SEAGATE

Lyve Cloud Object Storage API User Guide

LYVE™ Cloud

Click here to access an up-to-date online version
of this document. You will also find the most recent content as well as expandable illustrations, easier navigation, and search capability.

Contents

Introduction

This document describes the APIs offered by Seagate. The APIs are, where possible, modelled after the ones offered by AWS, to maximize interoperability with existing tools. The differences with the AWS APIs are explicitly noted.

Authentication

The provided APIs are authenticated via AWS Signature V4, using ar`access_key`, `secret_key` pair.
Requests can be signed either using any SDK compatible with AWS.

Credentials Management (STS)

The STS endpoint (https://sts.example.lyve.seagate.com) allows managing temporary user tokens and performing login with username and password. Requests to the STS endpoint must be signed specifying "sts" as service.

AssumeRole

Get a set of temporary keys for a user with access/secret keys:

`POST /?Action=AssumeRole`

POST Parameters

Parameter	Description
Action	must be set to AssumeRole
Version	must be set to 2011-06-15
DurationSeconds	token validity in seconds, between 15 minutes and 12 hours. Defaults to 15 minutes
RoleSessionName	required but ignored
RoleArn	required but ignored

Example Request

```
POST / HTTP/1.1
Host: sts.example.rstorcloud.io
Content-Length: 80
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Authorization: [...]
X-Amz-Date: [...]

Action=AssumeRole&RoleArn=Required1&RoleSessionName=Required2&Version=2011-06-15
```

Response

On success, an XML document containing the requested credentials at `AssumeRoleResult/Credentials` is returned. An error status code is returned otherwise.

```
<AssumeRoleResponse xmlns="https://sts.amazonaws.com/doc/2011-06-15/">
  <AssumeRoleResult>
    <Credentials>
      <SessionToken>42</SessionToken>
      <SecretAccessKey>EXAMPLESECRETACCESSKEYXXXXXXXXXXXXXXXXXXXXXXXXXXI6GGPQ</SecretAccessKey>
      <Expiration>2019-02-25T14:40:05.410Z</Expiration>
      <AccessKeyId>AWS4XXXEXAMPLEACCESSKEYID4UYJN3RQWFVFXBOP7FMOQLIXYZXYZ</AccessKeyId>
    </Credentials>
    <AssumedRoleUser>
      <Arn>arn:aws:sts::42:assumed-role/demo/Test</Arn>
      <AssumeRoleId>AWS4XXXEXAMPLEACCESSKEYID4UYJN3RQWFVFXBOP7FMOQLIDNXYZXYZ:Jim</AssumeRoleId>
    </AssumedRoleUser>
    <PackedPolicySize>42</PackedPolicySize>
  </AssumeRoleResult>
  <ResponseMetadata>
    <RequestId>42</RequestId>
  </ResponseMetadata>
</AssumeRoleResponse>
```

RSLogin

Perform a login retrieving the credential and the user details.

Parameter	Description
Customer	the name of the customer
UserName	the username (can be the email) of the user
Password	the current user password
OTP	a valid One Time Password (if needed)
DurationSeconds	specify the duration of the session (the validity time for the key)

Extra headers used to authenticate the request when whitelisting is enabled:

Header	Description

Header	Description
x-rstor-customer	specify the customer name

Example request

```
POST / HTTP/1.1
Host: sts.lyve.seagate.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:69.0) Gecko/20100101 Firefox/69.0
Accept: application/json, text/plain, */*
Accept-Language: en,it-IT;q=0.8,it;q=0.5,en-US;q=0.3
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded;charset=UTF-8
Content-Length: 123
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache
x-lyve-customer: fastvideo

Action=RSLogin&Version=2011-06-15&DurationSeconds=300&Customer=fastvideo&UserName=super@fastvideo.com&Password=Hg27TxBBfOzZntb&OTP=
```

Example response


```
{
   "RequestId":"050517212914402dd8aedfc50bb0dad0",
   "Credentials": {
      "SessionToken":"",
      "SecretAccessKey":"I/Ki9fZqmLkdWrAgIrKVdp6amEu9Q40CoRPVNpSeKVS",
      "Expiration":"2019-10-14T14:13:03.500533224Z",
      "AccessKeyId":"STX09VFCSKU9OZJY5Y4PLWD9"
   },
   "UserInfo": {
      "Id":"100000000209",
      "Name":"super@fastvideo.com",
      "PwdMustChange":false,
      "Email":"super@fastvideo.com",
      "Firstname":"Clark",
      "Lastname":"Kent",
      "Type":"user",
      "CreatedOn":"2019-10-03T15:37:52.635Z",
      "LastAccess":"2019-10-14T13:13:03.500533224Z",
      "Path":"",
      "Root":"lyve:fastvideo",
      "TFAEnabled":false
   }
}
```

RSAssumeCustomerRole

Retrieve credential to manage a Customer with `AssumeRole` enabled. In order to perform this action a specific account is used for security reason. Only a reseller account can perform this action.

Parameter	Description
Customer	the name of the customer
DurationSeconds	specify the duration of the session (the validity time for the key)

Example request

```
POST / HTTP/1.1
Host: sts.lyve.seagate.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:69.0) Gecko/20100101 Firefox/69.0
Accept: application/json, text/plain, */*
Accept-Language: en,it-IT;q=0.8,it;q=0.5,en-US;q=0.3
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded;charset=UTF-8
Content-Length: 83
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache

Action=RSAssumeCustomerRole&Version=2011-06-15&Customer=newcorp&DurationSeconds=300
```

Example response

```
{
   "Ok":true,
   "AccessKeyId":"STX06MSKQJCRTS4SJHHQJWOP",
   "SecretAccessKey":"RQTcszgc4xkZkmYXRJJLYR7adksTn8SBIalTP7rrpKa",
   "Expiration":"2019-10-14T14:29:43.919Z"
}
```

User Management (IAM)

The IAM endpoint (https://iam.example.lyve.seagate.com) allows managing user accounts and permissions. Requests to the IAM endpoint must be signed specifying "iam" as service.

User

A user can access the system either by logging in via email and password, or via one of its access keys, and perform actions according to its role and attached policies. Note that, for all users, the username coincides with the email.

A user belong to one of the following roles:

Role	Description
Root	Is the initial and main user of the account, has permissions on all buckets, and is able to manage users and policies. It cannot be deleted.
Admin	Can manage users and policies, with the exception of the Root account.
User	Can perform operations on buckets according to the policies it is attached. Can change own password, create and revoke own access keys.

Policies

A policy defines a set of permissions on buckets, and can be attached to a set of users.

AttachUserPolicy

Attaches a policy to a user.

POST parameters

Parameter	Description
Action	Must be set to `AttachUserPolicy`
Version	Must be set to `2010-05-08`
UserName	Name of the user the policy will be attached to
PolicyArn	ARN of the policy to attach

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: 86
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=AttachUserPolicy&
PolicyArn=arn:aws:iam::000000000000:policy/SomePolicy&
UserName=user@example.com&
Version=2010-05-08
```

Example response

```
<AttachUserPolicyResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>gf7hb0r6v5ix</RequestId>
  </ResponseMetadata>
</AttachUserPolicyResponse>
```

ChangePassword

Changes the password of the user making the request. It requires providing the old password.

POST parameters

Parameter	Description
Action	Must be set to `ChangePassword`
Version	Must be set to `2010-05-08`
OldPassword	Old password of the current user
NewPassword	New password for the current user. Must be at least 10 characters and contain numbers, lowercase and uppercase letters

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: 86
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=ChangePassword&
OldPassword=Password00&
NewPassword=Password01&
Version=2010-05-08
```

Example response

```
<ChangePasswordResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>EXAMPLE</RequestId>
  </ResponseMetadata>
</ChangePasswordResponse>
```

CreateAccessKey

Creates a new access key for the specified user.

POST parameters

Parameter	Description
Action	Must be set to `CreateAccessKey`
Version	Must be set to `2010-05-08`
UserName	Username of the user for whom the access key should be created. If not specified, it defaults to the user making the request.

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=CreateAccessKey&
UserName=user@example.com&
Version=2010-05-08
```

Example response

```
<CreateAccessKeyResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>830kau3a8zis</RequestId>
  </ResponseMetadata>
  <CreateAccessKeyResult>
    <AccessKey>
      <UserName>user@example.com</UserName>
      <AccessKeyId>AWS4AKIAKMEQGHROSDH7BKLB4QMNZAVANP4PGEZYJSWPBI4P4GTWGXDC</AccessKeyId>
      <Status>Active</Status>
      <SecretAccessKey>HCXL33NNPWX2EVEFHUZIGM234UJ64PUP2444OIFE3EOZHGVUG2GQ</SecretAccessKey>
    </AccessKey>
  </CreateAccessKeyResult>
</CreateAccessKeyResponse>
```

CreatePolicy

Creates a new policy. Note that, except for the PolicyDocument, the other parameters cannot be updated

after creation.

POST parameters

Parameter	Description
Action	Must be set to `CreatePolicy`
Version	Must be set to `2010-05-08`
Description	User-friendly description of the policy
PolicyDocument	Policy document in the AWS policy format
PolicyName	Name for the new policy

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=CreatePolicy&
Description=Gives all permissions for the bucket "abc-bucket"&
Path=/&
PolicyDocument={"Statement":[{"Action":
["s3:GetObject","s3:PutObject","s3:DeleteObject","s3:ListBucket"],"Effect":"Allow","Resource":"arn:aws:s3:::abc-bucket*"},{"A
ction":["s3:ListAllMyBuckets"],"Effect":"Allow","Resource":"*"}],"Version":"2012-10-17","Managed":true}&
PolicyName=NewPolicy4567&
Version=2010-05-08
```

Example response

```
<CreatePolicyResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>zd167x71f527</RequestId>
  </ResponseMetadata>
  <CreatePolicyResult>
```

```
    <Policy>
      <PolicyName>NewPolicy4567</PolicyName>
      <DefaultVersionId>v1</DefaultVersionId>
      <PolicyId>ANPASMVHTBHTUB7AMZUTK</PolicyId>
      <Path>/</Path>
      <Arn>arn:aws:iam::000000000000:policy/NewPolicy4567</Arn>
      <AttachmentCount>0</AttachmentCount>
      <CreateDate>2019-03-25T09:16:31.497Z</CreateDate>
      <UpdateDate>2019-03-25T09:16:31.497Z</UpdateDate>
      <Description></Description>
    </Policy>
  </CreatePolicyResult>
</CreatePolicyResponse>
```

CreatePolicyVersion

Updates a policy, creating a new version

POST parameters

Parameter	Description
Action	Must be set to `CreatePolicyVersion`
Version	Must be set to `2010-05-08`
SetAsDefault	If set to true, sets the new version as default, making it in effect
PolicyDocument	Policy document in the AWS policy format
PolicyArn	ARN of the policy to update

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]
```

```
Action=CreatePolicyVersion&
PolicyArn=arn:aws:iam::000000000000:policy/NewPolicy4567&
PolicyDocument={"Statement":[{"Action":["s3:GetObject","s3:ListBucket"],"Effect":"Allow","Resource":"*"},{"Action":["s3:List
AllMyBuckets"],"Effect":"Allow","Resource":"*"}],"Version":"2012-10-17"}&
SetAsDefault=true&
Version=2010-05-08
```

Example response

```
<CreatePolicyVersionResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>kbslm8tegjc8</RequestId>
  </ResponseMetadata>
  <CreatePolicyVersionResult>
    <PolicyVersion>
      <IsDefaultVersion>true</IsDefaultVersion>
      <VersionId>vGRC019YCN3QZ</VersionId>
      <CreateDate>2019-03-25T09:16:31.497Z</CreateDate>
    </PolicyVersion>
  </CreatePolicyVersionResult>
</CreatePolicyVersionResponse>
```

CreateUser

Creates a new user with the specified email address.

POST parameters

Parameter	Description
Action	Must be set to `CreateUser`
Version	Must be set to `2010-05-08`
UserName	Email of the user to create

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
```

```
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=CreateUser&
UserName=user2@example.com&
Version=2010-05-08
```

Example response

```
<CreateUserResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>t3v4bkwxvk0k</RequestId>
  </ResponseMetadata>
  <CreateUserResult>
    <User>
      <Path></Path>
      <UserName>cshhkvuc3sym@example.com</UserName>
      <UserId>100000000546</UserId>
      <Arn>arn:aws:iam::000000000000:user/cshhkvuc3sym@example.com</Arn>
    </User>
  </CreateUserResult>
</CreateUserResponse>
```

DeleteAccessKey

Deletes the specified access key. Both the email of the user the key belongs to and the access key id should be specified. If the email of the user is not provided, it defaults to the user making the request.

POST parameters

Parameter	Description
Action	Must be set to `DeleteAccessKey`
Version	Must be set to `2010-05-08`
UserName	Email of the user whose access key should be deleted
AccessKeyId	ID of the access key to delete

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

AccessKeyId=AWS4AKIAUAAECGIBEUZO3FG2GZBERAAGU2RXEVJYATF2A5XYSYCHCK4V&
Action=DeleteAccessKey&
UserName=user2@example.com&
Version=2010-05-08
```

Example response

```
<DeleteAccessKeyResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>wztbdx32kks5</RequestId>
  </ResponseMetadata>
</DeleteAccessKeyResponse>
```

DeletePolicy

Deletes the specified policy, if this is not attached to any user, and all of its previous versions (if any) have been previously deleted. You can use respectively `DetachUserPolicy` and `DeletePolicyVersion` to detach the policy from users, and to remove the previous policy versions.

POST parameters

Parameter	Description
Action	Must be set to `DeletePolicy`
Version	Must be set to `2010-05-08`
PolicyArn	ARN of the policy to delete

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=DeletePolicy&
PolicyArn=arn:aws:iam::000000000000:policy/TestA24152&
Version=2010-05-08
```

Example response

```
<DeletePolicyResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>alweq16xfjj9</RequestId>
  </ResponseMetadata>
</DeletePolicyResponse>
```

DeletePolicyVersion

Deletes the specified policy, if this is not attached to any user, and all of its previous versions (if any) have been previously deleted. You can use respectively `DetachUserPolicy` and `DeletePolicyVersion` to detach the policy from users, and to remove the previous policy versions.

POST parameters

Parameter	Description
Action	Must be set to `DeletePolicyVersion`
Version	Must be set to `2010-05-08`
PolicyArn	ARN of the policy whose version you wish to delete

Parameter	Description
VersionId	ID of the version to delete, as returned by `GetPolicy` or `ListPolicyVersions`

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=DeletePolicy&
PolicyArn=arn:aws:iam::000000000000:policy/TestA24152&
VersionId=vMFW10Z4EF4HL&
Version=2010-05-08
```

Example response

```
<DeletePolicyVersionResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>caonjqj712u7</RequestId>
  </ResponseMetadata>
</DeletePolicyVersionResponse>
```

DeleteUser

Deletes the specified user. Note that before deleting the user, you must first delete all of its keys, and detach all policies from it. Note also that a user cannot delete itself, and that a root user cannot be deleted.

POST parameters

Parameter	Description

Parameter	Description
Action	Must be set to `DeletePolicyVersion`
Version	Must be set to `2010-05-08`
UserName	Email of the user to delete

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=DeleteUser
&UserName=TestUser256@example.com
&Version=2010-05-08
```

Example response

```
<DeleteUserResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>5tk62707tkxv</RequestId>
  </ResponseMetadata>
</DeleteUserResponse>
```

DetachUserPolicy

Detaches a policy from a user.

