RSTOR

# RSTOR Space

## User Guide

December 2020

*Prepared for:* RSTOR Customers

*Prepared by:* RSTOR Support

# Abstract / Summary

This is a User Guide for both resellers and customers of RSTOR's Object Storage System, RSTOR Space.

# Table of Contents

# 1 Introduction

**RSTOR Space** is RSTOR's core product offering that is **S3 compatible** with public and hybrid infrastructures. RSTOR Space allows customers to safely store their data with multi-copy geographically distributed data protection, immutability options and eventually consistent replication. RSTOR Space is hyper-scalable to serve the needs of varying workload types.

As an S3 compatible storage system, RSTOR Space can easily be integrated with existing S3 compatible applications.

An **S3 bucket** is a public cloud storage resource which contains objects. The Simple Storage Service **(S3)** data model uses a flat structure, there is no hierarchy of folders and subfolders, all objects are stored inside the root of the bucket. It is possible to filter objects using prefixes and delimiters to a subset of the bucket. An object is uniquely identified by its bucket, its full name (also known as object ID) and optionally the associated metadata.

## 1.1 Clients and Network Connectivity

RSTOR Space exposes an S3 compatible interface over HTTPS. The interface can be used in two ways:

Interactively:

- RSTOR's Native Web GUI, best for simplified management access
- Third-Party S3 compatible GUI or CLI client

Programmatically:

- Via API calls with S3 compatible SDKs or libraries with Endpoint, Access Key and Secret Key credentials

## 1.2 IP Ranges

RSTOR publishes its storage service IP address ranges in JSON format. The network IP prefixes and ranges can be found through this link:

https://rstor.io/ip-range/ip-range-rstor.json

This displays all of the relevant IP addresses per region for each of our available data centers. This URL will be updated as more nodes and regions are added.


```
{
  "syncToken": "1572876220",
  "createDate": "2019-11-4-14-03-40",
```

```
  "prefixes": [
    {
      "ip_prefix": "209.163.124.136",
      "region": "dca02",
      "service": "RSTORAGE"
    },
    {
      "ip_prefix": "209.163.124.138",
      "region": "dca02",
      "service": "RSTORAGE"
    },
    {
      "ip_prefix": "209.163.127.135",
      "region": "den02",
      "service": "RSTORAGE"
    },
    {
      "ip_prefix": "209.163.127.136",
      "region": "den02",
      "service": "RSTORAGE"
    },
    {
      "ip_prefix": "209.163.122.138",
      "region": "sjc03",
      "service": "RSTORAGE"
    },
    {
      "ip_prefix": "209.163.122.145",
      "region": "sjc03",
      "service": "RSTORAGE"
    },
    {
      "ip_prefix": "216.180.121.136",
      "region": "toy01",
      "service": "RSTORAGE"
    },
    {
      "ip_prefix": "216.180.121.138",
      "region": "toy01",
      "service": "RSTORAGE"
    }
  ]
}
```


## 1.3 API Guide

The API Guide is accessible from the RSTOR Space Web GUI:

https://s3.<customername>.rstorcloud.io/apidoc/index.html

RSTOR Space Object Storage has been tested against all major SDKs for S3 clients (boto python library, aws sdk Go library, aws sdk JS library), S3 GUIs (CyberDuck, S3browser), and various S3 capable applications. Please check with RSTOR for the compatibility listing, if you do not see one applicable to your needs, let us know.

# 2 Getting Started

## 2.1 Roles and Types (Account Identities)

There are three distinct roles for those who access the RSTOR Space system:

| Role | Description |
|---|---|
| Admin | Can manage users and policies, with the exception of the Root account. A User identity will be assigned "Admin" rights. |
| Root | The initial and main identity of the customer account. Direct customer Root identities can also create access/secret key pairs for access. **The Root account cannot be deleted.* |
| User | Can perform operations on buckets according to the policies that are attached. It can change its own password, create access/secret key pairs for access. Reseller accounts do not allow creation or use of keys. |

The Root identity is assigned to the owner of the RSTOR Space account. The Root identity creates users and has the ability to assume the role of any underlying account.

In a Reseller account, the Root identity can also assume the role of accounts allocated below it either automatically or by manually allowing access. This is a feature to assist in debugging sub-account issues. This identity or account cannot be deleted.

## 2.2 Permissions Related to Roles and Identities

If you are configuring an application that needs to create buckets, there are two options for permissions:

1. Provide Root credentials so that application that will allow bucket creation based upon that application need. For example, backup applications will want to create a series of directories to manage the data structures that it writes to S3 storage.

2. Create a User that has "Admin" privileges and allow that user to perform *Read*, *Write*, *List* and *Delete* operations on "All Buckets" through a specified policy.

## 2.3 Login Attempt Rate Limits

The RSTOR Space service can be logged into from the Web GUI for any user with credentials to do so. As such, from a security perspective, rate limits have been implemented on the number of failed login attempts.

After 4 failed login attempts, the rate limit locks an account. A failed login attempt happens when any information provided during the login was incorrect. A locked account will not be allowed to login for 5 minutes from the last failed login attempt, where the rate limit occurred.

# 3 Reseller Management

## 3.1 Sign In As Reseller

Reseller accounts are provided to RSTOR customers. The reseller admin account needs to be whitelisted before access becomes available. The reseller account allocates customer accounts (known as subtenant accounts).

To access the Reseller interface, use the following URL: https://{resellername}.rstorcloud.io/admin/signin.

## 3.2 Set Up Customer Accounts

A Reseller sub-tenant (Customer Account) is created by using the add button under the customer menu in the upper right corner of the web GUI (or through the RSTOR API). This is used by the Root account to create the sub accounts. The example below is from a Reseller account.

