Providing Object Storage Using the Amazon S3 API

Objectives

After completing this section, you should be able to configure the RADOS Gateway to provide access to object storage compatible with the Amazon S3 API, and manage objects stored using that API.

Amazon S3 API in RADOS Gateway

The Amazon S3 API enables developers to manage object storage resources using an Amazon S3 compatible interface. Applications implemented with the S3 API can inter-operate with other S3-compatible object storage services besides the RADOS Gateway, and migrate storage from other locations to your Ceph storage cluster. In a hybrid cloud environment, you can configure your applications to use different authentication keys, regions, and vendor services to mix private enterprise and public cloud resources and storage locations seamlessly using the same API.

The Amazon S3 interface defines the namespace in which objects are stored as a *bucket*. To access and manage objects and buckets using the S3 API, applications use RADOS Gateway users for authentication. Each user has an *access key* that identifies the user and a *secret key* that authenticates the user.

There are object and metadata size limits to consider when using the Amazon S3 API:

- An object size is between a minimum of OB and a maximum of 5 TB.
- The maximum size is 5GB in a single upload operation.
- Upload objects larger than 100MB by using the *multipart upload* capability.
- The maximum metadata size is 16,000 bytes in a single HTTP request.

Creating a User for the Amazon S3 API

Create the needed RADOS Gateway users first, then use them to authenticate Amazon S3 clients with the gateway. Use the `radosgw-admin user create` command to create RADOS Gateway users.

When creating RADOS Gateway users, both the `--uid` and `--display-name` options are required, and specify a unique account name and a human friendly display name. Use the `--access-key` and `--secret` options to specify a custom AWS account and secret key for the RADOS user.

```
[ceph: root@node /]# radosgw-admin user create --uid=testuser \
  --display-name="Test User" --email= test@example.com \
  --access-key=12345 --secret=67890
...output omitted...
    "keys": [
        {
            "user": "testuser",
            "access_key": "12345",
```