POST parameters

Parameter	Description

Parameter	Description
Action	Must be set to `DetachUserPolicy`
Version	Must be set to `2010-05-08`
UserName	Email of the user to delete
PolicyArn	ARN of the policy the to detach

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=DetachUserPolicy&
PolicyArn=arn:aws:iam::000000000000:policy/ABCABC&
UserName=user1@example.com&
Version=2010-05-08
```

Example response

```
<DetachUserPolicyResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>a95qpalps1gc</RequestId>
  </ResponseMetadata>
</DetachUserPolicyResponse>
```

GetPolicy

Gets metadata about a given policy

POST parameters

Parameter	Description
Action	Must be set to `GetPolicy`
Version	Must be set to `2010-05-08`
PolicyArn	ARN of the policy

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=GetPolicy&
PolicyArn=arn:aws:iam::000000000000:policy/ABCABC&
Version=2010-05-08
```

Example response

```
<GetPolicyResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>1hxikd7dm49b</RequestId>
  </ResponseMetadata>
  <GetPolicyResult>
    <Policy>
      <PolicyName>ABCABC</PolicyName>
      <DefaultVersionId>vMFW10Z4EF4HL</DefaultVersionId>
      <PolicyId>ANVAP1OHCTVKCZR4TWB3F</PolicyId>
      <Path>/</Path>
      <Arn>arn:aws:iam::000000000000:policy/ABCABC</Arn>
      <AttachmentCount>2</AttachmentCount>
      <CreateDate>2019-03-14T11:23:36.776Z</CreateDate>
      <UpdateDate>2019-03-14T11:23:36.776Z</UpdateDate>
      <Description>policy description</Description>
    </Policy>
  </GetPolicyResult>
</GetPolicyResponse>
```

GetPolicyVersion

Gets the specified policy version, with its policy document

POST parameters

Parameter	Description
Action	Must be set to `GetPolicyVersion`
Version	Must be set to `2010-05-08`
PolicyArn	ARN of the policy
VersionId	ID of the policy version

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=GetPolicyVersion&
PolicyArn=arn:aws:iam::000000000000:policy/ABCABC&
Version=2010-05-08&
VersionId=vMFW10Z4EF4HL
```

Example response

```
<GetPolicyVersionResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>s9h55m6gm6st</RequestId>
  </ResponseMetadata>
  <GetPolicyVersionResult>
    <PolicyVersion>
```

<Document>{"Statement":[{"Action":["s3:GetObject","s3:PutObject","s3:DeleteObject","s3:ListBucket"],"Effect":"Allow","Resource":"arn:aws:s3:::testbucket*"},{"Action":["s3:PutObject","s3:DeleteObject"],"Effect":"Allow","Resource":"arn:aws:s3:::abc-bucket*"},{"Action":["s3:ListAllMyBuckets"],"Effect":"Allow","Resource":"*"}],"Version":"2012-10-17","Managed":true}</Document>
<IsDefaultVersion>true</IsDefaultVersion>
<VersionId>vMFW10Z4EF4HL</VersionId>
<CreateDate>2019-03-14T11:23:36.776Z</CreateDate>
</PolicyVersion>
</GetPolicyVersionResult>
</GetPolicyVersionResponse>

ListAccessKeys

Retrieves a list of the access keys belonging to a given user

POST parameters

Parameter	Description
Action	Must be set to `ListAccessKeys`
Version	Must be set to `2010-05-08`
UserName	Email of the user whose access keys should be listed

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=ListAccessKeys&
UserName=user@example.com&
Version=2010-05-08
```

Example response

```
<ListAccessKeysResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>hhbsxpz2r66y</RequestId>
  </ResponseMetadata>
  <ListAccessKeysResult>
    <AccessKeyMetadata>
      <member>
        <UserName>user@example.com</UserName>
        <AccessKeyId>AWS4AKIANWYVBGWUGE2PQY7HPOH4QNI6SDDSA6UPQFCTRFLHVCKBZCGA</AccessKeyId>
        <Status>Active</Status>
        <CreateDate>2019-03-15T16:56:56.434Z</CreateDate>
      </member>
      <member>
        <UserName>user@example.com</UserName>
        <AccessKeyId>AWS4AMQVFJACLX5R6FCBRFR3Z2LMYP5KBFBQCVQW3JFARBENYJC6SLZA</AccessKeyId>
        <Status>Active</Status>
        <CreateDate>2019-03-14T12:00:30.358Z</CreateDate>
      </member>
    </AccessKeyMetadata>
    <UserName>user@example.com</UserName>
    <IsTruncated>false</IsTruncated>
  </ListAccessKeysResult>
</ListAccessKeysResponse>
```

ListAttachedUserPolicies

Retrieves a list of the policies attached to a given user.

Parameter	Description
Action	Must be set to `ListAttachedUserPolicies`
Version	Must be set to `2010-05-08`
UserName	Email of the user whose attached policies should be listed

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
```

```
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=ListAttachedUserPolicies&
UserName=user@example.com&
Version=2010-05-08
```

Example response

```
<ListAttachedUserPoliciesResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>6brrrkcl3nem</RequestId>
  </ResponseMetadata>
  <ListAttachedUserPoliciesResult>
    <AttachedPolicies>
      <member>
        <PolicyName>Policy1</PolicyName>
        <PolicyArn>arn:aws:iam::000000000000:policy/Policy1</PolicyArn>
      </member>
    </AttachedPolicies>
    <IsTruncated>false</IsTruncated>
    <Marker></Marker>
  </ListAttachedUserPoliciesResult>
</ListAttachedUserPoliciesResponse>
```

ListEntitiesForPolicy

Retrieves a list of all the users to which the specified policy is attached.

POST parameters

Parameter	Description
Action	Must be set to `ListEntitiesForPolicy`
Version	Must be set to `2010-05-08`
PolicyArn	ARN of the policy

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=ListEntitiesForPolicy
&PolicyArn=arn:aws:iam::000000000000:policy/ABCABC
&Version=2010-05-08
```

Example response

```
<ListEntitiesForPolicyResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>7pzdvnk5er4i</RequestId>
  </ResponseMetadata>
  <ListEntitiesForPolicyResult>
    <PolicyUsers>
      <member>
        <UserName>user2@example.com</UserName>
      </member>
      <member>
        <UserName>user@example.com</UserName>
      </member>
    </PolicyUsers>
    <PolicyRoles></PolicyRoles>
    <PolicyGroups></PolicyGroups>
    <IsTruncated>false</IsTruncated>
  </ListEntitiesForPolicyResult>
</ListEntitiesForPolicyResponse>
```

ListPolicies

Retrieves a list of all the policies. The list can be optionally filtered by specifying PathPrefix or OnlyAttached.

POST parameters

Parameter	Description
Action	Must be set to `ListPolicies`
Version	Must be set to `2010-05-08`
PathPrefix	If specified, only the policies having a Path with the given prefix are returned
OnlyAttached	If specified, only the policies attached to some user are returned

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=ListPolicies
&Version=2010-05-08
```

Example response

```
<ListPoliciesResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
 <ResponseMetadata>
  <RequestId>s5eb8nlvydf2</RequestId>
 </ResponseMetadata>
 <ListPoliciesResult>
  <Policies>
   <member>
    <PolicyName>Policy1</PolicyName>
    <DefaultVersionId>vMFW10Z4EF4HL</DefaultVersionId>
    <PolicyId>ANVAP1OHCTVKCZR4TWB3F</PolicyId>
    <Path>/</Path>
    <Arn>arn:aws:iam::000000000000:policy/Policy1</Arn>
    <AttachmentCount>2</AttachmentCount>
    <CreateDate>2019-03-14T11:23:36.776Z</CreateDate>
    <UpdateDate>2019-03-14T11:23:36.776Z</UpdateDate>
    <Description>first test policy</Description>
   </member>
```

```
      <member>
        <PolicyName>Policy2</PolicyName>
        <DefaultVersionId>v0169a042bpk3</DefaultVersionId>
        <PolicyId>ANVAEQ3KKQOMWALATC1QT</PolicyId>
        <Path>/</Path>
        <Arn>arn:aws:iam::000000000000:policy/Policy2</Arn>
        <AttachmentCount>3</AttachmentCount>
        <CreateDate>2019-03-12T14:11:38.767Z</CreateDate>
        <UpdateDate>2019-03-12T14:11:38.767Z</UpdateDate>
        <Description>second test policy</Description>
      </member>
    </Policies>
    <IsTruncated>false</IsTruncated>
    <Marker></Marker>
  </ListPoliciesResult>
</ListPoliciesResponse>
```

ListPolicyVersions

Retrieves a list of versions of a specified policy

POST parameters

Parameter	Description
Action	Must be set to `ListPolicyVersions`
Version	Must be set to `2010-05-08`
PolicyArn	ARN of the policy

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=ListPolicyVersions
&PolicyArn=arn:aws:iam::000000000000:policy/TestPolicy
```

```
&Version=2010-05-08
```

Example response

```
<ListPolicyVersionsResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>s77xvbxffqqb</RequestId>
  </ResponseMetadata>
  <ListPolicyVersionsResult>
    <Versions>
      <member>
        <IsDefaultVersion>false</IsDefaultVersion>
        <VersionId>v1</VersionId>
        <CreateDate>2019-03-14T11:23:36.776Z</CreateDate>
      </member>
      <member>
        <IsDefaultVersion>true</IsDefaultVersion>
        <VersionId>v2Z0F1XI574I7</VersionId>
        <CreateDate>2019-03-14T11:37:20.523Z</CreateDate>
      </member>
    </Versions>
    <IsTruncated>false</IsTruncated>
    <Marker></Marker>
  </ListPolicyVersionsResult>
</ListPolicyVersionsResponse>
```

ListUsers

List all users.

POST parameters

Parameter	Description
Action	Must be set to `ListUsers`
Marker	Pagination support. Unused.
MaxItems	Pagination support. Unused.
PathPrefix	User filtering. Not supported yet.

Parameter	Description
Version	Must be set to `2010-05-08`

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: [...]
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=ListUsers
&Version=2010-05-08
```

Example response

```
<ListUsersResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>mujpkfxe7slp</RequestId>
  </ResponseMetadata>
  <ListUsersResult>
    <Users>
      <member>
        <UserId>100000000042</UserId>
        <Path>/</Path>
        <UserName>testuser@example.com</UserName>
        <Arn>arn:aws:iam::000000000000:user/testuser@example.com</Arn>
        <CreateDate>2019-03-25T10:56:28.523Z</CreateDate>
        <PasswordLastUsed>2019-03-25T10:56:28.523Z</PasswordLastUsed>
      </member>
      <member>
        <UserId>100000000024</UserId>
        <Path></Path>
        <UserName>example@example.com</UserName>
        <Arn>arn:aws:iam::000000000000:user/example@example.com</Arn>
        <CreateDate>2019-03-25T10:56:28.523Z</CreateDate>
        <PasswordLastUsed>2019-03-25T10:56:28.523Z</PasswordLastUsed>
      </member>
    </Users>
    <IsTruncated>false</IsTruncated>
  </ListUsersResult>
```

```
</ListUsersResponse>
```

SetDefaultPolicyVersion

Set the specified version of a policy as the default (active) version.

Note that this changes the currently active version for all attached entities.

POST parameters

Parameter	Description
Action	Must be set to `SetDefaultPolicyVersion`
PolicyArn	ARN of the policy. example: `arn:aws:iam::123456789012:policy/policy-name`
VersionId	Version of the policy that you want as default
Version	Must be set to `2010-05-08`

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Content-Sha256: [...]
X-Amz-Date: 20190325T103156Z
Authorization: [...]
Content-Length: [...]

Action=SetDefaultPolicyVersion
&PolicyArn=arn:aws:iam::123456789012:policy/policy-name
&VersionId=v42
&Version=2010-05-08
```

Example response

```
<SetDefaultPolicyResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>42bgut3b75zt</RequestId>
  </ResponseMetadata>
</SetDefaultPolicyResponse>
```

UpdateLoginProfile

Changes the password for the specified IAM user. For the current user use the "ChangePassword" API instead

POST parameters

Parameter	Description
Action	Must be set to `UpdateLoginProfile`
Password	New password. Needs uppercase, lowercase, digits and at least 10 total characters
PasswordResetRequired	True/false
UserName	Username to reset the password to
Version	Must be set to `2010-05-08`

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
```

```
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Content-Sha256: [...]
X-Amz-Date: 20190325T103156Z
Authorization: [...]
Content-Length: [...]

Action=UpdateLoginProfile
&Password=Password22
&PasswordResetRequired=false
&UserName=testuser@example.com
&Version=2010-05-08
```

Example response

```
<UpdateLoginProfileResponse xmlns="https://iam.amazonaws.com/doc/2010-05-08/">
  <ResponseMetadata>
    <RequestId>27bgut3b75zt</RequestId>
  </ResponseMetadata>
</UpdateLoginProfileResponse>
```

RSGetUserInfo

Retrieves information about a list of users. If no user is specified, information about the current user is returned. Only Root or Admin users can retrieve information about other users.

POST parameters

Parameter	Description
Action	Must be set to `RSGetUserInfo`
Version	Must be set to `2010-05-08`
UserList.$number	Username whose information should be retrieved

Response

The response is a json document containing, for each user:

- Id: the user id

- Name: username (coincides with the user email)
- PwdMustChange: password validity status
- Email
- Firstname
- Lastname
- Type: role (admin, root, or user)
- CreatedOn: time the user was created
- LastAccess: time of last access
- Path: path specified when the user was created
- Root: root account this user belongs to

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Content-Sha256: [...]
X-Amz-Date: 20190325T103156Z
Authorization: [...]
Content-Length: [...]

Action=RSGetUserInfo&
UserList.1=firstuser%40example.com&
UserList.2=seconduser%40example.com&
UserList.3=nonexistent&
Version=2010-05-08
```

Example response


```
[
  {
    "Id": "100000000780",
    "Name": "firstuser@example.com",
    "PwdMustChange": true,
    "Email": "firstuser@example.com",
    "Firstname": "First",
    "Lastname": "User",
    "Type": "admin",
    "CreatedOn": "2019-05-02T06:26:06.818Z",
    "LastAccess": "0001-01-01T00:00:00Z",
    "Path": "",
    "Root": "example",
```

```
    "AuthMethod": "password"
  },
  {
    "Id": "100000000781",
    "Name": "seconduser@example.com",
    "PwdMustChange": true,
    "Email": "seconduser@example.com",
    "Firstname": "First",
    "Lastname": "User",
    "Type": "user",
    "CreatedOn": "2019-05-02T06:26:06.859Z",
    "LastAccess": "0001-01-01T00:00:00Z",
    "Path": "",
    "Root": "example",
    "AuthMethod": "federated"
  }
]
```


RSSetUserInfo

Sets information for a given user.