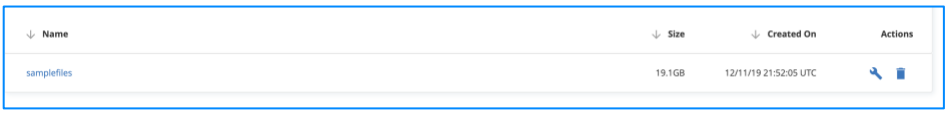
**Name:** Name of customer account

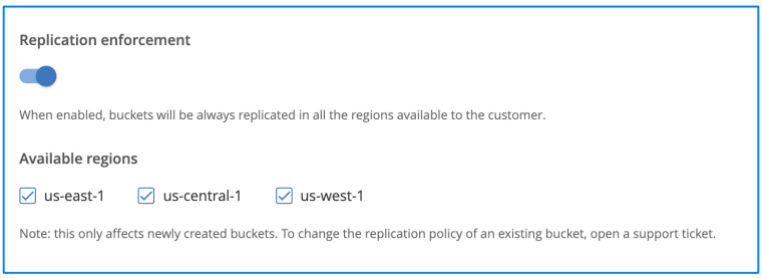
**Assume Role:** Grants permission for the reseller to assume the identity of the customer to provide support to the interface when required. The assume role feature has three options: *Enable*, *Disable*, and *Managed by Customer*. See Assume Role.

**Replication Enforcement:** An Administrator chooses which data centers to replicate data. If the setting is off, the UI will automatically select all 3 nodes as valid replication points. This feature prevents sub-accounts from adjusting the replication parameter. Replication Enforcement also affects the replication dots; if enabled, the dots will not be displayed.

**Disabled Replication Enforcement**


**Enabled Replication Enforcement**






**Customer Status:** disable the account without deleting contents

**Email:** address for the sub account administrator or owner/user

**Password:** initially set, but can also use "Reset Password" URI for end administrator/user

> **User must wait 1 minute before repeating a password reset request. No error is returned if the user does not respect the minimum interval between requests.*

**Regions:** may be locked to 3-region replication

## 3.3 Assume Role

Reseller accounts provide the ability to assume the role of a Customer Account in order to assist in debugging. When creating a Subtenant account, a reseller may set the Assume Role option to one of the three following options:

1. **Enabled:** Not managed by Subtenant; reseller will be able to assume the identity of the customer.
2. **Disabled:** Reseller will not be able to assume to identity of the customer.
3. **Managed by Customer:** Customer controls when this feature is enabled; reseller has to be authorized by the subtenant account before using the Assume Role feature. Once this feature is set during the account creation by the reseller, it cannot be modified.

To disable the feature, the subtenant root user must login and disable the feature in the "My Account" section. The subtenant account may choose to only allow Assume Role on a chosen basis, or not at all.

**This cannot be done via the "Assume Role" functionality provided to the Reseller. Once this feature is set during account creation by the reseller, it cannot be modified.*

When the Assume Role actions are completed, the Reseller account may return to the parent session by clicking on "Return to original session".

## 3.4 RProtect: Reseller

RProtect is a feature that enables more-granular whitelisting to resources for both Reseller and Customer personalities.

RProtect can explicitly ensure that no one, except a select IP address or a range of IP addresses have access to a specific resource. *It can also be used to remove all restrictions by using 0.0.0.0/0 as the whitelisted IP address.* RProtect currently works with IPv4 and is not designed for IPv6.

The RProtect feature may be enabled for a Reseller or fully disabled, meaning that anyone can access the buckets within the accounts. This feature is listed in the main navigation bar as well as being available via the RSTOR Space API.

1. Select RProtect in the menu bar and you will be presented with a subnet whitelisting page.

   If there are customers already created in the Reseller account, you will be presented with a list.

2. Select a customer from the drop-down list box. To add a new rule, select " + REQUEST RULE ".

3. Enter the subnet you want to create a whitelist rule for. The "Subnet" option may exist of a single IP address or a range of IP addresses. While there are no limitations on the size of the range, we have found the best practice is to minimize the range as small as possible.

   Confining the address range, a host address, or a /29 address range is the best scenario (8 addresses, 5 usable), where there might be a /22 (1024 IP addresses). In general, anything bigger than a /22 (/21 on down) may not be secure enough to meet security policy. For Single IPv4 address, use the format of 192.168.1.17/32 (with /32 on the end).

4. If desired, choose the RSTOR Space regions in which you want this whitelist to apply.

5. If desired, set a "Time to live" for the new IP address or range of addresses. Enter *0* or leave the time to live blank to make the enforceable period indefinite.

   **Zero (0) days is the default with no time limitations.*

6. There is also an option to add a 144-character note for future references; perhaps the reason, customer, support request number, or location of the whitelist, as an example.

7. Once finished, select "SAVE" and the IP address(es) is/are now in a pending state post submission of request.

There are three options next to the “Pending” status: the checkmark, “i”, and trash bin.

1. **Checkmark Icon**: “APPROVE” or “REJECT” the request

   Selecting “APPROVE” provides an option to add text to the notes field, if applicable. Approved requests are added to the list, with most recent to the top of the list, and will display “ACTIVE” next to it.

   Selecting “REJECT” will remain in the list of requests with “REJECTED” next to it.

2. **Trash Bin Icon**: Deletes all requests, whether “APPROVED” or “REJECTED”

3. **“i” or Information Icon**: Views the notes associated with the requests

As with the Reseller account, the Customer account can also request a range of addresses or a single IP address to be whitelisted.

