: User Administration menu -> User Management**

     The list of users in the cluster is displayed, as shown in the User Management dialog.

     ![User Management Dialog](image2)

     **Figure 11: User Management dialog**

     Select the user for which you want to send the security token, which will open the user’s Edit Profile dialog.

     • If you are logged in as the actual user, select the User icon ( ) at the top right. In the pulldown menu, select User Administration -> Edit Profile. This will open the Edit Profile dialog.

     ![Edit Profile Menu](image3)

     **Figure 12: User Administration menu -> Edit Profile**
3. In the **Edit Profile** dialog, select the **Security Token Options** toggle.

![Security Token Options](image)

*Figure 13: Security Token Options button in Edit Profile dialog*

This displays the **Send Security Token** button, which when selected will send an email to the user with the required security token credential.

![Edit Profile](image)

*Figure 14: Edit Profile dialog*

### Generating Security Tokens using the Hedvig CLI

1. Login into the secured CLI using the respective user’s credentials for whom you want to get the security token. This can be done on any storage node.

2. Type:

   `scripts` **OR**
   
   `cd /usr/local/hedvig/scripts`

3. Type:

   `./secured-cli.sh`

4. Type:

   `getsecuritytokenforuser`
Generating Security Tokens using the Hedvig REST API

To log in using the Hedvig REST API:

```
curl -G hedvig1_sncl.hedviginc.com/rest/ --data-urlencode
"request={type:Login, category:UserManagement, params:{userName:'admin',
password:'admin23', cluster:'cluster.hedviginc.com'}}" | python -m json.tool
```

Here is an example of a returned result with the session ID:

```
{
    "requestId": "5E7D8FD7-8390-C47E-3AAC-519C51AC5F79",
    "result": {
        "datacenters": [ 
            {
                "dcname": "sncl"
            },
            {
                "dcname": "snc2"
            },
            {
                "dcname": "snc3"
            }
        ],
        "displayActivityLog": true,
        "displayName": "admin",
        "dualdc": true,
        "hcisetup": false,
        "primaryTenant": "Hedvig",
        "rackAware": true,
        "roles": { 
            "Hedvig": "SuperUser"
        },
        "sessionId": "05b1b99eb3385ce5c1dc95e07e2812b5",
        "userName": "admin",
    },
    "status": "ok",
    "type": "Login"
}
```

To send or get security tokens using the Hedvig REST API

```
curl -X PUT hedvig1.hedviginc.com/rest/ --data-urlencode
"request={type:SendSecurityToken, category:UserManagement,
params:{userName:'admin', tenant:'Hedvig',
sessionId:'05b1b99eb3385ce5c1dc95e07e2812b5'}}" | python -m json.tool
```

Here is an example of a returned result with the security token:

```
{
    "requestId": "49928DE9-8AAB-9A11-2572-BA6A0C443D9A",
    "securitytoken": "4cd8d047-04ad-4c95-b739-4494bf369ff7",
    "status": "ok",
    "type": "SendSecurityToken",
    "user": "admin"
}
```
Connecting to the Hedvig S3 Endpoint

**Note:** Because of virtual-hosted style access bucket name recognizing, do not use IP as an endpoint.

The `<Hedvig_endpoint>` for object storage is identified by: `://<hostname>:<port_number>`

- `http://<hostname>:<port_number>` or `https://<hostname>:<port_number>`
- `<hostname>` – the hostname of the Hedvig Storage Proxy that you have configured for object storage or the hostname of the load balancer with which you may be fronting the storage proxies.
- `<port_number>` – the `<port>` in the `config.xml` file or the port on which your HTTP load balancer is running in your environment.

**Note:** For more about the `config.xml` file, see *Hedvig Configuration File (config.xml)*.

The endpoint connection process depends on the method used for S3 Service communication:

- **AWS CLI** – Use the following template command:
  
  ```
  aws s3api --endpoint <Hedvig_endpoint> --profile <or default> <rest of the S3 command>
  ```

  **Note:** The command name and options are the same as those specified in the command list/help.

  The profile corresponds to the user profiles configured in the AWS credentials file.

- **S3 Browser, Cyberduck, or similar applications** – Specify the endpoint as S3 protocol-compatible, and provide the endpoint information as instructed.

- **AWS SDK-based applications** – The SDKs provide APIs to set the endpoint for an S3 protocol-compatible connection.

**Using HTTPS with the AWS CLI**

To send ssl requests from the AWS CLI using self-signed certificates:

1. Copy the `hedvigs3.crt` file (generated in *Setting up for HTTPS support*) to the S3 client where the AWS CLI is installed.

2. Use the following command line request template to send HTTPS requests:

   ```
   aws s3api --endpoint https://<hostname>:<port> --ca-bundle <certificate_file_path> <rest of the command>
   ```
Setting S3 Server-Side Encryption (SSE)

Amazon S3 provides two options for server-side encryption:

- **Amazon S3-Managed Encryption Keys (SSE-S3)**
  
  With SSE-S3, each object is encrypted with a unique key with AES-256 (in the Hedvig environment, a unique key per bucket, not per object).

- **AWS KMS-Managed Keys (SSE-KMS) (currently unsupported)**
  
  With SSE-KMS, the client specifies customer master keys (CMKs) for encrypting this object (CMK must be provided in the header).

For Hedvig-supported SSE-S3, keys are generated uniquely at the bucket level, but data encryption is at the object level. This means a bucket could have mixed encrypted and unencrypted objects present.

To set server-side encryption when putting an object:

- **Python CLI**

  ```
  aws s3api put-object --endpoint http://<hostname>:<port_number>
  --bucket <bucket_name> --key <object_name> --body <upload_file_path>
  --server-side-encryption AES256
  ```

- **Java SDK**

  ```
  ObjectMetadata objectMetadata = new ObjectMetadata();
  objectMetadata.setSSEAlgorithm(ObjectMetadata.AES_256_SERVER_SIDE_ENCRYPTION);
  PutObjectRequest putRequest = new PutObjectRequest(bucketName, 
  keyName, 
  new ByteArrayInputStream(objectContent.getBytes()), 
  objectMetadata);
  ```

For more information about S3 server-side encryption with Amazon S3-Managed Encryption Keys (SSE-S3), see:

Specifying Hedvig Attributes during Bucket Creation

Each bucket in Hedvig maps to a virtual disk. Hedvig-specific attributes, such as disk residence, replication policy, data protection policy, etc., can be associated with a bucket during creation. These attributes are passed on to the underlying virtual disk.