POST parameters

Parameter	Description
Action	Must be set to `RSSetUserInfo`
Version	Must be set to `2010-05-08`
Name	User whose information should be set
Firstname	First name to set
Lastname	Last name to set
Type	Type of user (`root` , `admin` , `user`)

Response

Returns a status code reflecting the outcome of the operation.

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Content-Sha256: [...]
X-Amz-Date: 20190325T103156Z
Authorization: [...]
Content-Length: [...]

Action=RSSetUserInfo&
Version=2010-05-08&
Name=firstuser@example.com&
Firstname=John&
Lastname=Smith&
Type=user
```

RSGetPasswordResetToken (RSGetToken)

Returns the requested password reset token for a given user. Root and Admin users can obtain reset tokens for other users; in this case, the old password is immediately invalidated. Generated tokens are valid for 6 hours.

POST parameters

Parameter	Description
Action	Must be set to `RSGetToken`
Version	Must be set to `2010-05-08`
Type	Must be set to `pwd_reset`
UserName	User whose password reset token should be obtained (defaults to the current user)

Response

Returns the requested password reset token in a JSON object.

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Content-Sha256: [...]
X-Amz-Date: 20190325T103156Z
Authorization: [...]
Content-Length: [...]

Action=RSGetToken&
Version=2010-05-08&
Type=pwd_reset&
UserName=firstuser%40example.com
```

Example response

```
{
  "Token": "AAAAAAAAAAAAAFMCAAAAAAAAAAAAAAAAATMYFBVC2V3PJH3MDVOPV62XKJYRP7K2AUE6LCEV45ZQK4IA776W4F6A2OY2XYXJCHMJ5U4MJ2NXGQTK2XVBWVHQG43A3CC2PPX5NSKI="
}
```

RSResetPassword

Resets a user's password using a password reset token. Note this call is not authenticated.

POST parameters

Parameter	Description
Action	Must be set to `RSGetToken`
Version	Must be set to `2010-05-08`
Token	Password reset token
NewPassword	New password to set

Response

Returns a status code reflecting the outcome of the operation.

Example request

Newlines added for clarity.

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Content-Length: [...]

Action=RSResetPassword&
Version=2010-05-08&
Token=output.Token&
NewPassword=Password24
```

RSPrepareTFA

Prepares a 2FA secret for current user. After obtaining the secret, user can enable 2FA via `RSEnableTFA` API. The secret expires after 30 minutes.

POST parameters

Parameter	Description
Action	Must be set to `RSPrepareTFA`
Version	Must be set to `2010-05-08`

Response

Returns a secret key for generating time-based one-time passwords and a QR code image containing OTP URI with that secret.

Example request

```
POST / HTTP/1.1
```

```
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Content-Sha256: [...]
X-Amz-Date: [...]
Authorization: [...]
Content-Length: [...]

Action=RSPrepareTFA&
Version=2010-05-08
```

Example response

```
{
  "secret": "RCULN35A63IMSO5UMW6FSUJIOTUOCDE7...",
  "qrCodeUrl": "..."
}
```

RSEnableTFA

Verifies the received OTP and enables 2FA for current user.

Parameter	Description
Action	Must be set to `RSEnableTFA`
Version	Must be set to `2010-05-08`
OTP	6-digit OTP derived from the secret from `RSPrepareTFA`

Response

Returns 10 recovery codes that can be used instead of OTP in case the user loses their 2FA device.

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
```

```
X-Amz-Content-Sha256: [...]
X-Amz-Date: [...]
Authorization: [...]
Content-Length: [...]

Action=RSEnableTFA&
Version=2010-05-08&
OTP=880631
```

Example response

```
{
  "recoveryCodes": [
    "52895ace",
    "f20f43ef",
    "f1ebaf4d",
    ...
  ]
}
```

RSDisableTFA

Disables 2FA for current user.

Parameter	Description
Action	Must be set to `RSDisableTFA`
Version	Must be set to `2010-05-08`

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
X-Amz-Content-Sha256: [...]
X-Amz-Date: [...]
Authorization: [...]
Content-Length: [...]
```

```
Action=RSDisableTFA&
Version=2010-05-08
```

RSCreateCustomer

Create a new customer under the current Reseller. This action should be performed using a `Reseller` account.

Parameter	Description
CustomerName	The name of the new customer (is enforced lowercase by the backend)
CustomerEmail	The email of the new customer
AssumeRoleControl	Specify who can control "AssumeRole" option for this customer: {"reseller", "customer", ""}
	"reseller" => only the reseller can control it
	"customer" => only the customer can decide if want to be managed or not
	"" => if not set, "AssumeRole" is disabled
ReplicationEnforcement	Specify if the customer can choose the replica location
Regions	Is a list containing the available Regions for the given customer
RestrictServiceAccess	Optional; if true, customer will not have access to any services initially. Services can be enabled through `iam:RSSetCustomerServiceAccessLevel` action

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: 182
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]
```

```
Action=RSCreateCustomer&
Version=2010-05-08&
Name=NewCorp&
Email=admin%40newcorp.com&
Password=Password22&
AssumeRoleControl=customer&
ReplicationEnforcement=false&
ReplicationPolicy=%7B%7D
```

Example response

```
{
  "Ok": true,
  "Root": "lyve:newcorp",
  "RootID": "100000000288"
}
```

RSModifyCustomer

Modify an existing customer properties.

Parameter	Description
CustomerName	The name of the new customer
AssumeRoleEnabled	Set if the AssumeRole is enabled or not
Status	Set the status for the specified customer: "enabled", "disabled", "deleted"
ReplicationEnforcement	Specify if the customer can choose the replica location
Regions	Is a list containing the available Regions for the given customer

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: 133
```

```
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=RSModifyCustomer&
Version=2010-05-08&
CustomerName=newcorp&
ReplicationEnforcement=false&
ReplicationPolicy=null&
Status=disabled
```

Example response

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: Etag,X-Lyve-Size,X-Lyve-Replication-Status
Date: Mon, 14 Oct 2019 09:17:58 GMT
Content-Length: 0
```

RSListCustomer

Return a list of all Customers. Only a reseller account can perform this request.

Example request

```
POST /?Action=RSListCustomer&Version=2010-05-08
Host: iam.example.lyve.seagate.com
```

Example response


```
[
  {
    "Name":"newcorp",
    "RootID":"100000000288",
    "Active":false,
    "CreatedOn":"2019-10-14T08:28:10.029Z",
    "SuspendedAt":"2019-10-14T09:17:58.647Z",
    "AssumeRoleControl":"customer",
    "AssumeRoleEnabled":false,
    "Regions":["DCA02", "SJC03"],
```

```
    "ReplicationEnforcement":false
  },
]
```


RSCustomerDetails

Return details about a specific customer. Only a reseller account can perform this request.

Example request


```
POST /?Action=RSCustomerDetails&Version=2010-05-08&CustomerName=newcorp
Host: iam.example.lyve.seagate.com
```


Example response


```
{
  "Email":"admin@newcorp.com",
  "LastAccess":"2019-10-12T13:17:14.760Z",
  "AssumeRoleControl":"customer",
  "AssumeRoleEnabled":true,
  "Active":true,
  "NumBuckets":3,
  "NumFiles":12,
  "NumUsers":2,
  "NumAdminUsers":0,
  "NumAdminRoots":2,
  "UsedSpace":741987
}
```


RSAvailableRegions

Return a sorted list of regions available to the customer making the request and the value of the `ReplicationEnforcement` setting.

If `ReplicationEnforcement` is set to true, the customer is not able to create buckets with a different replication policy.

When called by a reseller, the `Customer` parameter allows retrieving the replication policy of a specific

customer.

Example request

```
POST /?Action=RSAvailableRegions&Version=2010-05-08&Customer=newcorp
Host: iam.example.lyve.seagate.com
```

Example response

```
{
  "Regions": ["SJC03", "DCA02"],
  "ReplicationEnforcement": true,
}
```

RSListBillingData

Serves per-day statistics and billing data. The response consists of:

- `Stats`: per-datacenter transfer statistics
- `UsedSpace`: used and ghost space, used for billing.
- `ObjectCount`: total number of total objects

Entries are in reverse-chronological order.

POST parameters

Parameter	Description
Action	Must be set to `RSListBillingData`
Version	Must be set to `2010-05-08`
From	ISO-8601-formatted UTC date of the first requested day
Till	ISO-8601-formatted UTC date of the last requested day

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Type: application/x-www-form-urlencoded
Authorization: [...]

Action=RSListBillingData&
Version=2010-05-08&
From=2020-03-04T00%3A00%3A00.000Z&
Till=2020-04-04T00%3A00%3A00.000Z
```

Example response

```
{
  "Stats": {
    "DCA02": [
      {
        "Date": "2020-04-03T00:00:00Z",
        "UploadBytes": 28398312,
        "DownloadBytes": 56709864,
        "DeleteBytes": 28398312
      },
      // ...
    ],
    // ...
  },
  "UsedSpace": [
    {
      "Date": "2020-04-03T00:00:00Z",
      "UsedSpace": 6485663588183,
      "GhostSpace": 402901552935
    },
    // ...
  ],
  "ObjectCount": [
    {
      "Date": "2020-04-03T00:00:00Z",
      "ObjectCount": 18499
    },
    // ...
  ]
}
```

RSLiveBilling

Serves recent billing metrics for the last hour. Entries for the last two minutes are omitted in order to ensure the returned data is complete.

The API returns stats aggregated by one-minute windows, ordered from the most recent to the oldest. In case there is no activity in a time window, the corresponding entry is omitted.

POST parameters

Parameter	Description
Action	Must be set to `RSLiveBilling`
Version	Must be set to `2010-05-08`

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Type: application/x-www-form-urlencoded
Authorization: [...]

Action=RSLiveBilling&
Version=2010-05-08
```

Example response


```
[
  {
    "date": "2020-04-23T12:35:00Z",
    "uploadedBytes": 24623534,
    "uploadedObjects": 3,
    "deletedBytes": 0,
    "deletedObjects": 0
  },
  {
    "date": "2020-04-23T12:34:00Z",
    "uploadedBytes": 24623534,
    "uploadedObjects": 3,
    "deletedBytes": 0,
    "deletedObjects": 0
  },
  {
    "date": "2020-04-23T12:29:00Z",
    "uploadedBytes": 2786761,
```

```
    "uploadedObjects": 12,
    "deletedBytes": 0,
    "deletedObjects": 0
  }
]
```

RSWhitelistAddRule

Add a new whitelist rule to the current list of whitelist subnets.

Property	Value
Permission required	ADMIN
Service	IAM
UrlPath	/api/whitelist/RSWhitelistAddRule
Method	PUT
Content-Type	json

Request body

```
{
  "root": <root_name>, // mandatory
  "subNet": <subnet>, // mandatory
  "regions": ["DCA02", "SCJ03", "DEN02", "LON01"], // optional (if empty apply to all regions)
  "notes": <some notes for the reseller>, // optional
  "ttl": <time to live in days after the approval> // optional
}
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json
Date: Mon, 27 Apr 2020 14:16:59 GMT
Content-Length: 109
```

{"ok":true,"ruleId":"c205e1b6-8891-11ea-968c-14cc2006610b","insertedAt":"2020-04-27T16:16:59.50222599+02:00"}

RSWhitelistDeleteRule

Remove a rule from the current subnet whitelist.

Property	Value
Permission required	RESELLER if the Rule is Active (was approved), ADMIN or RESELLER if the rule is still pending
Service	IAM
UrlPath	/api/whitelist/RSWhitelistDeleteRule
Method	DELETE
Content-Type	json

Request body

```
{
  "root": <root_name>, // mandatory
  "ruleId": <rule identity number> // mandatory
}
```

Response

Simple http response + json body { "ok": true/false, "errorDesc": "optional error description" }

RSWhitelistApproveRule

Approve a rule in `Pending` state.

Property	Value

Property	Value
Permission required	RESELLER
Service	IAM
UrlPath	/api/whitelist/RSWhitelistApproveRule
Method	POST
Content-Type	json

Request body

```
{
  "root": <root_name>, // mandatory
  "ruleId": <rule identity number>, // mandatory
  "notes": <some notes to report> // optional
}
```

Response

```
{
  "ok": <true,false>,
  "errorDesc": "" // optional,
  "ruleId": <rule identity number>,
  "status": <current_status>,
  "updatedAt": <update ts>
}
```

RSWhitelistRejectRule

Reject a rule in Pending state.

Property	Value

Property	Value
Permission required	RESELLER
Service	IAM
UrlPath	/api/whitelist/RSWhitelistRejectRule
Method	POST
Content-Type	json

Request body

```
{
  "root": <root_name>, // mandatory
  "ruleId": <rule identity number>, // mandatory
  "notes": <some notes to report> // optional
}
```

Response

```
{
  "ok": <true,false>,
  "errorDesc": "" // optional,
  "ruleId": <rule identity number>,
  "status": <current_status>,
  "updatedAt": <update ts>
}
```

RSWhitelistListRules

List all rules (subnets).

Property	Value

Property	Value
Permission required	RESELLER
Service	IAM
UrlPath	/api/whitelist/RSWhitelistListRules
Method	GET

Get Parameters

Param	Desc
root	root name
offset	0 (used for pagination, not yet implemented)
limit	0 (used for pagination, not yet implemented)

Responses

```
{
  "ok": <true,false>,
  "partial": <true,false> // if true means that only some rules are returned,
  "rules": [
    {
      "ruleId": <rule identifier>, // always present
      "insertedAt": <timestamp>, // always present
      "updatedAt": <timestamp>, // always present
      "subNet": <subnet>, // always present
      "status": <Active,Pending,Rejected,Suspended>, // always present
      "regions": ["DCA02", "SCJ03", "DEN02", "LON01"], // optional (if empty apply to all regions)
      "notes": <rule notes> // optional
    }
  ],
  "errorDesc": "optional error description"
}
```

RSWhitelistStats

Upcoming.

RSSetCustomerServiceAccessLevel

Configures a service for a customer with RestrictServiceAccess enabled.

Property	Value
CustomerName	Name of the customer to manage
Service	Name of the service to manage, must be one of:
	`space`, `transporter`, `insights`, `protect`
Enabled	`true` / `false` to enable / disable the service
ExpirationDate	This setting has no effect if `Enabled` is `false`
	If set, the service will be available until the specified date
	If omitted, the service will be available indefinitely

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
Content-Length: 86
Content-Type: application/x-www-form-urlencoded
Authorization: [...]
X-Amz-Date: [...]

Action=RSSetCustomerServiceAccessLevel&
Service=transporter&
AccessLevel=enabled
```

RSCreateSAMLConnection

Create SAML connection for Federated Login (SSO).

POST parameters

Parameter	Description
Action	Must be set to `RSCreateSAMLConnection`
Version	Must be set to `2010-05-08`
MetadataXML	XML file

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Content-Length: [...]

Action=RSCreateSAMLConnection&
Version=2010-05-08&
MetadataXML=xml
```

Example response

```
{
  "Success": "true",
  "ProviderUrl": "https://lyvespace-dev.us.auth0.com/login/callback?connection=resellerName-account-saml",
  "EntityId": "urn:lyvecloud:resellerName-account-saml",
  "Expiry": "05/07/2034",
  "Provider": "resellerName-account-saml"
}
```

RSGetSAMLConnection

Get SAML connection for Federated Login (SSO).