**Requests from Customer or sub-tenant accounts will require approval of the Reseller administration.*

## 3.5 Statistics of Selected Customer

When you are signed into the Reseller portal as the Root identity, overall account statistics are available. The stats shown are total transfer rates and the amount of storage used. It displays details by the day in which there are downloads and uploads occurring, along with a traffic overview. These statistics are also available for download using the down arrow on the righthand side of the screen.

1. To view the statistics, click on “Stats” in the top navigation bar.

## 3.6 Statistics of Transfer

In the RSTOR Space Reseller portal, you can display statistics of the buckets under your reseller sub-user (Customer) account along with the storage and transfer rates. It will also display a graph with the total used space (GB), number of objects stored, and egress and ingress traffic, all aggregated by day. This can give insight into when your users seem to be using the most bandwidth.

The admin can choose the date interval to display and can export the data to csv for easier processing using third-party programs.

1. To view the statistics, click on “Stats” in the top navigation bar.

Information on how to obtain the same data in a programmatic way is available in the API documentation.

# 4 Customer Management

S3 API has emerged as the dominant API for storing large amounts of unstructured data and RSTOR Space Object Storage supports the standard S3 RESTful API. The full API guide is available at:

https://<resellername>.rstorcloud.io/apidoc/index.html#introduction

## 4.1 Sign In As Customer

If you are a user, sign in as a customer. If you are an admin, sign in as a reseller. Customers may access their buckets through a DNS-style or Virtual-Host convention:

- DNS style uses https://$bucketname.$endpoint/$path/
- Virtual-Host style uses https://$endpoint/$bucketname/$path

Where $endpoint is:

- s3.rstorcloud.io for direct customers
- s3.resellername.rstorcloud.io/admin for reseller customers

To access a Customer Account, use the following URL: https://{resellername}.rstorcloud.io/signin.

## 4.2 Access and Configuration

To access the subaccount of a Reseller, use the following URL: https://<resellername> rstorcloud.io/signin.

Once logged in, the account page gives access to options such as configuring MFA, changing the account password, and generating keys.

## 4.3 Create a Bucket

To create a bucket, complete the following steps.

1. Go to “Buckets” using the top navigation bar.
2. Click on the “Add New Bucket” icon in the upper right corner.

3. A popup will appear where you can fill the necessary information to make a bucket. You can select which sites you would like your bucket to be replicated to.

   **If this option is not available, please contact your overarching admin as they may have disabled this setting.*

4. Through Access Mode, buckets can be set as “Private”, “Public”, or “Custom”. Check the boxes for the permissions that you want to provide. All new buckets are private by default. To grant access to your bucket to the general public (everyone in the world), select “Public” under Access Mode. Granting public access permissions means that anyone can access files in the bucket.

   The format for creating a public bucket is: https://bucketname.endpoint.path where there is case sensitivity in the letters.

   An example of using JSON to set a bucket to public is below, via web browser:

   https://s3.demo.rstorcloud.io/samplefiles/createrandomfiles.sh*

   **This file will be accessible to anyone who can access RSTOR Space.*

   RSTOR Space leverages the concept of buckets: each bucket is a container for objects. All new buckets are private by default. A bucket can be made public by applying the following policy to it:

```
{
"Version": "2019-10-17",
    "Statement": [
    {
        "Action": ["s3:GetBucketLocation"],
        "Effect": "Allow",
        "Principal": { "AWS": ["*"] },
        "Resource": ["arn:aws:s3:::BUCKETNAME"],
        "Sid": ""
    },
    {
        "Action": ["s3:GetObject"],
        "Effect": "Allow",
        "Principal": { "AWS": ["*"] },
        "Resource": ["arn:aws:s3:::BUCKETNAME/*"],
        "Sid": ""
    }
    ]
}
```

   The policy can be deleted any time with a `DELETE /BUCKETNAME/?policy` request to make the bucket private again.

Bucket policy can also be managed from the Web GUI using a simple point-and-click web interface.

1. Go to “Buckets” using the top navigation bar.
2. Select the wrench icon under the “Actions” column for that bucket to change the Access Mode (Private or Public) or change versioning.

## 4.4 Delete a Bucket

To delete a bucket, typically you have to empty all of its objects, folders, etc., and delete all of its associated policies. However, RSTOR offers a SuperDelete feature that deletes the bucket and its entire contents for you at once. To use either of these, do the following:

1. Navigate to the “Buckets” home screen.
2. On the far right, beneath the “Actions” column, click on the “Trash” icon.
3. Here you will have the option to delete your bucket using one of the two methods talked about above. Use the dropdown menu to choose your deletion option. Remember that if you decide to proceed with SuperDelete, you should *not* close or refresh your browser and the command may need to be run several times.

## 4.5 Uploads

The Web GUI has a feature to upload files into RSTOR Space. There are two methods to upload files: either (1) drag and drop to the screen, or (2) from within a bucket select the “+” from the upper right corner to expand options.

**Using the drag and drop method:**

1. Go to “Buckets” using the top navigation bar and select your chosen bucket.
2. Once the files have been dragged into the screen, select “START UPLOAD” to begin uploading the folders and files to the bucket.

**Using the upload method:**

1. Go to “Buckets” using the top navigation bar and select your chosen bucket.
2. Select the “+” from the upper right corner to expand options.
3. Select “Upload files”.
4. Select “UPLOAD FROM COMPUTER” (you still have the option to drag and drop here too). Here you have the option to enable server-side encryption under the “Advanced Options” drop down.
5. Then, select “START UPLOAD” to begin uploading the folders and files to the bucket.

## 4.6 Pre-Signed URL

Objects can be shared with a pre-signed URL. To share an object from the Web GUI, select a folder and object you would like to share to anyone.