The attributes are specified as <attribute-Key>=<value> pairs, with each pair separated by a colon (:).

```
aws s3api create-bucket --endpoint <Hedvig_endpoint>
    --bucket <bucketName> --create-bucket-configuration
    "LocationConstraint=<<attribute-Key>=<value>>"
```

The following example creates a bucket with:

- block size of 4k
- protection policy of mirroring
- replication factor of 3
- replication policy of data center aware (with three data centers)
- data to reside on Flash

```
aws s3api create-bucket --endpoint <Hedvig_endpoint>
    --bucket <bucketName> --create-bucket-configuration
    "x-amz-meta-hedvig-protection-policy=MIRROR:
    x-amz-meta-hedvig-blocksize=4:
    x-amz-meta-hedvig-replication-factor=3:
    x-amz-meta-hedvig-bucket-disk-residence=Flash:
    x-amz-meta-hedvig-replication-policy-info=snc1,snc2,snc3:
    x-amz-meta-hedvig-replication-policy=DataCenterAware"
```

All attribute keys and their values are listed in Table 1.

**Important:** Most of these attributes cannot be changed after bucket creation. Exceptions are noted in the table.
<table>
<thead>
<tr>
<th>attribute key</th>
<th>valid values</th>
<th>notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-amz-meta-hedvig-protection-policy</td>
<td>MIRROR or ERASURE</td>
<td>Hedvig supports the following Reed-Solomon Erasure Code configurations: RS_4_2, RS_8_2, RS_8_3, and RS_8_4, which stands for:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RS_&lt;datafragments&gt;_&lt;parityfragments&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Erasure Coding has very specific requirements. A minimum of datafragments+parityfragments nodes, racks, or data centers for Agnostic, RackAware, or DataCenterAware policies, respectively, are required.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, for an RS_4_2 configuration, you need:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• a minimum of 6 nodes for an Agnostic policy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• a minimum of 6 racks for a RackAware policy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• a minimum of 6 data centers for a DataCenterAware policy</td>
</tr>
<tr>
<td>x-amz-meta-hedvig-erasure-config</td>
<td>RS_4_2, RS_8_2, RS_8_3, or RS_8_4</td>
<td>applicable if x-amz-meta-hedvig-protection-policy=ERASURE</td>
</tr>
<tr>
<td>x-amz-meta-hedvig-consistency-mode</td>
<td>WEAK or STRONG</td>
<td></td>
</tr>
<tr>
<td>x-amz-meta-hedvig-replication-factor</td>
<td>from 1 - 6</td>
<td>applicable if x-amz-meta-hedvig-protection-policy=MIRROR</td>
</tr>
<tr>
<td>x-amz-meta-hedvig-replication-policy</td>
<td>Agnostic, DataCenterAware, or RackAware</td>
<td></td>
</tr>
<tr>
<td>attribute key</td>
<td>valid values</td>
<td>notes</td>
</tr>
<tr>
<td>---------------------------------------------------</td>
<td>--------------------------------------------------</td>
<td>------------------------------------------------------------</td>
</tr>
<tr>
<td>x-amz-meta-hedvig-replication-policy-info</td>
<td>names of data centers separated by commas</td>
<td>applicable if x-amz-meta-hedvig-replication-policy=DataCenterAware</td>
</tr>
<tr>
<td>x-amz-meta-hedvig-blocksize</td>
<td>4 or 64 (in KB)</td>
<td>For Erasure Coding, only a block size of 64 (in KB) is supported.</td>
</tr>
<tr>
<td>x-amz-meta-hedvig-bucket-compressed</td>
<td>TRUE or FALSE</td>
<td></td>
</tr>
</tbody>
</table>
| x-amz-meta-hedvig-bucket-versioning              | unversioned or versioningenabled                 | After a bucket is created, the versioning status of the bucket can be changed, using the put-bucket-versioning command, but only according to these rules:  
  - unversioned -> versioned — yes, but only when bucket is empty.  
  - versioned <-> version_suspended — yes, it can be changed. |
| x-amz-meta-hedvig-bucket-disk-residence           | HDD or Flash                                     |                                                            |
| x-amz-meta-hedvig-bucket-tenant                  | a tenant to which current user belongs           | If not specified, then the user’s primary tenant is the default. |
| x-amz-meta-hedvig-bucket-quota                   | from 0 to the available capacity in the primary tenant (in GB) | Specifies the quota for this bucket. See Bucket Quota Management. |
Request Access Styles

Hedvig supports both S3 path-style and virtual-hosted style requests. Path-style requests are recommended, except when hosting static websites backed by Hedvig Object Storage buckets.

- A path-style request URL has the following format.

  http://hostendpoint/bucketname/keypath

  The `hostendpoint` is the fully qualified hostname of the storage proxy or load balancer endpoint and port number, for example, `hedvigs3sp.hedviginc.com:16000`.

  **Note:** For more information, see the `<hostendpoints>` definition in S3 `<s3>` Element.

- A virtual-hosted style request includes the bucket name as part of the domain name in the URL.

  http://bucketname.hostendpoint/key_name

  For virtual hosting of a bucket, you must create CNAME (canonical name) alias DNS (domain name system) records to redirect requests for `bucketname.hostendpoint` as the hostname to be routed to the appropriate host endpoint.

Determining Path Style vs. Virtual-Hosted Style Requests

Fully qualified hostnames must be used to send requests. If that is impossible, and only a portion of the hostname is used for the request, then the hostname must be added to the storage proxy configuration file [see Hedvig Configuration File (config.xml)] so that the request completes successfully.

This is how to determine the style request type and how the bucket name is fetched.

- If the host header field is not available, or if it matches the storage proxy hostname or hostnames specified in the storage proxy’s configuration file, then it is a path-style request.

- If the host header ends with `<hostendpoints>` (see also S3 `<s3>` Element), then the request is a virtual-hosted style request. The bucket name is the leading part of the host header field, up to the hostname, that is `bucketname.`.<hostname>.

- If the host header does not match any of the endpoints, then the value of the host header is considered the bucket name. For this to be successful, the CNAME record is set up so that the bucket name can be routed to the host.

  **Note:**

  - Access with the IP address:port number for specifying the endpoint is not supported.
  - The bucket name must be a DNS-compliant name.
Access Control

Access control is available by using the bucket policy and the access control list (ACL). If a bucket policy exists that matches the Resource, Principal, and Action elements for a specific operation, then the request is allowed or denied based on the Request element. Otherwise, the access is governed by the ACL associated with the resource.

For bucket operations, if a matching bucket policy exists, then that bucket policy takes precedence. For object operations, Hedvig supports only the s3:GetObject action exclusively through the bucket policy. For all other object operations, the bucket and object ACLs allow or deny access for a specific Principal.