POST parameters

Parameter	Description
Action	Must be set to `RSGetSAMLConnection`

Parameter	Description
Version	Must be set to `2010-05-08`

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Content-Length: [...]

Action=RSGetSAMLConnection&
Version=2010-05-08&
```

Example response

```
{
  "ProviderUrl": "https://lyvespace-dev.us.auth0.com/login/callback?connection=resellerName-account-saml",
  "EntityId": "urn:lyvecloud:resellerName-account-saml",
  "Expiry": "05/07/2034",
  "Provider": "resellerName-account-saml"
}
```

RSUpdateSAMLConnection

Update SAML connection for Federated Login (SSO).

POST parameters

Parameter	Description
Action	Must be set to `RSUpdateSAMLConnection`
Version	Must be set to `2010-05-08`
MetadataXML	XML file

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Content-Length: [...]

Action=RSUpdateSAMLConnection&
Version=2010-05-08&
MetadataXML=xml
```

Example response

```
{
  "Success": "true",
  "ProviderUrl": "https://lyvespace-dev.us.auth0.com/login/callback?connection=resellerName-account-saml",
  "EntityId": "urn:lyvecloud:resellerName-account-saml",
  "Expiry": "05/07/2034",
  "Provider": "resellerName-account-saml"
}
```

RSDeleteSAMLConnection

Delete SAML connection for Federated Login (SSO).

POST parameters

Parameter	Description
Action	Must be set to `RSDeleteSAMLConnection`
Version	Must be set to `2010-05-08`

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
```

```
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Content-Length: [...]

Action=RSDeleteSAMLConnection&
Version=2010-05-08
```

Example response

```
{
  "Success": "true"
}
```

RSSetUserAuthMethod

Set user AuthMethod for login.

POST parameters

Parameter	Description
Action	Must be set to `RSSetUserAuthMethod`
Version	Must be set to `2010-05-08`
Name	UserName
AuthMethod	Must be set to `federated/password`

Example request

```
POST / HTTP/1.1
Host: iam.example.lyve.seagate.com
User-Agent: [...]
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Content-Length: [...]

Action=RSSetUserAuthMethod&
Version=2010-05-08&
Name=UserName&
AuthMethod=federated
```

Lyve S3

The Lyve S3 endpoint (https://s3.example.lyve.seagate.com) lets you perform operations on files and buckets. Requests to this endpoint must be signed with AWS Signature V4 or V2, specifying S3 as the service. Note that only path-style requests are supported.

Bucket: Retrieve Bucket List

`GET /`

Returns a list of all the buckets the current user has access to

Example response

```
<?xml version="1.0" encoding="UTF-8"?>
<ListAllMyBucketsResult xmlns="http://s3.amazonaws.com/doc/2006-03-01">
 <Buckets>
  <Bucket>
   <Name>private-bucket</Name>
   <CreationDate>2019-04-24T17:20:01.997Z</CreationDate>
  </Bucket>
  <Bucket>
   <Name>public-bucket</Name>
   <CreationDate>2019-04-24T17:18:59.552Z</CreationDate>
  </Bucket>
 </Buckets>
 <Owner>
  <ID>100000000013</ID>
  <DisplayName>example</DisplayName>
 </Owner>
</ListAllMyBucketsResult>
```

Bucket: Retrieve

`GET /` or `GET /?list-type=2`

GET Parameters

The following optional GET parameters can be specified:

Parameter	Description
list-type	When set to "2", specifies that the ListObjectsV2 format should be used
max-keys	Specifies a maximum number of keys to be returned. Defaults to the maximum value of 1000.
prefix	Filters the returned keys by prefix. Can be used to specify a directory.
continuation-token	Pagination token, which can be set to the NextContinuationToken element of the previous result page. (If list-type is not set, "marker" should be used instead.)
fetch-owner	Returns owner field with each key in the result
start-after	Space will start listing after this specified key. It can be any key in the bucket.

Note that, if specified, “/” is the only supported delimiter.

Response

On success, an XML document is returned, containing the first `max-keys` items of the requested listing. When the listing contains more than `max-keys` items, then the response contains the elements `<IsTruncated>true</IsTruncated>`, and a pagination token (either `<NextContinuationToken>` if `list-type=2`, or `NextMarker`) which can be used to retrieve the next page.

If the prefix parameter is specified and has a trailing slash, the contents of the listing are the files and folder in that directory. If the prefix parameter does not contain a slash, it is used to filter the contents of the parent folder. The listing is sorted in lexicographical order.

Example request

```
GET /abc-bucket/?list-type=2&delimiter=/&max-keys=300&prefix=test-a/
```

Example response (in ListObjectsV2 format)

```
<?xml version="1.0" encoding="UTF-8"?>
<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Name>abc-bucket</Name>
  <Prefix></Prefix>
  <ContinuationToken></ContinuationToken>
  <MaxKeys>300</MaxKeys>
  <KeyCount>4</KeyCount>
  <Delimiter>/</Delimiter>
  <IsTruncated>false</IsTruncated>
  <Contents>
    <Key>files.zip</Key>
    <LastModified>2019-02-23T12:41:11.000Z</LastModified>
    <Size>25617859</Size>
    <Owner>
      <ID>0</ID>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
  </Contents>
  <CommonPrefixes>
    <Prefix>example-directory/</Prefix>
  </CommonPrefixes>
  <CommonPrefixes>
    <Prefix>other-directory/</Prefix>
  </CommonPrefixes>
</ListBucketResult>
```

Bucket: Retrieve (Versioned)

Is similar to List Object on a non-versioned bucket but return all the versions for a specified prefix including all elements markerd as `DeleteMarker`. A delete marker is a marker that points to a specific object in the bucket and indicates that this object is deleted. Every deleted object has a specifi `VersionId`.

Response

On success, an XML document is returned, containing the first `max-keys` items of the requested listing. When the listing contains more than `max-keys` items, then the response contains the elements `<IsTruncated>true</IsTruncated>`. Every versions for the specified prefix is returned. The latest version is marked using the element `<IsLatest>true</IsLatest>`.

```
GET /abc-bucket/?prefix=myprefix&versions=
```

Example with aws client

```
aws s3api list-object-versions --bucket examplebucket --prefix myprefix
```

Example response


```
{
  "DeleteMarkers": [
    {
      "Owner": {
        "ID": "100000000001"
      },
      "IsLatest": true,
      "VersionId": "itdou5ddl1j558z4",
      "Key": "doc2.jpg",
      "LastModified": "2019-07-18T09:58:58.1718141Z"
    }
  ],
  "Versions": [
    {
      "LastModified": "2019-07-18T09:58:42.7619115Z",
      "VersionId": "s343opv4r1hv2vx3",
      "ETag": "\"324f6cbcfdf2420dd63890234a9f2f14\"",
      "StorageClass": "STANDARD",
      "Key": "doc2.jpg",
      "Owner": {
        "ID": "100000000001"
      },
      "IsLatest": true,
      "Size": 149951
    },
    {
      "LastModified": "2019-07-18T09:58:28.3141363Z",
      "VersionId": "scmsdt1jl0395uby",
      "ETag": "\"324f6cbcfdf2420dd63890234a9f2f14\"",
      "StorageClass": "STANDARD",
      "Key": "doc1.jpg",
      "Owner": {
        "ID": "100000000001"
      },
      "IsLatest": true,
      "Size": 149951
    },
    {
      "LastModified": "2019-07-18T09:58:12.0506257Z",
      "VersionId": "536znyl9hzlgczh3",
      "ETag": "\"f3658d735ac33d68a23909b1d2583421\"",
      "StorageClass": "STANDARD",
      "Key": "doc3.jpg",
      "Owner": {
        "ID": "100000000001"
      },
      "IsLatest": true,
```

```
      "Size": 578504
    }
  ]
}
```


Bucket: Add

`PUT /mybucket`

Creates a new bucket `mybucket`. Anonymous requests are not allowed to create buckets. The additional `replication-policy` parameter can be used to specify the list of regions where to replicate the bucket. If not specified, buckets are replicated in every region.

PUT Parameters

The following optional PUT parameters can be specified:

Form Parameter	Description
replication-policy	Specify a comma separated list of regions identifiers (Eg. DCA02,DEN02,SJC03)

Example response


```
HTTP/1.1 200 OK
Date: Mon, 22 Jul 2019 09:48:09 GMT
Content-Length: 0
```


Bucket: Delete

`DELETE /mybucket`

Deletes the bucket `mybucket`. Anonymous requests are not allowed to delete buckets.

Response

Returns 200 in case of success, 404 in case the bucket does not exists, or 409 conflict error in case bucket

is not empty.

Example response

```
HTTP/1.1 200 OK
Date: Mon, 22 Jul 2019 09:48:09 GMT
Content-Length: 0
```

Bucket CORS: Retrieve

Returns the cors configuration information set for the bucket.

To use this operation, you must have permission to perform the `s3:GetBucketCORS` action. By default, the bucket owner has this permission and can grant it to others.

Example request

```
GET /?cors HTTP/1.1
Host: Bucket.s3.lyve.seagate.com
```

Example response

```
HTTP/1.1 200
Content-Type: application/xml

<?xml version="1.0" encoding="UTF-8"?>
<CORSConfiguration>
  <CORSRule>
    <AllowedHeader>string</AllowedHeader>
    ...
    <AllowedMethod>string</AllowedMethod>
    ...
    <AllowedOrigin>string</AllowedOrigin>
    ...
    <ExposeHeader>string</ExposeHeader>
    ...
    <MaxAgeSeconds>integer</MaxAgeSeconds>
  </CORSRule>
  ...
</CORSConfiguration>
```

Bucket CORS: Set

Sets the cors configuration for your bucket. If the configuration exists, it replaces it.

To use this operation, you must be allowed to perform the `s3:PutBucketCORS` action. By default, the bucket owner has this permission and can grant it to others.

You set this configuration on a bucket so that the bucket can service cross-origin requests. For example, you might want to enable a request whose origin is `http://www.example.com` to access your Lyve bucket at `my.example.bucket.com` by using the browser's XMLHttpRequest capability.

To enable cross-origin resource sharing (CORS) on a bucket, you add the `cors` subresource to the bucket. The `cors` subresource is an XML document in which you configure rules that identify origins and the HTTP methods that can be executed on your bucket. The document is limited to 64 KB in size.

When Lyve Cloud receives a cross-origin request (or a pre-flight OPTIONS request) against a bucket, it evaluates the `cors` configuration on the bucket and uses the first CORSRule rule that matches the incoming browser request to enable a cross-origin request. For a rule to match, the following conditions must be met:

- The request's `Origin` header must match `AllowedOrigin` elements.
- The request method (for example, GET, PUT, HEAD, and so on) or the `Access-Control-Request-Method` header in case of a pre-flight OPTIONS request must be one of the `AllowedMethod` elements.
- Every header specified in the `Access-Control-Request-Headers` request header of a pre-flight request must match an `AllowedHeader` element.

In `CORS Configuration example #1` the first `CORSRule` allows cross-origin PUT, POST, and DELETE requests whose origin is `http://www.example.com` origins. The rule also allows all headers in a pre-flight OPTIONS request through the `Access-Control-Request-Headers` header. Therefore, in response to any pre-flight OPTIONS request, Lyve S3 will return any requested headers. The second rule allows cross-origin GET requests from all the origins. The '*' wildcard character refers to all origins.

In `CORS Configuration example #2`, the single `CORSRule` includes the following additional optional parameters:

- `MaxAgeSeconds` - Specifies the time in seconds that the browser will cache a Lyve Cloud response to a pre-flight OPTIONS request for the specified resource. In this example, this parameter is 3000 seconds. Caching enables the browsers to avoid sending pre-flight OPTIONS request to Lyve Cloud for repeated requests.
- `ExposeHeader` - Identifies the response header (in this case `x-lyve-example`) that you want customers to be able to access from their applications (for example, from a JavaScript XMLHttpRequest object).

PUT Bucket CORS Example request #1

```
PUT / ?cors HTTP/1.1
Host: Bucket.s3.amazonaws.com
Content-MD5: ContentMD5

<?xml version="1.0" encoding="UTF-8"?>
<CORSConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <CORSRule>
    <AllowedHeader>string</AllowedHeader>
    ...
    <AllowedMethod>string</AllowedMethod>
    ...
    <AllowedOrigin>string</AllowedOrigin>
    ...
    <ExposeHeader>string</ExposeHeader>
    ...
    <MaxAgeSeconds>integer</MaxAgeSeconds>
  </CORSRule>
  ...
</CORSConfiguration>
```

CORS Configuration example #1

```
<CORSConfiguration>
 <CORSRule>
   <AllowedOrigin>http://www.example.com</AllowedOrigin>
   <AllowedMethod>PUT</AllowedMethod>
   <AllowedMethod>POST</AllowedMethod>
   <AllowedMethod>DELETE</AllowedMethod>
   <AllowedHeader>*</AllowedHeader>
 </CORSRule>
 <CORSRule>
   <AllowedOrigin>*</AllowedOrigin>
   <AllowedMethod>GET</AllowedMethod>
 </CORSRule>
</CORSConfiguration>
```

Bucket CORS: Delete

If the action is successful, the service sends back an HTTP 204 response with an empty HTTP body.

Deletes the cors configuration information set for the bucket.

To use this operation, you must have permission to perform the `s3:PutBucketCORS` action. The bucket owner has this permission by default and can grant this permission to others.

Example request

```
DELETE /?cors HTTP/1.1
Host: Bucket.s3.lyve.seagate.com
```

Example response

```
HTTP/1.1 204
```

Bucket Encryption: Retrieve

Returns the default encryption configuration for a bucket.

> **Note**—By default, all buckets have a default encryption configuration that uses server-side encryption with Lyve-managed keys (SSE-S3).

Request

```
GET /<bucket>/?encryption
```

Request body

The request does not have a request body.

Response

```
HTTP/1.1 200
<?xml version="1.0" encoding="UTF-8"?>
<ServerSideEncryptionConfiguration>
  <Rule>
    <ApplyServerSideEncryptionByDefault>
```

```
        <SSEAlgorithm>string</SSEAlgorithm>
      </ApplyServerSideEncryptionByDefault>
    </Rule>
  </ServerSideEncryptionConfiguration>
```

Example with AWS CLI

```
aws s3api get-bucket-encryption --bucket <bucket-name>
```

Example response from AWS CLI

```
{
    "ServerSideEncryptionConfiguration": {
        "Rules": [
            {
                "ApplyServerSideEncryptionByDefault": {
                    "SSEAlgorithm": "AES256"
                }
            }
        ]
    }
}
```

Note

The following operations are not currently supported:

- PutBucketEncryption
- DeleteBucketEncryption

Permissions

The `s3:GetEncryptionConfiguration` permission is required. To call this API, this permission must be explicitly granted to the user, or the user must be root user.

Bucket Lifecycle: Retrieve

`GET /<bucket>/?lifecycle`

Return, if present, the current lifecycle configuration for a bucket.

Note: lifecycle configuration returned may differ from the PUTed one, as it may be enriched with some more informations.

Example request using the shell

```
aws s3api get-bucket-lifecycle-configuration --bucket examplebucket
```

Example request using http