**Folders in object storage are objects.*

1. Select the round “i” information icon next to the trash can icon on the far right of your object.
2. Click on the “Generate pre-signed link” hyperlink right beneath the Actions section.
3. Select the length of time for the expiration period and click “GENERATE”.
4. Select the “clipboard” icon next to the expiration time or perform an operating system interface select, then copy.

## 4.7 Create Users

As a sub account under a Reseller, you can grant users access to your buckets. They have the following permissions: policies, buckets, and adding more users. To create a new user, follow the next steps.

1. Under “Users” in the menu bar, click the round icon in the top right corner to add a user.
2. Once in this menu, provide the email you would like to give access to.
3. After that has been provided, press “Create User”.
4. Once created, you can edit what permissions the new sub-user has by clicking on their email in the Users page.

## 4.8 Create Policies

To create policies in your RSTOR Space bucket, begin by accessing the Policies menu from the main bar.

1. Press the create policies button in the top right corner as shown above. This will open up a menu that will allow you to create a name and description for your new policy.
2. The option to select a bucket is located underneath the naming schema. Once a bucket is selected you will have the option to control what permissions a given bucket has access to.
3. Once the options are selected, press save, and those permissions will now apply to the bucket. If you need to change given permissions, permission settings can be found in this menu as well.

## 4.9 Generate Access Keys

It is important to note the following about access keys.

- A ROOT user without an {AccessKeyID, SecretAccessKey} pair cannot generate a pre-signed link. The request will not send a valid credentials object and will fail.
- The current user cannot generate a pre-signed link without an {AccessKeyID, SecretAccessKey} pair.
- The ROOT user of an account does not have an {AccessKeyID, SecretAccessKey} pair generated for them by default.

1. To create application credentials (access/secret key pairs) for the account, go to the RSTOR Portal and click on your name in the upper right corner. From there choose “MY ACCOUNT”.
2. Select “+ GENERATE KEY” in the bottom right corner.
3. Download the CSV key and manage as you normally would, such as with a password manager.
4. Once downloaded, use your favorite .csv compatible application (Excel, Google Sheets, Numbers) to view the contents. For a onetime view, select the “eye” icon for the secret key.

**Note that multiple key pairs may be created for an account.*

### 4.9.1 Generate Access Keys Using CLI

To install awscli refer to Amazon's guide: https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-install.html. Run "aws configure" and set up the AWS command line. To begin initial installation, both access keys and secret keys are needed.


Once configured, generate keys on command using the following command,

```
aws iam create-access-key --endpoint-url
```

https://iam.demo.rstorcloud.io

This will generate a new key pair.

## 4.10 Set Up Multifactor Authentication

1. To begin set up for multifactor authentication, access your accounts page by clicking on the top right corner of the UI where your name is listed and choosing "My Account."
2. Next, press the "ENABLE 2FA".
3. This will display a QR code along with a secret key in a popup menu. Most MFA apps will ask to scan this QR code. Your MFA app will then provide a token that needs to be entered into the available space below the QR code to confirm the setup.
4. Once setup is completed, a handful of recovery keys will be given in case you lose access to your MFA device. Please make sure to copy these down. This will be the only way to recover your locked account. Popular MFA apps are Authy and Google Authenticator.

## 4.11 Single Site Validation Testing

Create any hash for a given file. For this example, we created a Python script for a SHA256 hash to validate the file.

```
import hashlib
import os
import sys
```

```
with open(sys.argv[1], 'rb') as f1:
  with open(sys.argv[2], 'rb') as f2:
          file_1 = f1.read()
          file_2 = f2.read()
          sha_1 = hashlib.sha256(file_1).hexdigest()
          sha_2 = hashlib.sha256(file_2).hexdigest()
          print("SHA256 of file 1: {0}".format(sha_1))
          print("SHA256 of file 2: {0}".format(sha_2))
          print("SHA256 of file 1 == SHA256 of file 2 returns
{0}".format(sha_1==sha_2))
exit()
```

**Sample output for script:**

```
(base) Christophers-MacBook-Air:Documents christophermetz$ python3 Compare_Files.py king_image.jpg king_image_copy.jpg
SHA256 of file 1: 229552ad428a61f4b924504e628f95954a8c55e78e5e9d529e34f38ecc566129
SHA256 of file 2: 229552ad428a61f4b924504e628f95954a8c55e78e5e9d529e34f38ecc566129
SHA256 of file 1 == SHA256 of file 2 returns True
(base) Christophers-MacBook-Air:Documents christophermetz$
```

1. Upload the file to RSTOR Space.
2. Download the file from RSTOR Space to your machine.
3. Re-hash the file using the same hash and compare hashes.

## 4.12 Multi-Site Validation Testing

RSTOR Space offers multi-site replication to add redundancy to your dataset locality.

For RSTOR Space multi-site redundancy, follow these steps:

1. Create a hash for any given file.
    a. For example, use the SHA256 hash written in Python under "Single Site Validation Testing."
2. Upload the file to an RSTOR Space location.
3. At each of the three sites, spin up a compute instance and run the hashing script.
4. Use $REGIONNAME.<customername>.rstorcloud.io instead of s3.<customername>.rstorcloud.io as the endpoint.

a. REGIONNAME can be: sjc03, dca02, den02

5. Compare the hashes of the different locations against the original hash.

## 4.13 Migration

Transporter is a feature that customers can use to either migrate or copy data in buckets from one of the Cloud Service Providers (AWS, GCP, Azure) or Cloud Storage Providers (Wasabi) to RSTOR Space. This feature allows the migration to occur from a Cloud:Cloud perspective without first requiring a local copy of the objects. This is an S3 compatible feature that is available via the RSTOR Space Web GUI interface or with S3 compatible tools such as AWS CLI or S3Curl.