- S3 Bucket Policy Support
- S3 Bucket and Object Action Support
- Bucket Policy Examples

S3 Bucket Policy Support

Hedvig has limited support for the S3 bucket policy. The bucket policy is configured by using GET, PUT, and DELETE bucket policy API operations on an existing bucket. Using this policy, a bucket owner can grant permission to other users to perform operations within the bucket when certain conditions are met.

The bucket policy definition loosely mirrors the S3 access control policy language — with some Hedvig-specific value definitions.

S3 Bucket Policy Processing

The S3 bucket policy is processed as follows.

For each request, if the Resource, Principal, and Action elements are satisfied, then a valid rule for this combination is assumed.

- If a rule exists and the effect is DENY, then conditions are not taken into consideration.
- If a rule exists and the effect is ALLOW, then conditions for the combination are evaluated and allowed or denied based on whether the Condition element is satisfied.
<table>
<thead>
<tr>
<th>element</th>
<th>description</th>
<th>values supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sid</td>
<td>Identifies the description or statement.</td>
<td>(This element is optional and is not substantially used by Hedvig.)</td>
</tr>
<tr>
<td>Effect</td>
<td>Determines if a rule is applied when the Principal, Resource, Action, and Condition elements are all satisfied.</td>
<td>ALLOW or DENY</td>
</tr>
</tbody>
</table>
| Principal | Identifies users who can perform, or users who cannot perform, specific actions. | To allow any user, including anonymous requests, specify the wildcards character as the value of the Principal element. Otherwise, the format is: 
\{ "HOSUID" : "uid1,uid2,uid3 ..."\} |
| Resource | Identifies the bucket and object path to which the specified rule statement applies. | To include all resources, specify the wildcards character as the value of the Resource element. Otherwise, the format is: 
[ "/bucketname", 
  
  
  
  "bucketname/objectpath" ...] 
The wildcards character can be used to denote all, and the wildcard character can be used for a single character. For example, the following value will match any object in a bucket: 
"/bucketname/*" |
<table>
<thead>
<tr>
<th>element</th>
<th>description</th>
<th>values supported</th>
</tr>
</thead>
</table>
| **Action** | Allows or denies the specified action the resource, based on the **Effect** element. (The **NotAction** element is **not** supported.) | To include all actions, specify the * wildcard character as the value of the **Action** element.  
To include only a specific set of actions, the format is:  
["s3:GetObject",  
  "s3:ListBucket" ...]  
See Hedvig-supported S3 actions.  
The * wildcard character can be used to denote a subset of operations. For example, the value:  
"s3:*Object"  
will include the following:  
s3:GetObject  
s3:PutObject  
s3:DeleteObject |
| **Condition** | Specifies a condition that, when satisfied, applies a rule.  
(The **IpAddress** and the **NotIpAddress** elements are both supported.) | The format for a condition is:  
Condition : { condition_type:  
  { condition_key:  
    condition_value } }  
The format for condition_key is **aws:SourceIp**.  
The condition_value can be specified as a single IP address or a comma-separated list of IP addresses in CIDR (classless interdomain routing) format. |
## S3 Bucket and Object Action Support

The following table lists all Hedvig-supported S3 bucket and object actions.

<table>
<thead>
<tr>
<th>bucket actions</th>
<th>object actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>s3:PutBucketAcl</td>
<td>s3:PutObject</td>
</tr>
<tr>
<td>s3:GetBucketAcl</td>
<td>s3:GetObject</td>
</tr>
<tr>
<td>s3:PutBucketCORS</td>
<td>s3:DeleteObject</td>
</tr>
<tr>
<td>s3:GetBucketCORS</td>
<td>s3:PutObjectAcl</td>
</tr>
<tr>
<td>s3:GetBucketLocation</td>
<td>s3:GetObjectAcl</td>
</tr>
<tr>
<td>s3:PutBucketPolicy</td>
<td>s3:PutObjectTagging</td>
</tr>
<tr>
<td>s3:GetBucketPolicy</td>
<td>s3:PutObjectVersionTagging</td>
</tr>
<tr>
<td>s3:DeleteBucketPolicy</td>
<td>s3:GetObjectTagging</td>
</tr>
<tr>
<td>s3:PutBucketTagging</td>
<td>s3:DeleteObjectTagging</td>
</tr>
<tr>
<td>s3:GetBucketTagging</td>
<td>s3:GetObjectVersionTagging</td>
</tr>
<tr>
<td>s3:DeleteBucketTagging</td>
<td>s3:DeleteObjectVersionTagging</td>
</tr>
<tr>
<td>s3:PutBucketVersioning</td>
<td>s3:DeleteObjectVersion</td>
</tr>
<tr>
<td>s3:GetBucketVersioning</td>
<td>s3:AbortMultipartUpload</td>
</tr>
<tr>
<td>s3:CreateBucket</td>
<td>s3:ListMultipartUploadParts</td>
</tr>
<tr>
<td>s3:DeleteBucket</td>
<td></td>
</tr>
<tr>
<td>s3:ListAllMyBuckets</td>
<td></td>
</tr>
<tr>
<td>s3:ListBucket</td>
<td></td>
</tr>
<tr>
<td>s3:ListBucket</td>
<td></td>
</tr>
<tr>
<td>s3:ListBucketMultipartUploads</td>
<td></td>
</tr>
<tr>
<td>s3:ListBucketVersions</td>
<td></td>
</tr>
</tbody>
</table>
Bucket Policy Examples

Here are some examples of the bucket policy with elements that Hedvig currently supports.

Bucket Policy Example 1

The following bucket policy allows all actions to be performed on the bucket and on all objects starting with the prefix `obj` in the bucket, by any user, but only when access is from the 172.22.67.168 or 172.22.61.79/24 subnet and not from 172.22.61.106.

```json
{
    "Id": "Policy1579910327757",
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "policyallaccessfromip",
            "Action": "*",
            "Effect": "Allow",
            "Resource": ["/ipbucket1", "/ipbucket1/obj*"],
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": ["172.22.67.168", "172.22.61.79/24"]
                }
            },
            "Principal": "*
        }
    ]
}
```

Figure 15: Bucket policy example 1

Bucket Policy Example 2

The following bucket policy allows public read access to a bucket. This policy can be used with virtual hosting to support running a static website using the bucket.

```json
[
    {
        "Id": "Policy1579910327757",
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "allow public read",
                "Action": ["s3:GetObject"],
                "Effect": "Allow",
                "Resource": ["/bucket-website/*", "/bucket-website/*/*"],
                "Principal": "*
            }
        ]
    }
]
```

Figure 16: Bucket policy example 2
Website Hosting

- *Website Redirection*
- *Applying the CORS Configuration to Buckets*

**Website Redirection**

When using buckets to host websites, Hedvig supports the *Website Redirect Location* feature to shorten a URL, because some resources sit in deep folder structures.