```
GET /examplebucket/?lifecycle HTTP/1.1
Host: s3.lyve.seagate.com
x-amz-date: Thu, 15 Nov 2012 00:17:21 GMT
Authorization: signatureValue
```

Example response

```
HTTP/1.1 200 OK
x-amz-id-2: ITnGT1y4RyTmXa3rPi4hklTXouTf0hccUjo0iCPjz6FnfIutBj3M7fPGlWO2SEWp
x-amz-request-id: 51991C342C575321
Date: Thu, 15 Nov 2012 00:17:23 GMT
Server: AmazonS3
Content-Length: 358

<?xml version="1.0" encoding="UTF-8"?>
<LifecycleConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Rule>
    <ID>Archive and then delete rule</ID>
    <Filter>
      <Prefix>projectdocs/</Prefix>
    </Filter>
    <Status>Enabled</Status>
    <Expiration>
      <Days>3650</Days>
    </Expiration>
  </Rule>
</LifecycleConfiguration>
```

Bucket Lifecycle: Set

`PUT /<bucket>/?lifecycle`

Store a Lifecycle configuration for a given bucket. A lifecycle configuration is a set of rules that describes how lifecycle management for a specific object is perfomed. The following kind of rules are supported:

- Expiration
- NoncurrentVersionExpiration (applies only to buckets with versioning enabled or suspended)
- ExpiredObjectDeleteMarker (applies only to buckets with versioning enabled)
- AbortIncompleteMultipartUpload

Note

Payload examples

Response

```
HTTP/1.1 200 OK
x-amz-id-2: aXQ+KbIrmMmoO//3bMdDTw/CnjArwje+J49Hf+j44yRb/VmbIkgIO5A+PT98Cp/6k07hf+LD2mY=
x-amz-request-id: 02D7EC4C10381EB1
Date: Wed, 14 May 2014 02:21:50 GMT
Content-Length: 0
Server: AmazonS3
```

Note:

- `Transitions` action is not supported
- `Tag` is not supported in filter conditions
- `And` is not supported in filter conditions
- `Date` must be in "YYYY-MM-DDT00:00:00Z" format

Objects Expiration

Note that RSTOR Space does not currently return the object expiration date in the `x-amz-expiration` header. To compute the object expiration date for an object, you can proceed as follows:

- fetch all bucket lifecycle policies with `GET /<bucket>/?lifecycle`
- find the Expiration policy with the lowest `Days` field among the ones whose Filter applies to the current object
- sum the creation date of the object taken from the `Last-Modified` header with the `Days` indicated in the Expiration policy
- in case Expiration policies with `Date` field are being used, find lowest between al `Date` fields whose Filter applies to the current object

Example with aws client

```
aws s3api put-bucket-lifecycle-configuration --bucket examplebucket --lifecycle-configuration file://lifecycle.json
```

Payload examples

```
{
  "Rules": [
    {
     "ID": "Delete logs",
     "Status": "Enabled",
     "Filter": {
       "Prefix": "myapp_"
     },
     "Expiration": {
       "Days": 10
     },
    },
    {
     "Status": "Enabled",
     "Filter": {},
     "Expiration": {
       "ExpiredObjectDeleteMarker": true
     },
    },
    {
     "ID": "Delete aborted multipart",
     "Status": "Enabled",
     "Filter": {},
     "AbortIncompleteMultipartUpload": {
       "DaysAfterInitiation": 7
     },
    },
    {
     "Status": "Disabled",
     "Filter": {
       "Prefix": "logs/"
     },
     "NoncurrentVersionExpiration": {
       "NoncurrentDays": 2
     },
     "ID": "Delete old versions"
    }
  ]
}
```

Example response

```
HTTP/1.1 200 OK
x-amz-id-2: aXQ+KbIrmMmoO//3bMdDTw/CnjArwje+J49Hf+j44yRb/VmbIkgIO5A+PT98Cp/6k07hf+LD2mY=
x-amz-request-id: 02D7EC4C10381EB1
Date: Wed, 14 May 2014 02:21:50 GMT
Content-Length: 0
Server: AmazonS3
```

Bucket Lifecycle: Delete

`DELETE /<bucket>/?lifecycle`

Delete, if present, a lifecycle configuration for a bucket.

Example request using the shell

```
aws s3api delete-bucket-lifecycle-configuration --bucket examplebucket
```

Example request using http

```
DELETE /examplebucket/?lifecycle HTTP/1.1
Host: s3.lyve.seagate.com
Date: date
Authorization: authorization string
```

Example response

```
HTTP/1.1 204 No Content
x-amz-id-2: Uuag1LuByRx9e6j5OnimrSAMPLEtRPfTaOAa==
x-amz-request-id: 656c76696e672SAMPLE5657374
Date: Wed, 14 Dec 2011 05:37:16 GMT
Connection: keep-alive
```

Bucket Logging: Retrieve

Retrieve a bucket logging configuration.

Example request

```
GET /mybucketlog?logging HTTP/1.1
Host: 127.0.0.1:32005
Accept-Encoding: identity
User-Agent: aws-cli/1.16.287 Python/3.7.3 Linux/4.19.0-8-amd64 botocore/1.13.23
Content-MD5: NKhuLiQuQT2Icj+AzLFcTQ==
X-Amz-Date: 20200212T203404Z
X-Amz-Content-SHA256: 265e84e2b334e5566fcacffc9277caf252adc4cd4ff835bbc02df1266e116085
Content-Length: 0
```

Example response

```
<?xml version="1.0" encoding="UTF-8"?>
<BucketLoggingStatus>
<LoggingEnabled>
   <TargetBucket>targetlog</TargetBucket>
   <TargetPrefix>logs/</TargetPrefix>
   <TargetGrants>
      <Grant>
      <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="AccessKey">
         <ID>STX_KEY_123</ID>
      </Grantee>
      </Grant>
   </TargetGrants>
</LoggingEnabled>
</BucketLoggingStatus>
```

Bucket Logging: Set

Set the logging configuration for a bucket and to specify. All logs are saved to the specified target bucket using the provided access key. To set the logging status of a bucket, you must be the bucket owner.

To delete a Bucket Logging configuration you can make a PUT passing an empty LoggingEnabledelement.

`TargetBucket`: is the destination bucket (where the logs will be uploaded`TargetPrefix`: is the prefix of the

file contains the log that will be uploaded in the TargetBucket

Then you should compose the `TargetGrants`. The `TargetGrants` must contains the `Grantee` element with `xsi:type="AccessKey"` and as a child an `ID` element contains the AccessKey used to perform the upload.

Log format

Field	Description
BucketOwner	The owner id of the monitored bucket
Bucket	The monitored bucket
Ts	The timestamp of the action [06/Feb/2019:00:00:38 +0000]
RemoteIp	The apparent IP of the requester. Intermediate proxies and firewalls might obscure the actual address of the machine making the request
Requester	The AccessKey used to perform the action
RequestId	The Request ID
Operation	The kind of operarion (ex.: s3.PuObject, s3.GetObject)
Key	The object Key (if present)
RequestUri	The Request-URI part of the HTTP request message
HttpStatus	The numeric HTTP status code of the response
ErrorCode	The S3 Error Code, or "-" if no error occurred
BytesSent	The transferred bytes
ObjectSize	The size of the object transferred
TotalTime	The number of milliseconds the request was in flight from the server's perspective
TurnAroundTime	The number of milliseconds spent processing the request
Referer	The value of the HTTP Referer header, if present

Field	Description
UserAgent	The value of the HTTP User-Agent header
VersionId	The version ID in the request (if present)
HostId	Not used (is always "-")
SignatureVersion	The signature version, "SigV2" or "SigV4", that was used to authenticate the request or a "-" in the other case
CipherSuite	The value is SSL if the session was encrypted
Auth	The type of request authentication used, "AuthHeader" for authentication headers, "QueryString" or "-" for the other cases
HostHeader	Not used (is always "-")
TlsVer	The TLS version used

Example request

```
PUT /mybucketlog?logging HTTP/1.1
Host: 127.0.0.1:32005
Accept-Encoding: identity
User-Agent: aws-cli/1.16.287 Python/3.7.3 Linux/4.19.0-8-amd64 botocore/1.13.23
Content-MD5: NKhuLiQuQT2Icj+AzLFcTQ==
X-Amz-Date: 20200212T203404Z
X-Amz-Content-SHA256: 265e84e2b334e5566fcacffc9277caf252adc4cd4ff835bbc02df1266e116085
Content-Length: 399

<BucketLoggingStatus xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <LoggingEnabled>
    <TargetBucket>targetlog</TargetBucket>
    <TargetPrefix>logs/</TargetPrefix>
    <TargetGrants>
      <Grant>
        <Grantee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="AccessKey">
          <ID>AWS4XZUHH4DVBPMSZB7ODUFVVHGUXMXLY4VRJRJ4BFHJ7CGJACI4LPLA</ID>
        </Grantee>
      </Grant>
    </TargetGrants>
  </LoggingEnabled>
</BucketLoggingStatus>
```

Example response

```
HTTP/1.1 200 OK
Date: [...]
Content-Length: 0
```

Examples of log generated

```
100000000001 testlog [12/Feb/2020:20:28:02 +0100] "[::1]:34178" "AWS4XZUHH4DVBPMSZB7ODUFVVHGUXMXLY4VRJRJ4BFHJ7CGJACI4LPLA" "qfqun6v6dido" s3:PutObject "Everest" "/testlog/Everest" "200" "-" 125829120 "0" 279 274 "" "aws-sdk-go/1.28.13 (go1.13.7; linux; amd64)" 01E0XDYR0H0DA24AYE1DV1PK5S - SigV4 SSL AuthHeader - "-"
100000000001 testlog [12/Feb/2020:20:28:02 +0100] "[::1]:34178" "AWS4XZUHH4DVBPMSZB7ODUFVVHGUXMXLY4VRJRJ4BFHJ7CGJACI4LPLA" "50gzg1qj4ftt" s3:HeadObject "Everest" "/testlog/Everest" "200" "-" 125829120 125829120 1 0 "" "aws-sdk-go/1.28.13 (go1.13.7; linux; amd64)" 01E0XDYR0H0DA24AYE1DV1PK5S - SigV4 SSL AuthHeader - "-"
100000000001 testlog [12/Feb/2020:20:28:02 +0100] "[::1]:34178" "AWS4XZUHH4DVBPMSZB7ODUFVVHGUXMXLY4VRJRJ4BFHJ7CGJACI4LPLA" "eybjzx8hvf5s" s3:DeleteObject "Everest" "/testlog/Everest" "200" "-" "0" "0" 2 0 "" "aws-sdk-go/1.28.13 (go1.13.7; linux; amd64)" "-" - SigV4 SSL AuthHeader - "-"
100000000001 testlog [12/Feb/2020:20:28:02 +0100] "[::1]:34178" "AWS4XZUHH4DVBPMSZB7ODUFVVHGUXMXLY4VRJRJ4BFHJ7CGJACI4LPLA" "69srbt1hije2" s3:PutObject "foo" "/testlog/foo" "200" "-" 20 "0" 3 1 "" "aws-sdk-go/1.28.13 (go1.13.7; linux; amd64)" 01E0XDYR0TABEJXCHQ300Z8BHE - SigV4 SSL AuthHeader - "-"
```

Bucket Metadata: Retrieve

`HEAD /<bucket>`

Returns 200 if the bucket exists, 404 otherwise. If available, the `X-Rstor-Size` header is returned, reporting the current size of the bucket in bytes.

Example response

```
HTTP/2.0 200 OK
Connection: close
Date: Tue, 30 Apr 2019 09:24:41 GMT
X-Lyve-Size: 35342254
```

Bucket Object Lock: Retrieve

Retrieve the Object Lock configuration for a bucket. The rule specified in this Object Lock configuration will be applied by default to every new object placed in the specified bucket.

> The related IAM permission is not s3:GetObjectLockConfiguration but rather s3:GetBucketObjectLockConfiguration.

Example request

```
GET /?object-lock HTTP/1.1
Host: bucketName.s3.lyve.seagate.com
```

Example response

```
HTTP/1.1 200 OK

<?xml version="1.0" encoding="UTF-8"?>
<ObjectLockConfiguration>
  <ObjectLockEnabled>Enabled</ObjectLockEnabled>
  <Rule>
    <DefaultRetention>
      <Days>15</Days>
    </DefaultRetention>
  </Rule>
</ObjectLockConfiguration>
```

Bucket Object Lock: Set

Places an Object Lock configuration on the specified bucket. The rules specified in the Object Lock configuration will be applied by default to every new object placed in the specified bucket.

> You cannot apply an object lock configuration to a bucket if it wasn't created with the S3-Locking activated - see CreateBucket for more information. Object lock configuration cannot be activated on existing buckets.

The ObjectLockEnabled field is optional and indicates whether the bucket has an Object Lock configuration enabled (valid values: Enabled).

> `DefaultRetention` requires either Days or Years. You can't specify both at the same time.

Example request

```
PUT /?object-lock HTTP/1.1
Host: bucketName.s3.lyve.seagate.com

<?xml version="1.0" encoding="UTF-8">
<ObjectLockConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <ObjectLockEnabled>string</ObjectLockEnabled>
  <Rule>
    <DefaultRetention>
      <Days>integer</Days>
      <Mode>string</Mode>
      <Years>integer</Years>
    </DefaultRetention>
  </Rule>
</ObjectLockConfiguration>
```

Example response

```
HTTP/1.1 200 OK
```

Bucket Policy: Retrieve

`GET /<bucket>?policy`

Response

Returns the current bucket policy. If no bucket policy is set, a 404 error is returned.

Bucket Policy: Retrieve Status

> `GetBucketPolicyStatus` is currently unsupported.

Bucket Policy: Set

`PUT /<bucket>?policy`

Sets the access policy for a given bucket. Currently, we only support using bucket policies to make a bucket publicly accessible.

In the example, a policy allowing anonymous GetObject requests is assigned to the bucket `abc-bucket`.

Response

Returns a status code indicating whether the operation was successful.

Example request body

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
     "Action": ["s3:GetBucketLocation"],
     "Effect": "Allow",
     "Principal": { "AWS": ["*"] },
     "Resource": ["arn:aws:s3:::abc-bucket"],
     "Sid": ""
    },
    {
     "Action": ["s3:GetObject"],
     "Effect": "Allow",
     "Principal": { "AWS": ["*"] },
     "Resource": ["arn:aws:s3:::abc-bucket/*"],
     "Sid": ""
    }
  ]
}
```

Bucket Policy: Delete

`DELETE /<bucket>?policy`

Removes the access policy set for a given bucket.

Response

Returns a status code indicating whether the operation was successful.

Bucket RS Info: Retrieve

Example request

```
GET /examplebucket?rs-info= HTTP/1.1
Host: 127.0.0.1:44355
User-Agent: Go-http-client/1.1
Authorization: AWS4-HMAC-SHA256 Credential=STX1ARO9Y8B0272GA5LAERGU/20190722/any/s3/aws4_request, SignedHeaders=host;x-amz-content-sha256;x-amz-date, Signature=000e8c03d6afe31c96bb616a8f67e29b7cf1308ab196adb2fa757d27750b66aa
X-Amz-Content-Sha256: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
X-Amz-Date: 20190722T110035Z
Accept-Encoding: gzip
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json
Date: Mon, 22 Jul 2019 11:00:35 GMT
Content-Length: 73

{
  "bucketSize": 57300,
  "replicationPolicy": [
    "DEN02",
    "SJC03"
  ],
  "isPublic": false
}
```

Bucket RS Stats: Retrieve

Serves per-bucket storage metrics.