Transporter is a simple API which takes in the inputs: source location, source credentials, destination and credentials at destination, then does the heavy lifting of object copying.

Transporter is visible in the Bucket view of the RSTOR Space web interface.

1. Log in to the RSTOR Portal and navigate to the buckets screen.
2. On the top right, click on the round network icon, labeled RSTOR Transporter, to the left of the “bucket” icon.
3. To create a new migration, click on the “plus” icon in the upper right corner.
4. Select the migration source type.
5. Enter the migration source information. See the Transporter User Guide for how to identify the credentials.
6. Click “Confirm” and you will be brought back to the Transporter configuration screen.
7. Repeat the steps 4 and 5 for the migration destination.
8. Once all the necessary information has been entered, click on “START MIGRATION” icon on the right.
9. Here, you will have the option to track the migration with the email your account is under. You may also enable incremental sync which migrates only the objects that are not present in your migration destination. You will also have the option to store an additional copy in RSTOR Space and/or delete the source after the transfer has been completed.

   To enable any of these, check the box next to it. If the words and box are greyed out, that means that specific feature is unavailable for the current migration.
10. Review your chosen migration then click “CONFIRM” in the popup.

11. As the migration completes, you will be able to click on the Status message of Completed to view the logs, estimated time till completion, and if needed, to abort or resume migration.

Migration with command line – if you want to leverage the S3 API and a command line tool such as the open source S3curl, here is an example which leverages the customer name of **"demo"**.

```
awscurl --service rramp
https://rramp.demo.rstorcloud.io/migrations/RANDOMSTRINGHERE \
 -X POST \
 --data
'{"source":{"type":"s3v4","accessKey":"SOURCEACCESSKEY","secretKey":"S
OURCESECRETKEY","bucket":"SOURCEBUCKETNAME","endpoint":"https://SOURCE
ENDPOINT","region":"us-west-
1"},"destination":{"type":"s3v4","accessKey":"DESTACCESSKEY","secretKe
y":"DESTSECRETKEY","bucket":"DESTBUCKET","endpoint":"https://DESTENDPO
INT","region":"us-east-1"}}' \
 --access_key RSTORADMINACCESSKEY --secret_key RSTORADMINSECRETKEY
```

## 4.14 Encryption

RSTOR enforces encryption of data in flight, using TLS.

RSTOR supports Server-Side Encryption with Customer-Provided Keys: the user manages the encryption keys and RSTOR manages the encryption, as it writes to disks, and decryption, when accessing objects.

Caveats:

- The ETag in the response is the MD5 of the encrypted data.
- The user manages the mapping of which encryption key was used to encrypt the object. The user is responsible for tracking which encryption key was used for which object.
- If the bucket has versioning enabled, each object version can have its own encryption key. The user is responsible for tracking which encryption key was used for which object version.
- User is responsible for ensuring the security of encryption keys and for managing all the required safeguards for encryption keys, such as key rotation, on the client side.
- If the encryption key is lost, any GET request for an object without its encryption key will fail, and effectively data is lost as well. There is no recovery mechanism.

### 4.14.1 Configure Data At Rest Encryption

When data from multiple places collects in one bucket, it is called data at rest. RSTOR Space currently supports Data At Rest Encryption, or DARE.

1. To begin, a user needs to have a bucket of data that they would like to encrypt. To create a bucket, please refer here.

2. To begin to set up this encryption, a key is needed from a 3rd party application. RSTOR Space will then use the key pair that was generated to encrypt your data server side. When making the following API calls, a key can be provided to allow server-side encryption of the information.

   - PUT/POST Object
   - PUT Object Copy
   - Initiate Multipart Upload
   - Upload Part
   - Upload Part - Copy
   - Complete Multipart Upload
   - Get Object
   - Head Object

3. When making this request, you must include the key in the headers of the request. The following headers must be provided.

```
x-amz-server-side -encryption -customer-algorithm: AFDLKSJFJK
x-amz-server-side -encryption -customer-key:  AKFLJDKLF
x-amz-server-side -encryption -customer-key-MD5: XAKFHLD
```

An example request for putting a file into RSTOR Space would look something like this:

```
aws s3api put-object --bucket demo-sse
--key "keyname" \
--body example.txt   \
--sse-customer-algorithm "AES256" \
 --sse-customer-key <key>  \
--sse-customer-key-md5 <keyMD5>  \
--profile <profileName> \
 --endpoint-url "https://s3.<customername>.rstorcloud.io)“
```

### 4.14.2 Configure Server-Side Encryption

A policy can be used to enforce all objects that are uploaded to a bucket, “mybucketname”, to be server-side encrypted with AES256. For creating a policy, see this section.

```
$ awscurl --access-key XXX --secret-key YYY --service s3
https://s3.CUSTOMER.rstorcloud.io/mybucketname/?policy -X PUT -H "Content-
Type: application/json" --data "
{
    "Version": "2012-10-17",
    "Id": "PutObjPolicy",
    "Statement": [
        {
            "Sid": "DenyIncorrectEncryptionHeader",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::mybucketname/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-server-side-encryption": "AES256"
                }
            }
        },
        {
            "Sid": "DenyUnEncryptedObjectUploads",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::mybucketname/*",
            "Condition": {
                "Null": {
                    "s3:x-amz-server-side-encryption": true
                }
            }
        }
    ]
}"
```

To remove the aforementioned policy:

```
awscurl --access-key XXX --secret-key YYY --service s3
https://s3.CUSTOMER.rstorcloud.io/mybucketname/?policy -X DELETE
```