To shorten a URL:

1. Create an empty object directly under a bucket.

2. Set the value of its metadata key, `x-amz-website-redirect-location`, to be the target object location.

   ```
   aws s3api --endpoint http://<hostname>:<port_number> put-object
   --bucket <bucket_name> --key redirect.html
   --website-redirect-location "/f1/f2/../*/f1/f2/../*target_image"
   ```

   The long URL virtual-hosted style request to get the empty object is then redirected to the target object.

3. When you put the following in your browser:

   ```
   http://<bucket_name>..<hostname>:<port_number>/redirect.html
   ```

4. You are redirected to:

   ```
   http://<bucket_name>..<hostname>:<port_number>/f1/f2/../*target_image
   ```

**Applying the CORS Configuration to Buckets**

A CORS (Cross-Origin Resource Sharing) configuration can be applied to Hedvig S3 buckets to allow client web applications access from outside the domain.

CORS configurations can be used in scenarios such as running a static website on Hedvig S3 and allowing JavaScript from webpages to communicate with S3 buckets with certain operations such as GET, PUT, and DELETE.
CORS Configuration Example

This example shows a json file named hedvigcors.json for s3cli to enable CORS and set related CORS rules on a bucket.

This configuration allows requests from all domains to send GET requests and allows all headers.

In addition, it also allows POST PUT DELETE, with max age seconds of 3000, from the origin of:

http://bucket-website1.nginxvm.hedviginc.com:16000

{
 "CORSRules": [
  {
   "AllowedHeaders": ["*"],
   "AllowedMethods": ["POST", "PUT", "DELETE"],
   "MaxAgeSeconds": 3000
  },
  {
   "AllowedOrigins": ["*"],
   "AllowedMethods": ["GET"],
   "AllowedHeaders": ["*"]
  }
 ]
}

Use the AWS S3 CLI to apply the CORS rules to a bucket:

aws s3api put-bucket-cors --endpoint http://<hostname>:<port_number>
 --bucket <bucket_name> --cors-configuration file://hedvigcors.json
Bucket and Object Naming Restrictions

Here are the naming restrictions for **buckets**:

- The maximum bucket name length is 256 characters.
- The bucket name must be DNS compliant.
- Allowed characters are numbers, the alphabet, and the special character:
  - underscore (_)
  - hyphen (-)
  - period (.)

Here are the naming restrictions for **objects**:

- The minimum object name length is 1 byte.
- Allowed characters are numbers, the alphabet, and the special characters
  - exclamation mark (!)
  - backward slash (\)
  - quotation mark ("")
  - ampersand (&)
  - single quote (')
  - open and close parentheses ()
  - plus sign (+)
  - comma (,)
  - hyphen (-)
  - underscore (_)
  - period (.)
  - forward slash (/)
  - semicolon (;)
  - equal sign (=)
  - at sign (@)
  - asterisk (*)
Hedvig-supported S3 API Calls

S3 API calls that are supported by Hedvig are listed in Table 4.

Hedvig does not currently support accelerate, logging, notification, replication (which is not the same as Hedvig replication), or website operations on buckets.

Some operations that are not fully supported are noted in the Hedvig-supported AWS CLI Reference or in other sections of this document.

Table 4: S3 APIs supported by Hedvig

<table>
<thead>
<tr>
<th>name</th>
<th>applies to</th>
<th>purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>PUT Bucket</td>
<td>bucket</td>
<td>to create a bucket</td>
</tr>
<tr>
<td>PUT Bucket acl</td>
<td>bucket</td>
<td>to set the permission on an existing bucket using ACL</td>
</tr>
<tr>
<td>PUT Bucket cors</td>
<td>bucket</td>
<td>to set the CORS configuration for a bucket</td>
</tr>
<tr>
<td>PUT Bucket policy</td>
<td>bucket</td>
<td>to set the bucket policy for a bucket</td>
</tr>
<tr>
<td>PUT Bucket tagging</td>
<td>bucket</td>
<td>to add a set of tags to an existing bucket</td>
</tr>
<tr>
<td>PUT Bucket versioning</td>
<td>bucket</td>
<td>to set the versioning state of an existing bucket</td>
</tr>
<tr>
<td>PUT Object</td>
<td>object</td>
<td>to upload an object to a bucket if the user has permission</td>
</tr>
<tr>
<td>PUT Object ACL</td>
<td>object</td>
<td>to set the ACL for an object</td>
</tr>
<tr>
<td>name</td>
<td>applies to</td>
<td>purpose</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>PUT Object Copy</td>
<td>object</td>
<td>to copy an existing object. Copies metadata from the source object or replaces it with user-specified metadata.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• To copy a multipart object across a bucket, you must do an Initiate Multipart Upload and then copy individual parts from the source using Upload Part - Copy and then Complete Multipart Upload.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Copying a multipart object using PUT Object - Copy across buckets is not currently supported.</td>
</tr>
<tr>
<td>PUT Object Tagging</td>
<td>object</td>
<td>to set tags on an object</td>
</tr>
<tr>
<td>GET Bucket ACL</td>
<td>bucket</td>
<td>to get the ACL information associated with a bucket.</td>
</tr>
<tr>
<td>GET Bucket CORS</td>
<td>bucket</td>
<td>to return the CORS configuration associated with a bucket</td>
</tr>
<tr>
<td>GET Bucket (List Objects)</td>
<td>bucket</td>
<td>to list objects in a bucket</td>
</tr>
<tr>
<td>GET Bucket location</td>
<td>bucket</td>
<td>to return a Hedvig-specific region parameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note: The only value currently returned is Hedvig.</td>
</tr>
<tr>
<td>GET Bucket Object versions</td>
<td>bucket</td>
<td>to get the metadata of all versions of the objects in a bucket</td>
</tr>
<tr>
<td>GET Bucket policy</td>
<td>Bucket</td>
<td>to return the bucket policy associated with a bucket.</td>
</tr>
<tr>
<td>GET Bucket tagging</td>
<td>bucket</td>
<td>to return the tag set associated with a bucket</td>
</tr>
<tr>
<td>GET Bucket versioning</td>
<td>bucket</td>
<td>to get the versioning state of a bucket. Versioning states include unversioned, versioningenabled, and versioningsuspended.</td>
</tr>
<tr>
<td>name</td>
<td>applies to</td>
<td>purpose</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------</td>
</tr>
<tr>
<td>GET Object</td>
<td>object</td>
<td>to download an object from a bucket if the user has permission</td>
</tr>
<tr>
<td>GET Object ACL</td>
<td>object</td>
<td>to get the ACL of an object</td>
</tr>
<tr>
<td>GET Object tagging</td>
<td>object</td>
<td>to get tags for an object</td>
</tr>
<tr>
<td>GET Service</td>
<td>service</td>
<td>to list all buckets</td>
</tr>
<tr>
<td>DELETE Bucket</td>
<td>bucket</td>
<td>to delete a bucket</td>
</tr>
<tr>
<td>DELETE Bucket CORS</td>
<td>bucket</td>
<td>to delete a CORS configuration associated with a bucket</td>
</tr>
<tr>
<td>DELETE Bucket policy</td>
<td>bucket</td>
<td>to delete the bucket policy associated with a bucket</td>
</tr>
<tr>
<td>DELETE Bucket tagging</td>
<td>bucket</td>
<td>to remove a tag set from a bucket</td>
</tr>
<tr>
<td>DELETE Object</td>
<td>object</td>
<td>to delete an object from a bucket</td>
</tr>
<tr>
<td>DELETE Multiple Objects</td>
<td>object</td>
<td>to delete multiple objects from a bucket using a single HTTP request</td>
</tr>
<tr>
<td>DELETE object tagging</td>
<td>object</td>
<td>to delete tags for an object</td>
</tr>
<tr>
<td>HEAD Bucket</td>
<td>bucket</td>
<td>to determine if a bucket exists and if a user has permission to access it</td>
</tr>
<tr>
<td>HEAD Object</td>
<td>object</td>
<td>to get the metadata of an object</td>
</tr>
<tr>
<td>name</td>
<td>applies to</td>
<td>purpose</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>List Multipart Uploads</strong></td>
<td>bucket</td>
<td>to list in-progress multipart uploads</td>
</tr>
<tr>
<td><strong>Abort Multipart Upload</strong></td>
<td>object</td>
<td>to abort a multipart upload</td>
</tr>
<tr>
<td><strong>Complete Multipart Upload</strong></td>
<td>object</td>
<td>to complete a multipart upload by assembling previously uploaded parts</td>
</tr>
<tr>
<td><strong>Initiate Multipart Upload</strong></td>
<td>object</td>
<td>to initiate a multipart upload and return an upload ID</td>
</tr>
<tr>
<td><strong>List Parts</strong></td>
<td>object</td>
<td>to list the parts that have been uploaded for a specific multipart upload</td>
</tr>
<tr>
<td><strong>Upload Part</strong></td>
<td>object</td>
<td>to upload a part in a multipart upload</td>
</tr>
<tr>
<td><strong>Upload Part Copy</strong></td>
<td>object</td>
<td>to upload part data in a multipart upload by copying data from an existing object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- The byte range option is not supported when the source is not a multipart object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- If the source is a multipart object, then you must specify the byte range, which is an exact match of part boundaries in the source object.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Copying a whole multipart object as part data in a multipart upload is not currently supported.</td>
</tr>
</tbody>
</table>
Hedvig-supported AWS CLI Reference