Example request

```
GET /?rs-bucket-stats= HTTP/1.1
Host: s3.example.lyve.seagate.com
Authorization: [...]
```

Example response

```
{
  "abc-images": {
  "size": 6304387203,
  "objects": 74854,
  "replicationPolicy": [
    "DCA02",
    "DEN02",
    "SJC03"
  ]
},
  "abc-backups": {
  "size": 729808896,
  "objects": 229,
  "replicationPolicy": [
    "DCA02",
    "DEN02",
    "SJC03"
  ]
}
}
```


Bucket Tags: Retrieve

Return, if present, the tagging of a bucket.

Example request

```
GET /bucket?tagging HTTP/1.1
Host: […]
Authorization: […]
X-Amz-Content-SHA256: […]
X-Amz-Date: […]
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Date: […]
Content-Length: 184

<?xml version="1.0" encoding="UTF-8"?>
<Tagging xmlns="">
```

```
  <TagSet>
    <Tag>
      <Key>KEYTEXT</Key>
      <Value>VALUETEXT</Value>
    </Tag>
    <Tag>
      <Key>Type</Key>
      <Value>Script</Value>
    </Tag>
  </TagSet>
</Tagging>
```

Bucket Tags: Add or Replace

Add tagging to a bucket. Overwrites tags if existing, to update tagging first get the old tagging, modify them locally and upload the new tagging.

Tagging are in the form of key-value pair: `Format=Image`

At most 10 tags per bucket can be specified. The length of a tag key can be up to to 128 unicode characters long. The length of a tag value can be up to to 256 unicode characters long.

Example request

```
PUT /bucket?tagging HTTP/1.1
Host: […]
Authorization: […]
X-Amz-Content-SHA256: […]
X-Amz-Date: […]

<Tagging xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <TagSet>
    <Tag>
      <Key>KEYTEXT</Key>
      <Value>VALUETEXT</Value>
    </Tag>
    <Tag>
      <Key>Type</Key>
      <Value>Script</Value>
    </Tag>
  </TagSet>
</Tagging>
```

Example response

```
HTTP/1.1 200 OK
Date: [...]
Content-Length: 0
```

Bucket Tags: Delete

Delete, if present, the tagging of a bucket.

Example request

```
DELETE /bucket?tagging HTTP/1.1
Host: [...]
Authorization: [...]
X-Amz-Content-SHA256: [...]
X-Amz-Date: [...]
Content-Length: 0
```

Example response

```
HTTP/1.1 204 No Content
x-amz-version-id: [...]
Date: [...]
```

Bucket Versioning: Retrieve

`GET /<bucket>/?versioning`

Return the status of versioning for the given bucket

Response

The possible value for status are:

- "Enabled"

- "Suspended"

Example with aws client

```
aws s3api get-bucket-versioning --bucket test
```

Example response

```
{
  "Status": "Enabled"
}
```

Bucket Versioning: Set

Example with aws client to Enable

```
aws s3api put-bucket-versioning --bucket test --versioning-configuration "Status=Enabled"
```

Example with aws client to Suspend

```
aws s3api put-bucket-versioning --bucket test --versioning-configuration "Status=Suspended"
```

Example xml request

```
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>[Enabled | Suspended]</Status>
</VersioningConfiguration>
```

Object: Retrieve

`GET /<bucket>/<object>` or `GET /<bucket>/<object>?versionId=<versionId>`

Retrieve an object in a bucket. In case the bucket has versioning enabled we can specify th`versionId`. If the current version of the object is a delete marker, the object was deleted, the flag x-amz-delete-marker: true is in included in the response. The API provides the replication status in the`X-Rstor-Replication-Status` header as a list of regions.

Header parameters

Header	Description
Range	Specifies a byte range to be retrieved, in the format `bytes:<start>-<end>`
x-amz-server-side-encryption-customer-algorithm	SSE-C algorithm (supported: AES256)
x-amz-server-side-encryption-customer-key	SSE-C base64-encoded decryption key
x-amz-server-side-encryption-customer-key-MD5	SSE-C base64-encoded key hash (MD5)
x-rstor-replication-status	If specified, the replication status is returned back as a list of regions
x-amz-checksum-mode	If specified, the checksum mode will be returned on headers, valid value: `ENABLED`

Query parameters

The following query parameters can be used to set or override the corresponding headers in the response the server sends back.

Parameter	Description
response-content-type	Sets the "Content-Type" header on the response
response-content-language	Sets the "Content-Language" header on the response
response-expires	Sets the "Expires" header on the response
response-cache-control	Sets the "Cache-Control" header on the response
response-content-disposition	Sets the "Cache-Disposition" header on the response
response-content-encoding	Sets the "Content-Encoding" header on the response

Response

On success, a status code of 200 is returned. An error code is returned otherwise. In case this operation is performed in a bucket that has the versioning enabled the corresponding `VersionId` is returned. Moreover, in case the bucket has one or more lifecycle Expiration rules that apply to newly created object, the corresponding `Expiration` is returned. Finally, if object was uploaded with SSE-S3 or SS3-C encryption, `ServerSideEncryption` is returned.

Example response

```
HTTP/1.1 200 OK
content-length: 25617859
content-type: binary/octet-stream
date: Mon, 25 Feb 2019 10:49:59 GMT
etag: "918a4f4a76bfa69c193fd4365bc12622"
last-modified: Mon, 25 Feb 2019 10:49:59 GMT
X-Lyve-Replication-Status: DEN02,SJC03

Object contents
```

Object: Add or Replace

`PUT /<bucket>/<object>`

Sets the contents of an object to the contents of the request's body, creating the object if it does not exists. If the `x-amz-copy-source` header is specified, the object is copied from the one specified in the header, and the body of the PUT request is ignored.

Object keys must be valid UTF8 strings and must not end with a trailing slash. However, a request with an empty body can specify a key with a trailing slash, in this case an empty directory is created.

Header parameters

The PutObject API supports the following optional headers

Header	Description
Content-Md5	The MD5 hash of the content. If present, it is checked as an additional guarantee that the object was correctly received
x-amz-copy-source	The object specified in the header is copied to the target key
x-amz-tagging	The tag-set to assign to the uploaded object, formatted as a query string (for example, "Key1=Value1")
x-amz-tagging-directive	Specifies whether the object tag-set are copied from the source object or replaced with tag-set provided in the request. Valid Values: `COPY` or `REPLACE`
x-amz-server-side-encryption	SSE-S3 Algorithm (supported: AES256)
x-amz-server-side-encryption-customer-algorithm	SSE-C algorithm (supported: AES256)

Header	Description
x-amz-server-side-encryption-customer-key	SSE-C base64-encoded encryption key
x-amz-server-side-encryption-customer-key-MD5	SSE-C base64-encoded encryption key hash (MD5)
x-amz-copy-source-server-side-encryption-customer-algorithm	SSE-C algorithm of the source object (supported: AES256)
x-amz-copy-source-server-side-encryption-customer-key	SSE-C base64-encoded decryption key of the source object
x-amz-copy-source-server-side-encryption-customer-key-MD5	SSE-C base64-encoded decryption key hash (MD5) of the source object
x-amz-checksum-algorithm or x-amz-sdk-checksum-algorithm	Custom checksum algorithm (valid values: `CRC32`, `CRC32C`, `SHA1`, `SHA256`), stored checksum can be retrieved by calling GetObject
x-amz-checksum	Optional base64-encoded checksum for custom checksum algorithm

Header	Description
x-amz-trailer	Optional compatibility header, may contain value: `x-amz-checksum-crc32`, `x-amz-checksum-crc32c`, `x-amz-checksum-sha1`, or `x-amz-checksum-sha256`
x-amz-checksum-crc32	CRC32 checksum of the object (base64-encoded)
x-amz-checksum-crc32c	CRC32C checksum of the object (base64-encoded)
x-amz-checksum-sha1	SHA1 checksum of the object (base64-encoded)
x-amz-checksum-sha256	SHA256 checksum of the object (base64-encoded)

If the checksum is calculated on the client side, it should be sent to the server in th `x-amz-checksum-crc32`, `x-amz-checksum-crc32c`, `x-amz-checksum-sha1`, or `x-amz-checksum-sha256` headers or a combination of `x-amz-checksum` and `x-amz-checksum-algorithm` or `x-amz-sdk-checksum-algorithm`. The server will verify the checksum, and will return an error if the checksums do not match.

Example valid base64-encoded checksum values for `Hello world\n123\n` " without quotes

- CRC32: `uWvPIg==`
- CRC32C: `Cy8XOQ==`
- SHA1: `LupGMeUw441P/33BhJlOZVSBpVg=`
- SHA256: `uzbBRoYAgN7yiuoYiZFk6kfOPcFad8E8uxFLXfuKVsA=`

If specified, the value of the following headers will be stored alongside the object, and then returned on the HeadObject or GetObject APIs:

- Cache-Control
- Content-Encoding
- Content-Disposition
- Content-Language
- Content-Md5
- Content-Type
- Expires

Custom metadata can be specified by means of headers with the `x-amz-meta-` prefix, up to a total of 2KB of data (counting both header names and values).

> **i** Copy operations can potentially take several seconds. To prevent the request from timing out, the server sends whitespace during the response. As a consequence, the outcome of the operation can either be specified as HTTP header or in the returned xml document.

Response

On success, a status code of 200 is returned. An error code is returned otherwise. In case this operation is performed in a bucket that has the versioning enabled the corresponding `VersionId` is returned. Moreover, in case the bucket has one or more lifecycle expiration rules that apply to newly created object, the corresponding `Expiration` is returned. Finally, if object was uploaded with SSE-S3 or SS3-C encryption, `ServerSideEncryption` is returned.

Example on a NON-versioned bucket

```
{
  "ETag": "\"324f6cbcfdf2420dd63890234a9f2f14\""
}
```

Example on a versioned bucket

```
{
  "VersionId": "khd1pmwdgewhlnd2",
  "ETag": "\"324f6cbcfdf2420dd63890234a9f2f14\""
}
```

Example: How to do upload and download using AWS-SDK-Go-v2

```
package main

import (
    "context"
    _ "embed"
    "encoding/json"
    "strings"
    "log"
```

```
    "github.com/aws/aws-sdk-go-v2/aws"
    "github.com/aws/aws-sdk-go-v2/aws/middleware"
    "github.com/aws/aws-sdk-go-v2/config"
    "github.com/aws/aws-sdk-go-v2/credentials"
    "github.com/aws/aws-sdk-go-v2/service/s3"
    "github.com/aws/aws-sdk-go-v2/service/s3/types"
    "github.com/aws/smithy-go/transport/http"
)

// the secret file will be embedded on compile time
// the file should contain something like this:
/*
 "url":"https://BUCKET.s3.RESELLER.lyve.seagate.com",
 "accessKey": "STX...",
 "secretKey": "..."
*/
// REGION: sjc03, dca02, den02, lon01, lon02, toy01, osa01
// BUCKET: bucket name

//go:embed secret.json
var secret []byte

type Cfg struct {
    Url       string `json:"url"`
    AccessKey string `json:"accessKey"`
    SecretKey string `json:"secretKey"`
}

// helper function for exit early CLI and print stack trace
// do not use for long-running service
func panicIf(err error, str string) {
    if err != nil {
        log.Println(str)
        panic(err)
    }
}

func main() {
    // load secrets from configuration
    var jsonCfg Cfg
    err := json.Unmarshal(secret, &jsonCfg)
    panicIf(err, `secret.json parsing failed`)

    // file name to upload
    key := aws.String(`FILENAME`)
    // example content of the file to be uploaded
    const helloWorld = "Hello world\n123\n"

    // retrieve bucket from config
    dotPos := strings.Index(jsonCfg.Url,".")
    bucketName := jsonCfg.Url[8:dotPos] // "https://" have length of 8
    log.Println(`bucket name:`, bucketName)
    bucket := aws.String(bucketName)
```

```
usingPF := strings.Contains(jsonCfg.Url, `lyve.seagate.com`) ||
    strings.Contains(jsonCfg.Url, `lyve-storage.com`)

// connection configuration
ctx := context.Background()
opts := [](func(options *config.LoadOptions) error){}
opts = append(opts, config.WithRegion(region))
cred := credentials.NewStaticCredentialsProvider(jsonCfg.AccessKey, jsonCfg.SecretKey, "")
opts = append(opts, config.WithCredentialsProvider(cred))
var awsCfg aws.Config
if usingPF { // override AWS default URL
    resolver := aws.EndpointResolverWithOptionsFunc(func(service, region string, options ...any) (aws.Endpoint, error) {
        return aws.Endpoint{
            SigningRegion:     region,
            URL:               jsonCfg.Url,
            HostnameImmutable: true,
        }, nil
    })
    opts = append(opts, config.WithEndpointResolverWithOptions(resolver))
}
awsCfg, err = config.LoadDefaultConfig(ctx, opts...)
panicIf(err, `failed config.LoadDefaultConfig`)

// create s3 client
s3Client := s3.NewFromConfig(awsCfg)

// create bucket
_, err := s3Client.CreateBucket(ctx, &s3.CreateBucketInput{
    Bucket: bucket,
})
if err != nil {
    log.Printf(`s3Client.CreateBucket failed %s: %s`, *bucket, err)
} else {
    log.Printf(`bucket %v created`, *bucket)
}

{
    log.Println(`PutObject start`)
    out, err := s3Client.PutObject(ctx, &s3.PutObjectInput{
        Bucket:           bucket,
        Key:              key,
        Body:             strings.NewReader(helloWorld),
        // enable one of these to do checksum verification:
        //ChecksumCRC32: aws.String(`uWvPlg==`),
        //ChecksumCRC32C: aws.String(`Cy8XOQ==`),
        //ChecksumSHA1: aws.String(`LupGMeUw441P/33BhJlOZVSBpVg=`),
        //ChecksumSHA256: aws.String(`uzbBRoYAgN7yiuoYiZFk6kfOPcFad8E8uxFLXfuKVsA=`),
    })
    panicIf(err, `failed s3Client.PutObject`)
    log.Println(`PutObject success`)
    log.Println(middleware.GetRawResponse(out.ResultMetadata).(*http.Response).Header)
}

{
```

```
        log.Println(`GetObject start`)
        out, err := s3Client.GetObject(ctx, &s3.GetObjectInput{
            Bucket:      bucket,
            Key:         key,
            // checksum always returned by default
            //ChecksumMode: types.ChecksumModeEnabled, // X-Amz-Checksum-Mode
        })
        panicIf(err, `failed s3Client.GetObject`)
        log.Println(`GetObject success`)
        log.Println(middleware.GetRawResponse(out.ResultMetadata).(*http.Response).Header)
    }
}
```

Object: Delete

`DELETE /<bucket>/<object>`

Deletes the requested object or empty directory. Deleting a non-empty directory causes an error. Deleting an object that does not exist returns no error. HTTP Response code of 204 is normal and indicates that the object was successfully deleted.

