# s3 make bucket (create bucket)

aws s3 mb s3://tgsbucket --region us-west-2

# s3 remove bucket

aws s3 rb s3://tgsbucket

aws s3 rb s3://tgsbucket --force

# s3 ls commands

aws s3 ls

aws s3 ls s3://tgsbucket

aws s3 ls s3://tgsbucket --recursive

aws s3 ls s3://tgsbucket --recursive --human-readable --summarize

# s3 cp commands

aws s3 cp getdata.php s3://tgsbucket

aws s3 cp /local/dir/data s3://tgsbucket --recursive

aws s3 cp s3://tgsbucket/getdata.php /local/dir/data

aws s3 cp s3://tgsbucket/ /local/dir/data --recursive

aws s3 cp s3://tgsbucket/init.xml s3://backup-bucket

aws s3 cp s3://tgsbucket s3://backup-bucket --recursive

# s3 mv commands

aws s3 mv source.json s3://tgsbucket

aws s3 mv s3://tgsbucket/getdata.php /home/project

aws s3 mv s3://tgsbucket/source.json s3://backup-bucket

aws s3 mv /local/dir/data s3://tgsbucket/data --recursive

aws s3 mv s3://tgsbucket s3://backup-bucket --recursive

# s3 rm commands

aws s3 rm s3://tgsbucket/queries.txt

aws s3 rm s3://tgsbucket --recursive

# s3 sync commands

aws s3 sync backup s3://tgsbucket

aws s3 sync s3://tgsbucket/backup /tmp/backup

aws s3 sync s3://tgsbucket s3://backup-bucket

# s3 bucket website

aws s3 website s3://tgsbucket/ --index-document index.html --error-document error.html

# s3 presign url (default 3600 seconds)

aws s3 presign s3://tgsbucket/dnsrecords.txt

aws s3 presign s3://tgsbucket/dnsrecords.txt --expires-in 60

1. Create New S3 Bucket

Use mb option for this. mb stands for Make Bucket.

The following will create a new S3 bucket

$ aws s3 mb s3://tgsbucket

make\_bucket: tgsbucket

In the above example, the bucket is created in the us-east-1 region, as that is what is specified in the user’s config file as shown below.

$ cat ~/.aws/config

[profile Sandy]

region = us-east-1

If the bucket already exists, and you own the bucket, you’ll get the following error message.

$ aws s3 mb s3://tgsbucket

make\_bucket failed: s3://tgsbucket An error occurred (BucketAlreadyOwnedByYou) when calling the CreateBucket operation: Your previous request to create the named bucket succeeded and you already own it.

If the bucket already exists, but owned by some other user, you’ll get the following error message.

$ aws s3 mb s3://paloalto

The bucket namespace is shared by all users of the system. Please select a different name and try again.

Under some situation, you might also get the following error message.

$ aws s3 mb s3://demo-bucket

make\_bucket failed: s3://demo-bucket An error occurred (IllegalLocationConstrainConstraintException) when calling the CreateBucket operation: The unspecified location constraint is incompatible for the region specific endpoint this request was sent to.)

2. Create New S3 Bucket – Different Region

To create a bucket in a specific region (different than the one from your config file), then use the –region option as shown below.

$ aws s3 mb s3://tgsbucket --region us-west-2

make\_bucket: tgsbucket

3. Delete S3 Bucket (That is empty)

Use rb option for this. rb stands for remove bucket.

The following deletes the given bucket.

$ aws s3 rb s3://tgsbucket

remove\_bucket: tgsbucket

If the bucket you are trying to delete doesn’t exists, you’ll get the following error message.

$ aws s3 rb s3://tgsbucket1

remove\_bucket failed: s3://tgsbucket1 An error occurred (NoSuchBucket) when calling the DeleteBucket operation: The specified bucket does not exist

4. Delete S3 Bucket (And all its objects)

If the bucket contains some object, you’ll get the following error message:

$ aws s3 rb s3://tgsbucket

remove\_bucket failed: s3://tgsbucket An error occurred (BucketNotEmpty) when calling the DeleteBucket operation: The bucket you tried to delete is not empty

To delete a bucket along with all its objects, use the –force option as shown below.

$ aws s3 rb s3://tgsbucket --force

delete: s3://tgsbucket/demo/getdata.php

delete: s3://tgsbucket/ipallow.txt

delete: s3://tgsbucket/demo/servers.txt

delete: s3://tgsbucket/demo/

remove\_bucket: tgsbucket

5. List All S3 Buckets

To view all the buckets owned by the user, execute the following ls command.