## 4.15 Object Locking

RSTOR Space supports Write Once Read Many (WORM) objects. A user can use this feature to prevent an object from getting overwritten for a period of time or indefinitely. Object locking can also be used for a legal hold. A legal hold is the same as a retention period, but it has no expiration date. Instead, a legal hold remains in place until you explicitly remove it. To use this feature, you must create a bucket with object lock configuration using the awscli. Enabling object locking on an existing bucket is possible; it also enables Versioning.

```
aws s3api create-bucket --bucket demo-lock --object-lock-enabled-for-
bucket --endpoint-url https://s3.demo.rstorcloud.io --profile local
```

1. Create a file in this bucket.
   ```
   aws s3api put-object --bucket demo-lock --key "divina" --body
   ~/divina_commedia.txt --endpoint-url  https://s3.demo.rstorcloud.io
   --profile local

   {
   "ETag": "\"d43dc972416413a6114fbe4321ee1979\"",
   "VersionId": "01DXHDYQ6Q7J0VGXPETMQY8W0Y-v"
   }
   ```

2. Create a legal-hold constraint for that file.
   ```
   aws s3api put-object-legal-hold --bucket demo-lock --key divina --
   legal-hold "Status=ON" --endpoint-url https://s3.demo.rstorcloud.io
   --profile local
   ```

3. Now remove the legal hold.
   ```
   aws s3api put-object-legal-hold --bucket demo-lock --key divina --
   version-id "01DXHDYQ6Q7J0VGXPETMQY8W0Y-v" --legal-hold "Status=OFF"
   --endpoint-url https://s3.demo.rstorcloud.io --profile local
   ```

4. Delete object lock.
   ```
   aws s3api delete-object --bucket demo-lock --key divina --version-id
   "01DXHDYQ6Q7J0VGXPETMQY8W0Y-v" --endpoint-url
   https://s3.demo.rstorcloud.io --profile local

   {
   "VersionId": "01DXHDYQ6Q7J0VGXPETMQY8W0Y-v"
   }
   ```

5. List the bucket.
   ```
   aws s3api list-object-versions --bucket demo-lock --prefix divina --
   endpoint-url https://s3.demo.rstorcloud.io --profile local
   ```

```
{
"DeleteMarkers": [
{
"Owner": {
"ID": "100000000001"
},
"Key": "divina",
"VersionId": "01DXHE1AW5HYQQ3DH24Q13Z1TM-v",
"IsLatest": true,
"LastModified": "2020-01-01T20:51:28.7730474Z"
}
]
}
```

6. Retain Lock commands. Create a file (called divina-sec).

```
aws s3api put-object --bucket demo-lock --key divina-sec --body
~/divina_commedia.txt --endpoint-url https://s3.demo.rstorcloud.io -
-profile local
```

```
{
"ETag": "\"d43dc972416413a6114fbe4321ee1979\"",
"VersionId": "01DXHEMXSDT3GB8Y9Y6DENXX0Q-v"
}
```

7. Create a retention lock.

```
aws s3api put-object-retention --bucket demo-lock --key divina-sec -
-retention="Mode=GOVERNANCE,RetainUntilDate=2020-01-
15T00:00:00.000Z" --endpoint-url https://s3.demo.rstorcloud.io --
profile local
```

8. Check the status.

```
aws s3api get-object-retention --bucket demo-lock --key divina-sec -
-endpoint-url https://s3.demo.rstorcloud.io --profile local
```

```
{
"Retention": {
"Mode": "GOVERNANCE",
"RetainUntilDate": "2020-01-15T00:00:00Z"
}
}
```

## 4.16 Bucket Logging

Bucket logging is a feature that provides the ability log access of a bucket for statistical or audit purposes. It also provides a mechanism to optionally store logs in a bucket other than the one being monitored.

There are some particulars for bucket logging that should be noted:

- Log records are periodically collected and consolidated into the bucket that was enabled for logging
- Bucket logging requires credentials
- If there is not activity in the bucket where logging is enabled, log files will not be created
- Logging is not meant to be a complete accounting of all requests

There are two methods to establish this feature:

1. Through the command line with tools like AWSCLI.
2. Through the RSTOR Space Web GUI.

### 4.16.1 Bucket Logging: CLI

Bucket activities will be uploaded every 15 minutes in the destination bucket in a file formatted like this one:

`bucket_logging_example_testlog_2020_02_12_20:28:08_819361559`

Example: `bucket_logging.json`

```
{
  "LoggingEnabled": {
    "TargetBucket": "bucketlogs",
    "TargetPrefix": "logs/",
    "TargetGrants": [
        {"Grantee": { "ID": "RSTOR_KEY_123", "Type":
"AccessKey"}}
    ]
  }
}
```

In order to activate the logging on a given bucket with success, we should provide:

- **TargetBucket**
- **TargetGrants** with at least one *Grantee* element with:
  - **ID:** a valid **Access Key**. You might already have configured in ~/.aws/credentials
  - **Type:** "AccessKey"
- **TargetPrefix** is optional and it's use only to

Example command:

```
aws s3api put-bucket-logging --bucket testbucket --bucket-logging-
status file:///user/test/bucket_logging.json --endpoint-url
"https://s3.demo.rstorcloud.io"
```

**Remove a bucket logging configuration:**

In order to remove a Bucket Logging configuration for a specific bucket is enough to perform a PUT giving an empty `BucketLoggingStatus` request, for example you can create a `disable.json`, so formed:

```
{
}
```

```
aws s3api put-bucket-logging --bucket testbucket --bucket-logging-
status file:///user/test/ disable.json --endpoint-url
"https://s3.demo.rstorcloud.io"
```

**Retrieve bucket logging configuration:**

To retrieve a bucket logging configuration for a bucket:

```
aws s3api get-bucket-logging --bucket testlog --endpoint-url
https://s3.demo.rstorcloud.io
```