In case of a versioned bucket is possible to:

- Delete an object without specify any `versionId`: in this case a `DeleteMarker` is created
- Delete an object speicify a particular `versionId`

If the versioning, in a specific bucket, is in `Suspend State` then the `Delete` operation can remove the object only if his version ID is `null`. In any case it add a new `DeleteMarker` with `versionId` set to `null`.

To remove a specific `versionId`: `DELETE /<bucket>/<object>?versionId=<version_id>`

Object: Delete Multiple

`POST /<bucket>?delete`

The request body must contain a XML document in the format specified by the example, with a list of keys to delete (up to 1000).

Response

The response is a XML document, containing, for each key, either a `<Deleted>` element or an `<Error>` element, according to the result of the operation.

Example request body

```
<Delete>
  <Object>
    <Key>example1</Key>
  </Object>
  <Object>
    <Key>example2</Key>
  </Object>
</Delete>
```

Example response

```
<?xml version="1.0" encoding="UTF-8"?>
<DeleteResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Deleted>
    <Key>example1</Key>
  </Deleted>
  <Error>
    <Key>example2</Key>
    <Code>AccessDenied</Code>
    <Message>Access Denied</Message>
  </Error>
</DeleteResult>
```

Object Legal Hold: Retrieve

Gets an object's current Legal Hold status.

Parameter	Description
versionId	specify, if present, the version of the resource where to apply the lock

The body of the response is an XML, formatted as in the example, where the fiel`Status` could assume only two values: *ON*, *OFF*

Example request

```
GET /<object-key>?legal-hold&versionId=<version-id> HTTP/1.1
```

Example response

```
HTTP/1.1 200

<?xml version="1.0" encoding="UTF-8"?>
<LegalHold>
  <Status>ON</Status>
</LegalHold>
```

Object Legal Hold: Set

Set a Legal-Hold lock on an existent object in a specific bucket where the S3-Locking (se`CreateBucket`) is enabled.

Parameter	Description
VersionId	specify, if present, the version of the resource where to apply the lock

The body of the request is an XML, formatted as in the example, where the fiel`Status` could assume only two values: *ON, OFF*

Example request

```
PUT /{Key+}?legal-hold&VersionId=VersionId HTTP/1.1
Host: s3.lyve.seagate.com
x-amz-date: Thu, 15 Nov 2012 00:17:21 GMT
Content-MD5: ContentMD5

<?xml version="1.0" encoding="UTF-8"?>
<LegalHold xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Status>ON</Status>
</LegalHold>
```

Example response

```
HTTP/1.1 200
```

Object Metadata: Retrieve

`HEAD /<bucket>/<object>`

Allows retrieving metadata about a given object. It supports the same parameters and returns the same response as the "GET Object" call, but with an empty body. If th`x-rstor-replication-status: true` header is specified in the request, the API provides the replication status in the`X-Rstor-Replication-Status` header as a list of regions. This request support`x-amz-checksum-mode: ENABLED` to retrieve checksum stored if using custom checksum algorithm (`x-amz-checksum-algorithm` or `x-amz-checksum-*`).

Example Response

```
HTTP/1.1 200 OK
Accept-Ranges: bytes
Content-Length: 57301
Content-Type: text/plain
ETag: "18e20a76b174f184fad0febced4dbc51"
Last-Modified: Fri, 19 Jul 2019 17:14:35 GMT
X-Lyve-Replication-Status: DEN02,SJC03
Date: Mon, 22 Jul 2019 10:45:57 GMT
```

Object Retention: Retrieve

Retrieves an object's retention settings.

Parameter	Description
versionId	specify, if present, the version of the resource where to apply the lock

The body of the response is an XML, formatted as in the example, where the fiel`Mode` could assume only two values: *GOVERNANCE*, *COMPLIANCE*

Example request

```
GET /<object-key>?legal-hold&versionId=<version-id> HTTP/1.1
```

Example response

```
HTTP/1.1 200

<?xml version="1.0" encoding="UTF-8"?>
<Retention>
  <Mode>GOVERNANCE</Mode>
  <RetainUntilDate>2020-10-02T15:00:00.05Z</RetainUntilDate>
</Retention>
```

Object Retention: Set

Set a retention lock on an existent object in a specific bucket where the S3-Locking (see `CreateBucket`) is enabled.

Parameter	Description
VersionId	specify, if present, the version of the resource where to apply the lock

The body of the request is an XML, formatted as in the example, where the field `Mode` could assume only two values: *GOVERNANCE*, *COMPLIANCE*.

The `RetainUntilDate` field is the timestamp: should be set in the future (see RFC3339 as a format reference).

Example request

```
PUT /{Key+}?retention&VersionId=VersionId HTTP/1.1
Host: s3.lyve.seagate.com
x-amz-bypass-governance-retention: BypassGovernanceRetention
Content-MD5: ContentMD5

<?xml version="1.0" encoding="UTF-8"?>
<Retention xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
```

```
  <Mode>GOVERNANCE</Mode>
  <RetainUntilDate>2020-10-02T15:00:00.05Z</RetainUntilDate>
</Retention>
```

Example response

```
HTTP/1.1 200
```

Object RS Presign Link: Retrieve

Obtain a link to the object, pre-signed with your most recently used Access Key

Parameter	Description
ExpirationDate	Expiration date for the pre-signed URL. Cannot be more than 30 days (720h) in the future.

Example request

```
POST /bucket/path/to/object?rs-presign= HTTP/1.1
Host: [...]
Authorization: [...]
Content-Type: application/x-www-form-urlencoded
X-Amz-Content-Sha256: [...]
X-Amz-Date: [...]

ExpirationDate=2019-10-24T07%3A04%3A47.123Z
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/json
Date: [...]
Content-Length: [...]

{
  "link": "https://s3.example.com/bucket/...",
```

```
  "accessKeyId": "STX1JH..."
}
```

Object Tags: Retrieve

Return, if present, the tagging of an object.

Example request

```
GET /bucket/objectkey?tagging&versionId=objectVersion HTTP/1.1
Host: [...]
Authorization: [...]
X-Amz-Content-SHA256: [...]
X-Amz-Date: [...]
```

Example response

```
HTTP/1.1 200 OK
Content-Type: application/xml
Date: [...]
Content-Length: 184

<?xml version="1.0" encoding="UTF-8"?>
<Tagging xmlns="">
<TagSet>
<Tag>
  <Key>KEYTEXT</Key>
  <Value>VALUETEXT</Value>
</Tag>
<Tag>
  <Key>Type</Key>
  <Value>Script</Value>
</Tag>
</TagSet>
</Tagging>
```

Object Tags: Add or Replace

Add tagging to an object version. Overwrites tags if existing, to update tagging first get the old tagging, modify them locally and upload the new tagging.

Tagging are in the form of key-value pair: `Format=Image`

At most 10 tags per object version can be specified.

The length of a tag key can be up to to 128 unicode characters long. The length of a tag value can be up to to 256 unicode characters long.

Example request

```
PUT /bucket/objectkey?tagging&versionId=objectVersion HTTP/1.1
Host: [...]
Authorization: [...]
X-Amz-Content-SHA256: [...]
X-Amz-Date: [...]

<Tagging xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<TagSet>
<Tag>
  <Key>KEYTEXT</Key>
  <Value>VALUETEXT</Value>
</Tag>
<Tag>
  <Key>Type</Key>
  <Value>Script</Value>
</Tag>
</TagSet>
</Tagging>
```

Example response

```
HTTP/1.1 200 OK
x-amz-version-id: 01DVAXPTT7XFVAMQ7P248ZNWNQ
Date: Thu, 05 Dec 2019 13:00:58 GMT
Content-Length: 0
```

Object Tags: Delete

Delete, if present, the tagging of an object.

Example request

```
DELETE /bucket/objectkey?tagging&versionId=objectVersion HTTP/1.1
Host: [...]
Authorization: [...]
X-Amz-Content-SHA256: [...]
X-Amz-Date: [...]
Content-Length: 0
```

Example response

```
HTTP/1.1 204 No Content
x-amz-version-id: [...]
Date: [...]
```

Multipart Upload: Retrieve

`GET /<bucket>?uploads`

Lists the multipart uploads for a given bucket that have been initialized but not yet completed or aborted.

Example response

```
<ListMultipartUploadsResult xmlns="http://s3.amazonaws.com/doc/2006-03-01">
  <Bucket>examplebucket</Bucket>
  <Upload>
    <Key>testMultipartTest.txt</Key>
    <UploadId>sNyryKj9Jzd5T1gdFpotuiRRcAASIz4nBNa24NSOYgHZYdl5aN</UploadId>
    <Initiator>
      <ID>99woic3r@example.com</ID>
      <DisplayName>99woic3r@example.com</DisplayName>
    </Initiator>
    <Owner>
      <ID>100000000001</ID>
      <DisplayName>example</DisplayName>
    </Owner>
    <StorageClass>STANDARD</StorageClass>
    <Initiated>2019-04-29T13:35:48.420Z</Initiated>
  </Upload>
</ListMultipartUploadsResult>
```

Multipart Upload: Initiate

`POST /<bucket>/<object>?uploads`

Initiates a new multipart upload, returning a new upload ID.

Post parameters

No request parameters used

Header parameters

The following optional headers are supported:

Header	Description
x-amz-tagging	The tag-set to assign to the uploaded object, formatted as a query string (for example, "Key1=Value1")
x-amz-server-side-encryption	SSE-S3 Algorithm (supported: AES256)
x-amz-server-side-encryption-customer-algorithm	SSE-C algorithm (supported: AES256)
x-amz-server-side-encryption-customer-key	SSE-C base64-encoded encryption key
x-amz-server-side-encryption-customer-key-MD5	SSE-C base64-encoded encryption key hash (MD5)

Header	Description
x-amz-checksum-algorithm or x-amz-sdk-checksum-algorithm	Custom checksum algorithm for the uploaded part, valid values: `CRC32`, `CRC32C`, `SHA1`, `SHA256`)

Additionally the following metadata headers are supported, and will be stored together with the object once the upload is completed.

- Cache-Control
- Content-Encoding
- Content-Disposition
- Content-Language
- Content-Type
- Expires
- `x-amz-meta-` metadata headers (see PutObject)

Example response

```
<?xml version="1.0" encoding="UTF-8"?>
<InitiateMultipartUploadResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <Bucket>bucketA</Bucket>
  <Key>objectKey</Key>
  <UploadId>ifWDmLwLgAR8QDN/tg67DObvjR4=</UploadId>
</InitiateMultipartUploadResult>
```

Example: How to do upload multipart using AWS-SDK-Go-v2

```
// connection part see PutObject example above

out, err := s3Client.CreateMultipartUpload(ctx, &s3.CreateMultipartUploadInput{
    Bucket:            bucket,
    Key:               key,
    //ChecksumAlgorithm: types.ChecksumAlgorithmCRC32,
})
panicIf(err, `failed s3Client.CreateMultipartUpload`)
```

Multipart Upload: Complete

On each part, there's optional `ChecksumCRC32`, `ChecksumCRC32C`, `ChecksumSHA1`, `ChecksumSHA256` fields, for example: `<ChecksumCRC32>abcdef==</ChecksumCRC32>`, which is used to verify the integrity of the concatenated checksum of the uploaded parts, this should match with the custom checksum-algorithm (`x-amz-checksum-algorithm` or `x-amz-sdk-checksum-algorithm`) that started in Initiate Multipart Upload phase. PartNumber must be in sequential order or the server will return an error.

Example response

```
<?xml version="1.0" encoding="UTF-8"?>
<CompleteMultipartUploadResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
   <Location>http://s3.amazonaws.com/abc-bucket/gparted-live-0.33.0-1-amd64.iso</Location>
   <Bucket>abc-bucket</Bucket>
   <Key>gparted-live-0.33.0-1-amd64.iso</Key>
   <ETag>&amp;quot;7b2ca2b72d990a59a6ac992e5a7f910c-9&amp;quot;</ETag>
</CompleteMultipartUploadResult>
```

Example response (in case of error)

```
<Error>
  <Code>InternalError</Code>
  <Message>We encountered an internal error. Please try again.</Message>
</Error>
```

`POST /<bucket/<object>?uploadId=<uploadId>`

Completes a multipart upload, assembling the parts in ascending partNumber order.

> **i** You are only allowed to complete a multipart upload from the same datacenter where the multipart was inited.

The request body is an XML document listing the parts to assemble, each with the partNumber and ETag respectively in the "Upload Part" request and response.

> **i** This operation can potentially take several seconds. To prevent the request from timing out, the server sends whitespace during the response. As a consequence, the outcome of the operation can either be specified as HTTP header or in the returned xml document.

Response

The response is an XML document, which specifies the outcome of the upload, and either the reassembled

file key and ETag, or an error. Moreover, in case the bucket has one or more lifecycle Expiration rules that apply to newly created object, the corresponding `Expiration` is returned.

Example request

```
<CompleteMultipartUpload xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<Part>
   <ETag>"a2f4f30ec726440bcc748981e82f43fc"</ETag>
   <PartNumber>1</PartNumber>
</Part>
<Part>
   <ETag>"d12b6b5a7f8f850c39f32a96e36b5f69"</ETag>
   <PartNumber>2</PartNumber>
</Part>
<Part>
   <ETag>"e882613760157a24939f87a210043d47"</ETag>
   <PartNumber>3</PartNumber>
</Part>
</CompleteMultipartUpload>
```

Example response

```
<?xml version="1.0" encoding="UTF-8"?>
<CompleteMultipartUploadResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
   <Location>http://s3.amazonaws.com/abc-bucket/gparted-live-0.33.0-1-amd64.iso</Location>
   <Bucket>abc-bucket</Bucket>
   <Key>gparted-live-0.33.0-1-amd64.iso</Key>
   <ETag>"7b2ca2b72d990a59a6ac992e5a7f910c-9"</ETag>
</CompleteMultipartUploadResult>
```

Example response (in case of error)

```
<Error>
  <Code>InternalError</Code>
  <Message>We encountered an internal error. Please try again.</Message>
</Error>
```

Multipart Upload: Abort

`DELETE /<bucket>/<object>?uploadId=<uploadId>`

Removes a multipart upload and all the parts that have been uploaded up to that moment.

> **i** You are only allowed to abort a multipart upload from the same datacenter where the multipart was inited.

Response

Returns a status code reflecting the outcome of the operation.

Multipart Upload: List Parts

`GET /<bucket>/<object>?uploadID=<uploadId>`

Lists the uploaded parts for a given multipart upload, with their ETag, part number, upload id, size and last modified time. It does not support pagination-related parameters: `max-parts` and `NextPartNumberMarker`.