$ aws s3 ls

2019-02-06 11:38:55 tgsbucket

2018-12-18 18:02:27 etclinux

2018-12-08 18:05:15 readynas

..

..

In the above output, the timestamp is the date the bucket was created. The timezone was adjusted to be displayed to your laptop’s timezone.

The following command is same as the above:

aws s3 ls s3://

6. List All Objects in a Bucket

The following command displays all objects and prefixes under the tgsbucket.

$ aws s3 ls s3://tgsbucket

PRE config/

PRE data/

2019-04-07 11:38:20 13 getdata.php

2019-04-07 11:38:20 2546 ipallow.php

2019-04-07 11:38:20 9 license.php

2019-04-07 11:38:20 3677 servers.txt

In the above output:

Inside the tgsbucket, there are two folders config and data (indicated by PRE)

PRE stands for Prefix of an S3 object.

Inside the tgsbucket, we have 4 files at the / level

The timestamp is when the file was created

The 2nd column display the size of the S3 object

Note: The above output doesn’t display the content of sub-folders config and data

7. List all Objects in a Bucket Recursively

To display all the objects recursively including the content of the sub-folders, execute the following command.

$ aws s3 ls s3://tgsbucket --recursive

2019-04-07 11:38:19 2777 config/init.xml

2019-04-07 11:38:20 52 config/support.txt

2019-04-07 11:38:20 1758 data/database.txt

2019-04-07 11:38:20 13 getdata.php

2019-04-07 11:38:20 2546 ipallow.php

2019-04-07 11:38:20 9 license.php

2019-04-07 11:38:20 3677 servers.txt

Note: When you are listing all the files, notice how there is no PRE indicator in the 2nd column for the folders.

8. Total Size of All Objects in a S3 Bucket

You can identify the total size of all the files in your S3 bucket by using the combination of following three options: recursive, human-readable, summarize

Note: The following displays both total file size in the S3 bucket, and the total number of files in the s3 bucket

$ aws s3 ls s3://tgsbucket --recursive --human-readable --summarize

2019-04-07 11:38:19 2.7 KiB config/init.xml

2019-04-07 11:38:20 52 Bytes config/support.txt

2019-04-07 11:38:20 1.7 KiB data/database.txt

2019-04-07 11:38:20 13 Bytes getdata.php

2019-04-07 11:38:20 2.5 KiB ipallow.php

2019-04-07 11:38:20 9 Bytes license.php

2019-04-07 11:38:20 3.6 KiB servers.txt

Total Objects: 7

Total Size: 10.6 KiB

In the above output:

recursive option make sure that it displays all the files in the s3 bucket including sub-folders

human-readable displays the size of the file in readable format. Possible values you’ll see in the 2nd column for the size are: Bytes/MiB/KiB/GiB/TiB/PiB/EiB

summarize options make sure to display the last two lines in the above output. This indicates the total number of objects in the S3 bucket and the total size of all those objects

9. Request Payer Listing

If a specific bucket is configured as requester pays buckets, then if you are accessing objects in that bucket, you understand that you are responsible for the payment of that request access. In this case, bucket owner doesn’t have to pay for the access.

To indicate this in your ls command, you’ll have to specify –request-payer option as shown below.

$ aws s3 ls s3://tgsbucket --recursive --request-payer requester

2019-04-07 11:38:19 2777 config/init.xml

2019-04-07 11:38:20 52 config/support.txt

2019-04-07 11:38:20 1758 data/database.txt

2019-04-07 11:38:20 13 getdata.php

2019-04-07 11:38:20 2546 ipallow.php

2019-04-07 11:38:20 9 license.php

2019-04-07 11:38:20 3677 servers.txt

For signed URL, make sure to include x-amz-request-payer=requester in the request

10. Copy Local File to S3 Bucket

In the following example, we are copying getdata.php file from local laptop to S3 bucket.

$ aws s3 cp getdata.php s3://tgsbucket

upload: ./getdata.php to s3://tgsbucket/getdata.php

If you want to copy the getdata.php to a S3 bucket with a different name, do the following

$ aws s3 cp getdata.php s3://tgsbucket/getdata-new.php

upload: ./getdata.php to s3://tgsbucket/getdata-new.php

For the local file, you can also specify the full path as shown below.