This will look similar to:

```
<?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>RSTOR_KEY_123</ID>
            </Grantee>
            </Grant>
```

```
        </TargetGrants>
    </LoggingEnabled>
</BucketLoggingStatus>
```

**LogFormat**

**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.PutObject, 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

**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

**Log Examples**

```
100000000001 testlog [12/Feb/2020:20:28:02 +0100] "[::1]:34178"
"RSTOR_KEY_123" "qfqun6v6dido" s3:PutObject "testkey" "/testlog/testkey"
"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"
"RSTOR_KEY_123" "50gzg1qj4ftt" s3:HeadObject "testkey"
"/testlog/testkey" "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"
"RSTOR_KEY_123" "eybjzx8hvf5s" s3:DeleteObject "testkey"
"/testlog/testkey" "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"
"RSTOR_KEY_123" "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 - "-"
100000000001 testlog [12/Feb/2020:20:28:02 +0100] "[::1]:34178"
"RSTOR_KEY_123" "misyyofy69qw" s3:HeadObject "foo" "/testlog/foo" "200"
"-" 20 20 1 0 "" "aws-sdk-go/1.28.13 (go1.13.7; linux; amd64)"
01E0XDYR0TABEJXCHQ300Z8BHE - SigV4 SSL AuthHeader - "-"
100000000001 testlog [12/Feb/2020:20:28:02 +0100] "[::1]:34178"
"RSTOR_KEY_123" "7lkfytqs6d7x" s3:DeleteObject "foo" "/testlog/foo"
"200" "-" "0" "0" 2 0 "" "aws-sdk-go/1.28.13 (go1.13.7; linux; amd64)"
"-" - SigV4 SSL AuthHeader - "-"
```

## 4.16.2 Bucket Logging: RSTOR Space Web GUI

1. In the Web GUI, select the following icon on the right side of the bucket, in the “Actions” column.

2. By default, bucket logging is “Disabled”. By changing it to “Enable”, the Web User Access Key will display by default. This field will accept any RSTOR Space Access Key.

3. Select the bucket for logging storage.

4. Create an optional folder name in the bucket.

**Status**
When logging status is “Disabled”, all other form fields are disabled. When changing the status from “Enabled” to “Disabled”, current values are still saved in the form until submission.

Submitting a form with the status “Disabled” will delete the bucket logging configuration and all data in the form will be lost.

**Grantee**
The form has a dropdown select of AccessKeyIDs associated with the current user and will autocomplete using the prefix substring currently in the text input.

If the desired AccessKeyID is not in the pre-fetched list, any arbitrary string (i.e. AccessKeyIDs managed by other users) is a valid input. AccessKeyIDs must be alphanumeric and contain precisely 26 characters.

**Saving Logs**
You must select a bucket from the dropdown list. It will prefetch with all buckets the current user has access to.

**Log Prefix**
Log prefixes are optional. By default, it will save log files in the bucket without a prefix. Prefix strings must end with '/'.

## 4.17 Custom Meta Data

RSTOR Space has the ability to have custom meta data applied to objects. Here is an example of applying meta data with the API and commonly used AWS CLI tool.

```
aws --profile=local --endpoint="http://demo.rstorcloud.io" s3 cp
octocats/original.png s3://testbucket/ --metadata Test=Pluto
```

And then to retrieve the object with the custom meta data:

```
% aws --profile=local --endpoint="https://demo.rstorcloud.io"
s3api head-object --bucket=testbucket --key=original.png
{
    "AcceptRanges": "bytes",
    "LastModified": "Mon, 23 Mar 2020 19:42:16 GMT",
```

```
    "ContentLength": 50758,
    "ETag": "\"2ec6d2a9d4888c31c5ae802e8cc9efd6\"",
    "VersionId": "01E44ENHPQKG54VAFC8FB2RR10",
    "ContentType": "image/png",
    "Metadata": {
        "test": "Pluto"
    }
}
```

## 4.18 Object Byte Range Access

RSTOR Space supports reading with Byte Range and updating using MultiPart upload copy. The next few sections display an example for each.

### 4.18.1 Byte Range Read

```
[testing@vbook:~]
% aws --profile pre --endpoint https://s3.demo.rstorcloud.io' s3api
get-object --bucket simplebucket-testing --key='poignant.txt'
'poignant2.txt' --range bytes=0-500
{
    "AcceptRanges": "bytes",
    "LastModified": "Mon, 30 Mar 2020 18:59:08 GMT",
    "ContentLength": 501,
    "ETag": "\"2909cf38d280b0d88586740cd86e45e6\"",
    "VersionId": "01E4PCZK3W672JT0X74WK4BAHR",
    "ContentRange": "bytes 0-500/1476",
    "Metadata": {}
}
[testing@vbook:~]

% cat poignant2.txt
Pretend that you've opened this book (although you probably have opened
this book), just to find a huge onion right in the middle crease of the
book. (The manufacturer of the book has included the onion at my
request.) So you're like, "Wow, this book comes with an onion!" (Even
if you don't particularly like onions, I'm sure you can appreciate the
logistics of shipping any sort of produce discreetly inside of an
alleged programming manual.) Then you ask yourself, "Wait a minute. I
```

### 4.18.2 Object Update

Partially updating an object can be very useful in cases where a full object updated is not desired, such as with large objects. While the standard S3 protocol does not offer a byte-range update, there are a few use cases for updating an object with Multipart copy. Once a file has been written, it cannot be modified, only overwritten (as a whole) with a newer version.

The closest behavior to an object update may be obtained by using the UploadPartCopy API to create a new version of the file starting from the old one and from the changed parts. Note however that this still triggers a full copy of the file.