Example response

```
<ListPartsResult xmlns="http://s3.amazonaws.com/doc/2006-03-01">
  <Bucket>examplebucket</Bucket>
  <IsTruncated>false</IsTruncated>
  <Part>
    <ETag>"3e7bb4a9c59d736603a0b764f1696b7d"</ETag>
    <PartNumber>1</PartNumber>
    <LastModified>2019-04-29T13:35:48.425Z</LastModified>
    <Size>5</Size>
  </Part>
  <Part>
    <ETag>"3e7bb4a9c59d736603a0b764f1696b7d"</ETag>
    <PartNumber>2</PartNumber>
    <LastModified>2019-04-29T13:35:48.430Z</LastModified>
    <Size>5</Size>
  </Part>
  <Key>testMultipartTest.txt</Key>
  <UploadId>sNyryKj9Jzd5T1gdFpotuiRRcAASIz4nBNa24NSOYgHZYdl5aN</UploadId>
  <Initiator>
    <ID>99woic3r@example.com</ID>
    <DisplayName>99woic3r@example.com</DisplayName>
  </Initiator>
  <Owner>
    <ID>100000000001</ID>
    <DisplayName>example</DisplayName>
```

```
  </Owner>
  <StorageClass>STANDARD</StorageClass>
  <Initiated>2019-04-29T13:35:48.420Z</Initiated>
</ListPartsResult>
```

Multipart Upload: Upload Part

`PUT /<bucket>/<object>?partNumber=<partNumber>&uploadID=<uploadID>`

The request is analogous to "Put Object", but specifies two additional query parameters: `partNumber` and `uploadId`. The parameters `object` and `uploadID` must be equal respectively to the ones provided and returned in the "Initiate Multipart Upload" call. The partNumber will be used to reassemble the uploaded parts.

Header parameters

The UploadPart operation supports the following optional headers:

Header	Description
Content-Md5	The MD5 hash of the content. If present, it is checked as an additional guarantee that the object was correctly received
x-amz-copy-source	The object specified in the header is copied to the target `partNumber` for specified `uploadId` multipart
x-amz-server-side-encryption-customer-algorithm	SSE-C algorithm (supported: AES256)
x-amz-server-side-encryption-customer-key	SSE-C base64-encoded encryption key
x-amz-server-side-encryption-customer-key-MD5	SSE-C base64-encoded encryption key hash (MD5)

Header	Description
x-amz-copy-source-server-side-encryption-customer-algorithm	SSE-C algorithm of the source object (supported: AES256)
x-amz-copy-source-server-side-encryption-customer-key	SSE-C base64-encoded decryption key of the source object
x-amz-copy-source-server-side-encryption-customer-key-MD5	SSE-C base64-encoded decryption key hash (MD5) of the source object
x-amz-checksum-algorithm or x-amz-sdk-checksum-algorithm	Custom checksum algorithm for the uploaded part, valid values: `CRC32`, `CRC32C`, `SHA1`, `SHA256`)
x-amz-checksum	Optional base64-encoded checksum for custom checksum-algorithm
x-amz-trailer	Optional compatibility header, may contain value: `x-amz-checksum-crc32`, `x-amz-checksum-crc32c`, `x-amz-checksum-sha1`, or `x-amz-checksum-sha256`
x-amz-checksum-crc32	CRC32 checksum of the object (base64-encoded)
x-amz-checksum-crc32c	CRC32C checksum of the object (base64-encoded)
x-amz-checksum-sha1	SHA1 checksum of the object (base64-encoded)

Header	Description
x-amz-checksum-sha256	SHA256 checksum of the object (base64-encoded)

If checksum-algorithm supplied, it would behave the same as Put Object. Optionally if checksum calculation done on client side, `x-amz-checksum-crc32` or `x-amz-cheecksum-crc32c`, `x-amz-checksum-sha1` or `x-amz-checksum-sha256` can be sent, server will verify the checksum and return error if it does not match. `x-amz-checksum-*` not set, server will calculate the checksum that can be used optionally on Complete Multiple Upload.

Response

The response has an empty body, and specifies a status code (reflecting the outcome of the operation), and the ETag of the uploaded object as a header. The returned ETag must be noted, as it is required in the Complete Multipart Upload request.

Example response

```
HTTP/1.0 200 OK
date: Mon, 25 Feb 2019 12:10:46 GMT
etag: "a2f4f30ec726440bcc748981e82f43fc"
```

RS Speed Test (Download): Retrieve

Perform a download speed test. Backend sends random binary data for 10 seconds. Available only to resellers.

Example request

```
GET /speedtest/download HTTP/1.1
Host: [...]
Accept-Encoding: [...]
Authorization: [...]
User-Agent: [...]
X-Lyve-Speedtest: 1
```

Example response

```
HTTP/1.1 200 OK
Date: [...]
Content-Type: application/octet-stream

(random binary data)
```

RS Speed Test (Upload): Retrieve

Perform an upload speed test, probably through a pre-signed link. Backend reads request body for up to 10 seconds. Available only to resellers.

Example request

```
PUT /speedtest/upload HTTP/1.1
Host: [...]
Accept-Encoding: [...]
Authorization: [...]
Content-Length: [...]
Content-Type: application/octet-stream
User-Agent: [...]
X-Lyve-Speedteest: 1

(random binary data)
```

Example response

```
HTTP/1.1 204 No Content
Date: [...]
```

Header Parameters

Headers	Description

Headers	Description
x-amz-bucket-object-lock-enabled	If set to true create the bucket in locking mode and activate the versioning
x-amz-grant-read	
x-amz-grant-write	

Response

Returns 200 in case of success, 400 in case the bucket name is not valid, or 409 conflict error when you try to create a bucket which is already present.

POST s3v4 Uploads

The `POST` operation adds an object to a specified bucket using HTML forms`POST` is an alternate form of PUT that enables browser-based uploads as a way of putting objects in buckets. Parameters that are passed to `PUT` via HTTP Headers are instead passed as form fields to`POST` in the `multipart/form-data` encoded message body. You must have`WRITE` access on a bucket to add an object to it. Lyve Cloud never stores partial objects: if you receive a successful response, you can be confident the entire object was stored.

Lyve Cloud is a distributed system. If Lyve Cloud receives multiple write requests for the same object simultaneously, all but the last object written is overwritten.

To ensure that data is not corrupted traversing the network, use the`Content-MD5` form field. When you use this form field, Lyve Cloud checks the object against the provided MD5 value. If they do not match, Lyve Cloud returns an error. Additionally, you can calculate the MD5 value while posting an object to Lyve Cloud and compare the returned ETag to the calculated MD5 value. The ETag only reflects changes to the contents of an object, not its metadata.

Policy

The policy required for making authenticated requests using HTTP POST is a UTF-8 and base64-encoded document written in JavaScript Object Notation (JSON) that specifies conditions that the request must meet. Depending on how you design your policy document, you can control the access granularity per-upload, per-user, for all uploads, or according to other designs that meet your needs.

> **i** Note: although the policy document is optional, we highly recommend that you use one in order to control what is allowed in the request. If you make the bucket publicly writable, you have no control at all over which users can write to your bucket.

The POST policy always contains the `expiration` and `conditions` elements. The example policy uses two condition matching types (exact matching and starts-with matching). The following sections describe these elements.

The `expiration` element specifies the expiration date and time of the POST policy in ISO8601 GMT date format. For example, `2013-08-01T12:00:00.000Z` specifies that the POST policy is not valid after midnight GMT on August 1, 2013.

Condition matching

Following is a table that describes condition matching types that you can use to specify POST policy conditions (described in the next section). Although you must specify one condition for each form field that you specify in the form, you can create more complex matching criteria by specifying multiple conditions for a form field.

Condition Match Type	Description

Exact Matches	The form field value must match the value specified. This example indicates that the ACL must be set to public-read:
	`{"acl": "public-read" }`
	This example is an alternate way to indicate that the ACL must be set to public-read:
	`[ "eq", "$acl", "public-read" ]`

Starts With	The value must start with the specified value. This example indicates that the object key must start with user/user1: `["starts-with", "$key", "user/user1/"]`

Matching Content-Types in a Comma-Separated List	Content-Types values for a `starts-with` condition that include commas are interpreted as lists. Each value in the list must meet the condition for the whole condition to pass. For example, given the following condition:
	`["starts-with", "$Content-Type", "image/"]`
	The following value would pass the condition:
	`"image/jpg,image/png,image/gif"`
	The following value would not pass the condition:
	`["image/jpg,text/plain"]`
	Note: data elements other than `Content-Type` are treated as strings, regardless of the presence of commas.

Condition Match Type	Description
Matching Any Content	To configure the POST policy to allow any content within a form field, use starts-with with an empty value (""). This example allows any value for success_action_redirect: `["starts-with", "$success_action_redirect", ""]`
Specifying Ranges	For form fields that accept a range, separate the upper and lower limit with a comma. This example allows a file size from 1 to 10 MiB: `["content-length-range", 1048576, 10485760]`

Conditions

The `conditions` in a POST policy is an array of objects, each of which is used to validate the request. You can use these conditions to restrict what is allowed in the request. For example, the provided example policy conditions require the following:

- Request must specify the `mybucket` bucket name.
- Object key name must have the `user/paul` prefix.
- Object ACL must be set to public-read.

Each form field that you specify in a form (except `x-amz-signature`, `file`, `policy`, and field names that have an `x-ignore-` prefix) must appear in the list of conditions.

> **i** All variables within the form are expanded prior to validating the POST policy. Therefore, all condition matching should be against the expanded form fields. Suppose that you want to restrict your object key name to a specific prefix `user/user1`). In this case, you set the key form field to `user/user1/${filename}`. Your POST policy should be `[ "starts-with", "$key", "user/user1/" ]` (do not enter `[ "starts-with", "$key", "user/user1/${filename}" ]`).

Policy document conditions are described in the following table.

Element Name	Description
acl	Specifies the ACL value that must be used in the form submission. This condition supports exact matching and `starts-with` condition match type discussed in the preceding section.

Element Name	Description
bucket	Specifies the acceptable bucket name. This condition supports exact matching condition match type.
content-length-range	The minimum and maximum allowable size for the uploaded content. This condition supports `content-length-range` condition match type.
Cache-Control Content-Type Content-Disposition Content-Encoding Expires	REST-specific headers. For more information, see POST Object. This condition supports exact matching and `starts-with` condition match type.
key	The acceptable key name or a prefix of the uploaded object. This condition supports exact matching and `starts-with` condition match type.
success_action_redirect redirect	The URL to which the client is redirected upon successful upload. This condition supports exact matching and `starts-with` condition match type.
success_action_status	The status code returned to the client upon successful upload if `success_action_redirect` is not specified. This condition supports exact matching.
x-amz-algorithm	The signing algorithm that must be used during signature calculation. For AWS Signature Version 4, the value is `AWS4-HMAC-SHA256`. This condition supports exact matching.

Element Name	Description

x-amz-credential	The credentials that you used to calculate the signature. It provides access key ID and scope information identifying region and service for which the signature is valid. This should be the same scope you used in calculating the signing key for signature calculation.
	It is a string of the following form:
	
	`<your-access-key-id>/<date>/<optional-lyve-region>/s3/aws4_request`
	
	For example:
	
	`LyveACCESSKEY92724/20130728/us-east-1/s3/aws4_request`
	
	For Lyve Cloud, the service string is always s3. Regions are specified for compatibility reasons, although multi-geographical distribution is automatically enforced. This is required if a POST policy document is included with the request.
	This condition supports exact matching.

x-amz-date	The date value specified in the ISO8601 formatted string. For example, `20130728T000000Z`. The date must be same that you used in creating the signing key for signature calculation. This is required if a POST policy document is included with the request. This condition supports exact matching.
x-lyve-meta-*	User-specified metadata. This condition supports exact matching and starts-with condition match type.
x-lyve-*	Any other documented Lyve-specific meta This condition supports exact matching.

Character Escaping

Characters that must be escaped within a POST policy document are described in the following table.

Escape Sequence	Description
\	Backslash

Escape Sequence	Description
\$	Dollar symbol
\b	Backspace
\f	Form feed
\n	New line
\r	Carriage return
\t	Horizontal tab
\v	Vertical tab
\uxxxx	All Unicode characters

Example form

```
<form action="http://s3.lyve.seagate.com/bucketname"
target="s3target"
enctype="multipart/form-data"
method="POST">

<!-- Key is required (key name of uploaded object) -->
<input id="key"
type="hidden"
name="key"
value="test_browser_myfile.txt" />

<!-- Policy is required for authenticated requests -->
<input id="Policy"
type="hidden"
name="Policy"
value="[BASE64_ENCODED_SECURITY_POLICY]" />

<!-- X-Amz-Algorithm is required for authenticated requests -->
<input id="X-Amz-Algorithm"
type="hidden"
name="X-Amz-Algorithm"
value="AWS4-HMAC-SHA256" />

<!-- X-Amz-Credential is required for authenticated requests -->
<input id="X-Amz-Credential"
type="hidden"
name="X-Amz-Credential"
```

```
  value="AWS4XZUHH4DVBPMSZB7ODUFVVHGUXMXLY4VRJRJ4BFHJ7CGJACI4LPLA/20200819/eu-east-1/s3/aws4_request" />

  <!-- X-Amz-Date is required for authenticated requests -->
  <input id="X-Amz-Date"
  type="hidden"
  name="X-Amz-Date"
  value="20200819T142942Z" />

  <!-- X-Amz-Signature is required for authenticated requests -->
  <input id="X-Amz-Signature"
  type="hidden"
  name="X-Amz-Signature"
  value="[HMAC_SHA256_HASH]" />

  <!-- optional sample fields -->
  <input id="acl"
  type="hidden"
  name="acl"
  value="public-read" />

  <input id="success_action_redirect"
  type="hidden"
  name="success_action_redirect"
  value="http://somewhere.com/redir/redirect" />

  <input id="x-lyve-meta-something"
  type="hidden"
  name="x-lyve-meta-something"
  value="c25f87c5-2a43-4da6-901c-b3fac171e572" />

  <input id="Content-Type"
  type="hidden"
  name="Content-Type"
  value="text/plain" />

  <!-- the file content (mandatory) -->
  <input id="file"
  type="file"
  name="file"/>

  <button type="submit"
  class="btn btn-default">Upload</button>
  </form>
```

Example policy

```
  {
  "expiration": "2007-12-01T12:00:00.000Z",
  "conditions": [
```

```
  {
  "acl": "public-read"
  },
  {
  "bucket": "mybucket"
  },
  [
  "starts-with", "$key", "user/paul/"
  ]
]
}
```

Example request

```
POST / HTTP/1.1
Host: destinationBucket.s3.lyve.seagate.com
User-Agent: browser_dataAmazon
Accept: file_types
Accept-Language: Regions
Accept-Encoding: encoding
Accept-Charset: character_set
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=9431149156168
Content-Length: length

--9431149156168
Content-Disposition: form-data; name="key"

acl
--9431149156168
Content-Disposition: form-data; name="tagging"

<Tagging><TagSet><Tag><Key>Tag Name</Key><Value>Tag Value</Value></Tag></TagSet></Tagging>
--9431149156168
Content-Disposition: form-data; name="success_action_redirect"

success_redirect
--9431149156168
Content-Disposition: form-data; name="Content-Type"

content_type
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-uuid"

uuid
--9431149156168
Content-Disposition: form-data; name="x-amz-meta-tag"

metadata
```

```
--9431149156168
Content-Disposition: form-data; name="LyveAccessKeyId"

access-key-id
--9431149156168
Content-Disposition: form-data; name="Policy"

encoded_policy
--9431149156168
Content-Disposition: form-data; name="Signature"

signature=
--9431149156168
Content-Disposition: form-data; name="file"; filename="MyFilename.jpg"
Content-Type: image/jpeg

file_content
--9431149156168
Content-Disposition: form-data; name="submit"

Upload to Lyve Cloud
--9431149156168--
```