$ aws s3 cp /home/project/getdata.php s3://tgsbucket

upload: ../../home/project/getdata.php to s3://tgsbucket/getdata.php

11. Copy Local Folder with all Files to S3 Bucket

In this example, we are copying all the files from the “data” folder that is under /home/projects directory to S3 bucket

$ cd /home/projects

$ aws s3 cp data s3://tgsbucket --recursive

upload: data/parameters.txt to s3://tgsbucket/parameters.txt

upload: data/common.txt to s3://tgsbucket/common.txt

..

In the above example, note that only the files from the local data/ folder is getting uploaded. Not the folder “data” itself

If you like to upload the data folder from local to s3 bucket as data folder, then specify the folder name after the bucket name as shown below.

$ aws s3 cp data s3://tgsbucket/data --recursive

upload: data/parameters.txt to s3://tgsbucket/data/parameters.txt

upload: data/common.txt to s3://tgsbucket/data/common.txt

..

..

12. Download a File from S3 Bucket

To download a specific file from an S3 bucket do the following. The following copies getdata.php from the given s3 bucket to the current directory.

$ aws s3 cp s3://tgsbucket/getdata.php .

download: s3://tgsbucket/getdata.php to ./getdata.php

You can download the file to the local machine with in a different name as shown below.

$ aws s3 cp s3://tgsbucket/getdata.php getdata-local.php

download: s3://tgsbucket/getdata.php to ./getdata-local.php

Download the file from S3 bucket to a specific folder in local machine as shown below. The following will download getdata.php file to /home/project folder on local machine.

$ aws s3 cp s3://tgsbucket/getdata.php /home/project/

download: s3://tgsbucket/getdata.php to ../../home/project/getdata.php

13. Download All Files Recursively from a S3 Bucket (Using Copy)

The following will download all the files from the given bucket to the current directory on your laptop.

$ aws s3 cp s3://tgsbucket/ . --recursive

download: s3://tgsbucket/getdata.php to ./getdata.php

download: s3://tgsbucket/config/init.xml ./config/init.xml

..

If you want to download all the files from a S3 bucket to a specific folder locally, please specify the full path of the local directory as shown below.

$ aws s3 cp s3://tgsbucket/ /home/projects/tgsbucket --recursive

download: s3://tgsbucket/getdata.php to ../../home/projects/tgsbucket/getdata.php

download: s3://tgsbucket/config/init.xml to ../../home/projects/tgsbucket/config/init.xml

..

In the above command, if the tgsbucket folder doesn’t exists under /home/projects, it will create it automatically.

14. Copy a File from One Bucket to Another Bucket

The following command will copy the config/init.xml from tgsbucket to backup bucket as shown below.

$ aws s3 cp s3://tgsbucket/config/init.xml s3://backup-bucket

copy: s3://tgsbucket/config/init.xml to s3://backup-bucket/init.xml

In the above example, eventhough init.xml file was under config folder in the source bucket, on the destination bucket, it copied the init.xml file to the top-level / in the backup-bucket.

If you want to copy the same folder from source and destination along with the file, specify the folder name in the desintation bucketas shown below.

$ aws s3 cp s3://tgsbucket/config/init.xml s3://backup-bucket/config

copy: s3://tgsbucket/config/init.xml to s3://backup-bucket/config/init.xml

If the destination bucket doesn’t exist, you’ll get the following error message.

$ aws s3 cp s3://tgsbucket/test.txt s3://backup-bucket-777

copy failed: s3://tgsbucket/test.txt to s3://backup-bucket-777/test.txt An error occurred (NoSuchBucket) when calling the CopyObject operation: The specified bucket does not exist

15. Copy All Files Recursively from One Bucket to Another

The following will copy all the files from the source bucket including files under sub-folders to the destination bucket.

$ aws s3 cp s3://tgsbucket s3://backup-bucket --recursive

copy: s3://tgsbucket/getdata.php to s3://backup-bucket/getdata.php

copy: s3://tgsbucket/config/init.xml s3://backup-bucket/config/init.xml

..

16. Move a File from Local to S3 Bucket

When you move file from Local machine to S3 bucket, as you would expect, the file will be physically moved from local machine to the S3 bucket.

$ ls -l source.json

-rw-r--r-- 1 ramesh sysadmin 1404 Apr 2 13:25 source.json

$ aws s3 mv source.json s3://tgsbucket

move: ./source.json to s3://tgsbucket/source.json

As you see the file doesn’t exists on the local machine after the move. Its only on S3 bucket now.