- If you want to copy a large file (there's a size limit on the PutObjectCopy, and there might be timeouts on the client side), you can copy the file in parts, then use CompleteMultipartUpload to obtain the complete file. **aws s3 cp does that in some cases.*
- If you want to append to the end of a file without downloading and uploading it again.
- If you want to change some bytes in the middle of a file, you can UploadPartCopy the part before, then UploadPart the changed part, and then UploadPartCopy the part after.

To clarify, you cannot change a part in a file that is already in S3, but you can create a new file (using multipart uploads), and tell the server you want to use some range of the old file as a "part" instead of uploading from your computer. Having immutable versions helps us greatly when replicating a file as there is no risk of conflicts between the same file on two datacenters, or replicating an older version over a newer one, or the client reading a partially updated version. This is an example using multipart upload:

```
% aws --profile pre --endpoint 'https://s3.demo.rstorcloud.io' s3api
create-multipart-upload --bucket simplebucket-testing --key='some-
multipart-upload'
{
    "Bucket": "simplebucket-testing",
    "Key": "some-multipart-upload",
    "UploadId": "7xDckkL3iSJAvFBvZw5GmJWKkjfCxFXE3tAPalCVHOQZhaESNu"
}%


aws --profile pre --endpoint 'https://s3.demo.rstorcloud.io' s3api
upload-part --bucket simplebucket-testing --key='some-multipart-
upload' --upload-id 7xDckkL3iSJAvFBvZw5GmJWKkjfCxFXE3tAPalCVHOQZhaESNu
--part-number 1 --body ~/Downloads/poignant-guide.pdf
```

```
{
    "ETag": "\"61062673c8b37d22354523849923bbd2\""
}%


aws --profile pre --endpoint 'https://s3.demo.rstorcloud.io' s3api
upload-part --bucket simplebucket-testing --key='some-multipart-
upload' --upload-id 7xDckkL3iSJAvFBvZw5GmJWKkjfCxFXE3tAPalCVHOQZhaESNu
--part-number 2 --body ~/Downloads/poignant-guide.pdf


{
    "ETag": "\"61062673c8b37d22354523849923bbd2\""
}%


aws --profile pre --endpoint 'https://s3.demo.rstorcloud.io' s3api
complete-multipart-upload --bucket simplebucket-testing --key='some-
multipart-upload' --upload-id
7xDckkL3iSJAvFBvZw5GmJWKkjfCxFXE3tAPalCVHOQZhaESNu --multipart-upload
'Parts=[{ETag="\"61062673c8b37d22354523849923bbd2\"",PartNumber=1},{ET
ag="\"61062673c8b37d22354523849923bbd2\"",PartNumber=2}]'


{
    "VersionId": "01E4PDSTT5DBJC5CFE57CTYG7E",
    "Location": "",
    "Bucket": "simplebucket-testing",
    "Key": "some-multipart-upload",
    "ETag": "\"05c7f4ca2b92bedafcab99a3094c435b-2\""
}
```

## 4.19 RProtect: Customer

RProtect is a feature that enables more-granular whitelisting to resources for both Reseller and Customer personalities.

RProtect can explicitly ensure that no one, except select IP addresses or a range of IP addresses have access to a specific resource.

* *It can also be used to remove all restrictions by using 0.0.0.0 as the whitelisted IP address.*

The RProtect feature may be enabled for a Reseller or fully disabled, meaning that anyone can access the buckets within the accounts. This feature is listed in the main menu bar as well as being available via the RSTOR Space API.

1. For Customer accounts through the Web GUI, select RProtect from the menu bar.
2. To add a new rule, select "+ ADD RULE" in the upper right corner.

   The "Subnet" option may exist of a single IP address or range of IP addresses. While there are no limitations on the size of the range, we have found the best practice is to minimize the range as small as possible.

   Confining the address range, a host address or a /29 address range is the best scenario (8 addresses, 5 usable), where there might be a /22 (1024 IP addresses). In general, anything bigger than a /22 (/21 on down) may not be secure enough to meet security policy.

   You may also set the RSTOR Space regions in which you want this whitelist to apply for this account, along with a time to live for the new IP address or range of addresses. Zero (0) days is the default with no time limitations.

   There is also an option to add a 144-character note for future references; perhaps the reason, customer, support request number or location of the whitelist, as an example.
3. The single IP address goes into a pending state post submission of request.
4. Customer Account RProtect requests are approved by the Reseller Account administration. See the RProtect: Reseller section in this guide for how to APPROVE pending requests.

   When requesting an RProtect IP address or range of addresses from a Customer account, the Reseller account will need to approve the request. Once the Reseller administrator approves, the request, it will change from Pending to Active.

   **The Customer/sub-tenant account can view each of the RProtect requests, whereas the Reseller will be able to manage such as deleting or rejecting.*

# 5 Appendix

## 5.1 Reseller Support

In the Reseller account, there is the ability to perform a speed test. This simplified test tool generates some random data and sends it to an RSTOR Space access point.

**It does not implement multipart upload and is not intended for achieving maximum capabilities.*

If you are using a CLI or another tool to compare benchmarks with Speed Test, we recommend setting signatures to S3v2 and set multipart upload (MPU) to 1.

1. Under the Support tab in the menu bar, click on “RUN TEST” under the Speed test square.
2. Select a region to test. The current RSTOR regions include us-east-1, us-central-1, and us-west-1.
3. A series of download and upload tests will be run and once it is finished you will be able to see both the download and upload speed.

## 5.2 Subtenant Support

To receive further support, click on Support in the top menu bar. Here you can (1) access and explore our API documentation, or (2) contact us and we will get back to you via email as soon as possible.