This section is organized as follows:

- Operations on Buckets
- Operations on Objects
- Multipart Operations

Note the following about all operations:

- The Usage section shows the required parameters. The Supported Options section lists all Hedvig-supported parameters.

- Two options are common to most operations. For more information about these options, see Connecting to the Hedvig S3 Endpoint.
  --endpoint
  --profile

- For ACL related commands:
  - You can grant access to users using `id=<hedvigusername>` for specific users.
    --acl is used for canned ACLs with predetermined permission sets, for example:
    public-read, private, etc.
    --grant-read/write/acp for example:
    --grant-read 'id=testuer1,id=testuser2'

  --access-control-policy is used for defining access to the system using policy language.
  Predefined groups, all users, and authenticated users are supported.

- In operations with both a canned ACL and a grant ACL, the operation exclusively takes the canned ACL or the grant ACL. Mixed usage is ignored.
Operations on Buckets

Common Notes for Operations on Buckets

- Buckets can be deleted only when empty. For a versioned bucket, leftover objects might exist with delete markers to mask them from list-objects. Use `list-object-versions` to list them and explicitly delete the same, as needed.

- For the `--create-bucket-configuration <value>` option, see Specifying Hedvig Attributes during Bucket Creation.

put-bucket-acl

Sets the permission on an existing bucket using the access control list (ACL). To set the ACL of a bucket, you must have WRITE_ACP permission.

Usage

```
put-bucket-acl --bucket <value> [--acl | --grant-full-control |
    --grant-read | --grant-read-acp | --grant-write | --grant-write-acp |
    --access-control-policy ]
```

Supported Options

- **--bucket <value>**: Bucket name
- **--acl <value>**: Access Control List lets you manage access to buckets and objects. <value> specifies the following Hedvig-supported options:
  - private: Object owner permission only
  - public-read: Read permission by any users
  - public-read-write: Read and write permission by any users
  - authenticated-read: Read permission by authenticated users
- **--access-control-policy <value>**: Sets the ACL permissions for an object per grantee.
- **--grant-full-control <value>**: Allows grantee to read, write, read ACP, and write ACP permission on a bucket.
- **--grant-read <value>**: Allows grantee to list the objects in a bucket.
- **--grant-read-acp <value>**: Allows grantee to read the bucket ACL.
- **--grant-write <value>**: Allows grantee to create, overwrite, and delete any object in a bucket.
- **--grant-write-acp <value>**: Allows grantee to write the ACL for a bucket.

Example

```
aws s3api put-bucket-acl --endpoint http://<hostname>:<port_number>
    --profile testuser1 --bucket example_bucket --grant-read-acp id=testuser2
```
get-bucket-acl

Returns the access control list (ACL) of a bucket.

Usage
get-bucket-acl --bucket <value>

Supported Option
--bucket <value>: Bucket name

Example
aws s3api get-bucket-acl --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket

put-bucket-cors

Sets the CORS rules configuration for a bucket. For more information on supported policy elements, see Applying the CORS Configuration to Buckets.

Usage
put-bucket-cors --bucket <value> --cors-configuration

Supported Options
--bucket: Bucket name
--cors-configuration: a JSON structured string or a JSON file

Example
aws s3api put-bucket-cors --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket
   --cors-configuration file:///root/s3cors.json

get-bucket-cors

Gets the CORS rules configuration for a bucket.

Usage
get-bucket-cors --bucket <value>

Supported Option
--bucket: Bucket name

Example
aws s3api get-bucket-cors --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket
delete-bucket-cors

Deletes the CORS rules configuration for a bucket.

Usage
delete-bucket-cors --bucket <value>

Supported Option
--bucket: Bucket name

Example
aws s3api delete-bucket-cors --endpoint http://<hostname>:<port_number> --profile testuser1 --bucket example_bucket

put-bucket-policy

Associates a bucket policy with a bucket. For more information on the supported policy elements, see Access Control.

Usage
put-bucket-policy --bucket <value> --policy <value>

Supported Options
--bucket: Bucket name
--policy: Policy as a JSON document.

Example
aws s3api put-bucket-policy --endpoint http://<hostname>:<port_number> --profile testuser1 --bucket example_bucket --policy file://example_policy.json

get-bucket-policy

Gets the bucket policy associated with a bucket in JSON format.

Usage
get-bucket-policy --bucket value

Supported Options
--bucket: Bucket name

Example
aws s3api get-bucket-policy --endpoint http://<hostname>:<port_number> --profile testuser1 --bucket example_bucket
delete-bucket-policy

Deletes the bucket policy associated with a bucket.

Usage
delete-bucket-policy --bucket value

Supported Option
--bucket: bucket name

Example
aws s3api delete-bucket-policy --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket

put-bucket-tagging

Sets the tags for a bucket.

Usage
put-bucket-tagging --bucket <value> --tagging <value>

Supported Options
--bucket <value>: Bucket name
--tagging <value>: Key value pairs for the tag set

Example
aws s3api put-bucket-tagging --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket
   --tagging 'TagSet=[{Key=company,Value=commvault}]'

get-bucket-tagging

Returns the tag set associated with a bucket.

Usage
get-bucket-tagging --bucket <value>

Supported Option
--bucket <value>: Existing bucket name

Example
aws s3api get-bucket-tagging --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket
delete-bucket-tagging

Deletes all tags for a bucket.

**Usage**
delete-bucket-tagging --bucket <value>

**Supported Option**
--bucket <value>: Existing bucket name

**Example**
aws s3api delete-bucket-tagging --endpoint http://<hostname>:<port_number>  
--profile testuser1 --bucket example_bucket

put-bucket-versioning

Sets the versioning status of a bucket (must be done by bucket owner). For rules regarding versioning status transitions for bucket, see *Specifying Hedvig Attributes during Bucket Creation*.

**Usage**
put-bucket-versioning --bucket <value> --versioning-configuration <value>

**Supported Options**
--bucket <value>: Bucket name  
--versioning-configuration <value>: Values are Enabled and Suspended, for example: --versioning-configuration '{"Status": "Enabled"}'

**Example**
aws s3api put-bucket-versioning --endpoint http://<hostname>:<port_number>  
--profile testuser1 --bucket example_bucket  
--versioning-configuration '{"Status": "Enabled"}''

get-bucket-versioning

Returns the versioning state of a bucket.

**Usage**
get-bucket-versioning --bucket <value>

**Supported Option**
--bucket <value>: Existing bucket name

**Example**
aws s3api get-bucket-versioning --endpoint http://<hostname>:<port_number>  
--profile testuser1 --bucket example_bucket
create-bucket

Creates a new bucket.

Usage
create-bucket --bucket <value>

Supported Options
--acl <value>: Access Control List lets you manage access to buckets and objects. <value> specifies the following Hedvig-supported options:
  private: Object owner permission only
  public-read: Read permission by any users
  public-read-write: Read and write permission by any users
  authenticated-read: Read permission by authenticated users
--bucket <value>: Bucket name
--grant-full-control <value>: Grant full control (read, write, read ACP, and write ACP permissions) to user for this bucket
--grant-read <value>: Grant read control to user for objects in bucket
--grant-read-acp <value>: Grant read control to user for bucket ACL
--grant-write <value>: Grant control to user to create, overwrite, and delete any object in bucket
--grant-write-acp <value>: Grant control to user to write ACL for bucket

Example
aws s3api create-bucket --endpoint http://<hostname>:<port_number>
  --profile testuser1 --bucket example_bucket
  --create-bucket-configuration '{"LocationConstraint": "x-amz-meta-hedvig-blocksize=64"}"

delete-bucket

Deletes an existing bucket.

Usage
delete-bucket --bucket <value>

Supported Option
--bucket <value>: Existing bucket name

Example
aws s3api delete-bucket --endpoint http://<hostname>:<port_number>
  --profile testuser1 --bucket example_bucket
head-bucket

Determines if a bucket exists and if you have permission to access it.

- If it exists, and you have permission to access it, the operation returns a 200 OK.
- Otherwise, the operation might return 404 Not Found or 403 Forbidden.

**Usage**

```
head-bucket --bucket <value>
```

**Supported Option**

```
--bucket <value>: Existing bucket name
```

**Example**

```
aws s3api head-bucket --endpoint http://<hostname>:<port_number>
  --profile testuser1 --bucket example_bucket
```

list-buckets

Returns the list of buckets owned by the authenticated sender of the request.

**Usage**

```
list-buckets
```

**Supported Options**

None

**Example**

```
aws s3api list-buckets --endpoint http://<hostname>:<port_number>
  --profile testuser1
```
list-objects or list-objects-v2

Lists objects in a bucket.

Usage
list-objects --bucket <value>

Supported Options
--bucket: Bucket name
--delimiter: Separator for object path
--prefix: Prefix of object name
--starting-token: Starting token to retrieve parts
--pages-size: Pagination size
--max-items: Total object number to retrieve for this command’s output

Example
aws s3api list-objects-v2 --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket

list-object-versions

Lists object versions in a bucket.

Usage
list-object-versions --bucket <value>

Supported Options
--bucket: Bucket name
--delimiter: Separator for object path
--prefix: Prefix of object name
--starting-token: Starting token to retrieve parts
--pages-size: Pagination size
--max-items: Total object number to retrieve for this command’s output

Example
aws s3api list-object-versions --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket
Operations on Objects

Common Notes for Operations on Objects

The following table shows the limitations applicable to both copy-object and upload-part-copy.

Table 5: Limitations applicable to both copy-object and upload-part-copy

<table>
<thead>
<tr>
<th>source</th>
<th>destination</th>
<th>supported/implemented</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal Object</td>
<td>Normal Object</td>
<td>Yes</td>
</tr>
<tr>
<td>Normal Object</td>
<td>Part in Multipart Upload</td>
<td>Yes</td>
</tr>
<tr>
<td>Multipart Object</td>
<td>Multipart Object</td>
<td>Yes, for copy within bucket No, for copy across bucket</td>
</tr>
<tr>
<td>Multipart Object</td>
<td>Part in Multipart Upload</td>
<td>No</td>
</tr>
<tr>
<td>Multipart Object</td>
<td>Part in Multipart Upload with byte range</td>
<td>Yes, with byte range being part boundaries of part in source No, otherwise</td>
</tr>
</tbody>
</table>
put-object

Puts object to a bucket.

Usage
put-object --bucket <value> --key <value> --body <local_file_path>

Supported Options
--acl <value>: Access Control List (acl) permission for managing access to buckets. <value> can be one of these Hedvig-supported options:
  private: Object owner permission only
  public-read: Read permission by any users
  public-read-write: Read and write permission by any users
  authenticated-read: Read permission by authenticated users
  bucket-owner-read: Read permission by bucket owner
  bucket-owner-full-control: Full control by bucket owner
--body: File to put in object (local file path to upload)
--bucket: Bucket name
--content-disposition: Object presentational information
--content-encoding: Object content encoding method
--content-length: Object size in bytes.
--content-md5: Object MD5, used for object integrity check
--content-type: Content type of object
--grant-full-control: Grant full control to user
--grant-read: Grant read permission to user
--grant-read-acp: Grant read ACP to user
--grant-write-acp: Grant write ACP to user
--key: Object name
--metadata: User-defined map of key value pairs
--server-side-encryption: Object encryption, value: AES256

Example
aws s3api put-object --endpoint http://<hostname>:<port_number>
  --profile testuser1 --bucket example_bucket --key example_object
  --body ~/example_object
get-object

Downloads an object or versioned object.

Usage
get-object --bucket <value> --key <value>  <output file path>

Supported Options
--bucket: Bucket name
--key: Object name
--range: Object byte range for partial read
--version-id: Version id for versioned object
output file path: The local path to which the downloaded object is saved

Example
aws s3api get-object --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket --key example_object
testresult/example_object

delete-object

Deletes an object or versioned object.

Usage
delete-object --bucket <value> --key <value>

Supported Options
--bucket: Bucket name
--key: Object name
--version-id: Version id for versioned object

Example
aws s3api delete-object --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket --key example_object
delete-objects

Deletes multiple objects or versioned objects.

Usage
delete-objects --bucket <value> --delete <value>

Supported Options
--bucket: Bucket name
--delete: Object list with or without version id

Example
aws s3api delete-objects --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket
   --delete "Objects=[{Key=mobj},{Key=mobj2}],Quiet=false"

put-object-acl

Puts an ACL with a specified object.

Usage
put-object-acl --bucket <value> --key <value>

Supported Options
--acl <value>: Access Control List (acl) permission for managing access to
   buckets. <value> can be one of these Hedvig-supported options:
   private: Object owner permission only
   public-read: Read permission by any users
   public-read-write: Read and write permission by any users
   authenticated-read: Read permission by authenticated users
   bucket-owner-read: Read permission by bucket owner
   bucket-owner-full-control: Full control by bucket owner
--access-control-policy: access control policy, for example,
   --access-control-policy '{"Grants": [ { "Grantee": { "ID":
      "testuser2", "Type": "Group"}, "Permission":"FULL_CONTROL" } ] }'
--bucket: Bucket name
--content-md5: Object MD5
--grant-full-control: Grant full control to user
--grant-read: Grant read permission to user
--grant-read-acp: Grant read ACP to user
--grant-write-acp: Grant write ACP to user
--key: Object name
--version-id: Version id for versioned object

Example
aws s3api put-object-acl --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket --key example_object
   --acl public-read-write
get-object-acl

Retrieves the ACL for an object.

Usage
get-object-acl --bucket <value> --key <value>

Supported Options
--bucket: Bucket name
--key: Object name
--version-id: Version id for versioned object

Example
aws s3api get-object-acl --endpoint http://<hostname>:<port_number>
--profile testuser1 --bucket example_bucket --key example_object

put-object-tagging

Sets the tags for an object.

Usage
put-object-tagging --bucket <value> --key <value> --tagging <value>

Supported Options
--bucket: Bucket name
--key: Object name
--version-id: Version id for versioned object
--tagging <value>: Key value pairs for a tag set

Example
aws s3api put-object-tagging --endpoint http://<hostname>:<port_number>
--profile testuser1 --bucket example_bucket --key example_object
--tagging 'TagSet=[{Key=Head, Value=Commvault},
{Key=Branch, Value=Hedvig}]'
get-object-tagging

Gets the tags for an object.

Usage
get-object-tagging --bucket <value> --key <value>

Supported Options
--bucket: Bucket name
--key: Object name
--version-id: Version id for versioned object

Example
aws s3api get-object-tagging --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket --key example_object

delete-object-tagging

Deletes the tags from an object.

Usage
delete-object-tagging --bucket <value> --key <value>

Supported Options
--bucket: Bucket name
--key: Object name
--version-id: Version id for versioned object

Example
aws s3api delete-object-tagging --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket --key example_object
copy-object

Copies an object from a source object to a target object, or can be used to rename an object, or to change the metadata of an object.

Usage

copy-object --bucket <value> --copy-source <value> --key <value>

Supported Options

--acl <value>: Access Control List(acl) permission for managing access to buckets. <value> can be one of these Hedvig-supported options:
  - private: Object owner permission only
  - public-read: Read permission by any users
  - public-read-write: Read and write permission by any users
  - authenticated-read: Read permission by authenticated users
  - bucket-owner-read: Read permission by bucket owner
  - bucket-owner-full-control: Full control by bucket owner
--bucket: Bucket name
--content-disposition: Object presentation information
--content-encoding: Object content-encoding method
--content-type: Object content type
--copy-source: Source object to copy from. Format: <bucketName>/<objectName>?versionId=<versionid>
--grant-full-control: Grant full control to user
--grant-read: Grant read permission to user
--grant-read-acp: Grant read ACP permission to user
--grant-write-acp: Grant write ACP permission to user
--key: Object name
--metadata: User-defined map of key value pairs
--metadata-directive: REPLACE or COPY source object’s metadata
--server-side-encryption: Object encryption, value: AES256

Example

aws s3api copy-object --endpoint http://<hostname>:{<port_number>}
  --profile testuser1 --bucket target_example_bucket
  --key target_example_object --acl private
  --copy-source '/source_example_bucket/source_example_object'
head-object

Retrieves metadata from an object.

Usage
head-object --bucket <value> --key <value>

Supported Options
--bucket: Bucket name
--key: Object name
--version-id: Version id for versioned object

Example
aws s3api head-object --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_bucket --key example_object

Multipart Operations

Common Notes for Multipart Operations

- If a multipart upload is not aborted, space that is used by the parts that belong to this
  multipart upload will not be reclaimed. It is the user’s responsibility to make sure
  unnecessary or failed multipart uploads are cleaned up correctly.

- The size of each part, except the last, of a multipart upload object should be at least 5 MB.

- The part-number is not necessarily contiguous, that is, upload parts could be 3, 1, 5,
  but part numbers must be in ascending order when calling complete-multipart-upload.

- See Table 5: Limitations applicable to both copy-object and upload-part-copy.
abort-multipart-upload

Aborts the multipart upload of an object.

Usage
aborted-multipart-upload --bucket <value> --key <value> --upload-id <value>

Supported Options
--bucket: Bucket name
--key: Object name
--upload-id: Upload id of multipart upload object

Example
aws s3api abort-multipart-upload
  --endpoint http://<hostname>:<port_number> --profile testuser1
  --bucket example_mp_bucket --key example_mp_object --upload-id 4

complete-multipart-upload

Completes the multipart upload of an object.

Usage
complete-multipart-upload --bucket <value> --key <value>
  --upload-id <value> --multipart-upload <value>

Supported Options
--bucket: Bucket name
--key: Object name
--multipart-upload: Parts of object with ETag and PartNumber
--upload-id: Upload id of multipart upload object

Example
aws s3api complete-multipart-upload
  --endpoint http://<hostname>:<port_number> --profile testuser1
  --bucket example_mp_bucket --key example_mp_object --upload-id 2
  --multipart-upload "Parts=[{ETag="f66d18d577045c00b2e70f278a8e7508", PartNumber=1},{ETag="f66d18d577045c00b2e70f278a8e7508",PartNumber=2}]"
create-multipart-upload

Creates a multipart upload object.

Usage
create-multipart-upload --bucket <value> --key <value>

Supported Options
--acl <value>: Access Control List (acl) permission for managing access to buckets. <value> can be one of these Hedvig-supported options:
  - private: Object owner permission only
  - public-read: Read permission by any user
  - public-read-write: Read and write permission by any user
  - authenticated-read: Read permission by authenticated user
  - bucket-owner-read: Read permission by bucket owner
  - bucket-owner-full-control: Full control by bucket owner
--bucket: Bucket name
--grant-full-control: Grant full control to user
--grant-read: Grant read permission to user
--grant-read-acp: Grant read ACP to user
--grant-write-acp: Grant write ACP to user
--key: Object name
--metadata: User-defined key-value pair for target object
--server-side-encryption: Object encryption, value: AES256

Example
aws s3api create-multipart-upload
  --endpoint http://<hostname>:<port_number> --profile testuser1
  --bucket example_mp_bucket --key example_mp_object

list-multipart-uploads

Lists in-progress multipart upload objects in a bucket.

Usage
list-multipart-uploads --bucket <value>

Supported Option
--bucket: Bucket name

Example
aws s3api list-multipart-uploads
  --endpoint http://<hostname>:<port_number> --profile testuser1
  --bucket example_mp_bucket
list-parts

Lists parts for a multipart upload object.

Usage
list-parts --bucket <value> --key <value> --upload-id <value>

Supported Options
--bucket: Bucket name
--key: Object name
--upload-id: Upload id of multipart upload object

Example
aws s3api list-parts --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_mp_bucket --key example_mp_object
   --upload-id 7

upload-part

Uploads a part to a multipart upload object.

Usage
upload-part --bucket <value> --key <value> --part-number <value>
   --upload-id <value>

Supported Options
--body: File to upload to part
--bucket: Bucket name
--content-length: Part size in bytes
--content-md5: Part MD5, used for part integrity check
--key: Object name
--part-number: Sequence (user-specified part sequence number) of part in part list
--upload-id: Upload id of multipart upload object
--server-side-encryption: Object encryption, value: AES256

Example
aws s3api upload-part --endpoint http://<hostname>:<port_number>
   --profile testuser1 --bucket example_mp_bucket --key example_mp_object
   --part-number 1 --upload-id 2 --body ~/testinput/part1
upload-part-copy

Copies an object to form a part of a multipart upload object.

Usage

upload-part-copy --bucket <value> --copy-source <value> --key <value>
--part-number <value> --upload-id <value>

Supported Options

--bucket: Bucket name
--copy-source: Source object to copy from
--key: Object name
--part-number: Sequence of part in part list
--upload-id: Upload id of multipart upload object
--copy-source-range: Range of bytes to copy from source object/part
  (supported with limitations – see below)

Example

aws s3api upload-part-copy --endpoint http://<hostname>:<port_number>
--profile testuser1 --bucket example_mp_bucket --key example_mp_object
--copy-source '/example_mp_bucket/example_mp_object' --part-number 3
--upload-id 5
# Glossary

Table 3: Glossary

<table>
<thead>
<tr>
<th>term</th>
<th>definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACL</td>
<td>An <em>access control list</em> is a list of permissions attached to an object.</td>
</tr>
<tr>
<td>API</td>
<td>An <em>application programming interface</em> is a set of routines, protocols, and tools for building software and applications.</td>
</tr>
<tr>
<td>CIDR</td>
<td><em>Classless inter-domain routing</em> is a set of IP standards that is used to create unique identifiers for networks and individual devices. The IP addresses allow particular information packets to be sent to specific computers</td>
</tr>
<tr>
<td>CNAME</td>
<td><em>Canonical Name Record</em> or Alias Record is a type of resource record in the Domain Name System (DNS), that specifies that one domain name is an alias of another canonical domain name. Any system hosting a Web site must have an IP address in order to be connected to the World Wide Web.</td>
</tr>
<tr>
<td>DNS</td>
<td>The <em>Domain Name System</em> is a hierarchical and decentralized naming system for computers, services, or other resources connected to the Internet or a private network.</td>
</tr>
<tr>
<td><strong>Hedvig Storage Cluster</strong></td>
<td>A <em>Hedvig Storage Cluster</em> is an elastic cluster, formed by using any type of commodity server(s).</td>
</tr>
<tr>
<td><strong>Hedvig Storage Cluster Node</strong></td>
<td>A <em>Hedvig Storage Cluster Node</em> is an individual commodity server running Hedvig Storage Service software.</td>
</tr>
<tr>
<td>term</td>
<td>definition</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Hedvig Storage Proxy</td>
<td>A <em>Hedvig Storage Proxy</em> is a lightweight software component that deploys at the application tier as a virtual machine or Docker container, or on bare metal, to provide storage access to any physical host or virtual machine in the application tier. The storage proxy presents block, file, and object (Amazon S3) storage access to app hosts, accelerates read performance with flash caching, drives efficiency with deduplication, and secures data with encryption.</td>
</tr>
<tr>
<td>SDK</td>
<td>A <em>software development kit</em> is typically a set of software development tools that allows the creation of applications for a certain software package, software framework, hardware platform, computer system, video game console, operating system, or similar development platform.</td>
</tr>
<tr>
<td>S3</td>
<td><em>Amazon S3 (simple storage service)</em> is an online file storage web service.</td>
</tr>
</tbody>
</table>