$ ls -l source.json

ls: source.json: No such file or directory

17. Move a File from S3 Bucket to Local

The following is reverse of the previou example. Here, the file will be moved from S3 bucket to local machine.

As you see below, the file now exists on the s3 bucket.

$ aws s3 ls s3://tgsbucket/getdata.php

2019-04-06 06:24:29 1758 getdata.php

Move the file from S3 bucket to /home/project directory on local machine.

$ aws s3 mv s3://tgsbucket/getdata.php /home/project

move: s3://tgsbucket/getdata.php to ../../../home/project/getdata.php

After the move, the file doesn’t exists on S3 bucketanymore.

$ aws s3 ls s3://tgsbucket/getdata.php

18. Move a File from One S3 Bucket to Another S3 Bucket

Before the move, the file source.json is in tgsbucket.

$ aws s3 ls s3://tgsbucket/source.json

2019-04-06 06:51:39 1404 source.json

This file is not in backup-bucket.

$ aws s3 ls s3://backup-bucket/source.json

$

Move the file from tgsbucketto backup-bucket.

$ aws s3 mv s3://tgsbucket/source.json s3://backup-bucket

move: s3://tgsbucket/source.json to s3://backup-bucket/source.json

Now, the file is only on the backup-bucket.

$ aws s3 ls s3://tgsbucket/source.json

$

$ aws s3 ls s3://backup-bucket/source.json

2019-04-06 06:56:00 1404 source.json

19. Move All Files from a Local Folder to S3 Bucket

In this example, the following files are under data folder.

$ ls -1 data

dnsrecords.txt

parameters.txt

dev-setup.txt

error.txt

The following moves all the files in the data directory on local machine to tgsbucket

$ aws s3 mv data s3://tgsbucket/data --recursive

move: data/dnsrecords.txt to s3://tgsbucket/data/dnsrecords.txt

move: data/parameters.txt to s3://tgsbucket/data/parameters.txt

move: data/dev-setup.txt to s3://tgsbucket/data/dev-setup.txt

move: data/error.txt to s3://tgsbucket/data/error.txt

20. Move All Files from S3 Bucket to Local Folder

In this example, the localdata folder is currently empty.

$ ls -1 localdata

$

The following will move all the files in the S3 bucketunder data folder to localdata folder on your local machine.

$ aws s3 mv s3://tgsbucket/data/ localdata --recursive

move: s3://tgsbucket/data/dnsrecords.txt to localdata/dnsrecords.txt

move: s3://tgsbucket/data/parameters.txt to localdata/parameters.txt

move: s3://tgsbucket/data/dev-setup.txt to localdata/dev-setup.txt

move: s3://tgsbucket/data/error.txt to localdata/error.txt

Here is the output after the above move.

$ aws s3 ls s3://tgsbucket/data/

$

$ ls -1 localdata

dnsrecords.txt

parameters.txt

dev-setup.txt

error.txt

21. Move All Files from One S3 Bucket to Another S3 Bucket

Use the recursive option to move all files from one bucket to another as shown below.

$ aws s3 mv s3://tgsbucket s3://backup-bucket --recursive

move: s3://tgsbucket/dev-setup.txt to s3://backup-bucket/dev-setup.txt

move: s3://tgsbucket/dnsrecords.txt to s3://backup-bucket/dnsrecords.txt

move: s3://tgsbucket/error.txt to s3://backup-bucket/error.txt

move: s3://tgsbucket/parameters.txt to s3://backup-bucket/parameters.txt

22. Delete a File from S3 Bucket

To delete a specific file from a S3 bucket, use the rm option as shown below. The following will delete the queries.txt file from the given S3 bucket.

$ aws s3 rm s3://tgsbucket/queries.txt

delete: s3://tgsbucket/queries.txt

23. Delete All Objects from S3 buckets

When you specify rm option just with a bucket name, it doesn’t do anything. This will not delete any file from the bucket.

aws s3 rm s3://tgsbucket

To delete all the files from a S3 bucket, use the –recursive option as show nbelow.

$ aws s3 rm s3://tgsbucket --recursive

delete: s3://tgsbucket/dnsrecords.txt

delete: s3://tgsbucket/common.txt

delete: s3://tgsbucket/parameters.txt

delete: s3://tgsbucket/config/init.xml

..

24. Sync files from Laptop to S3 Bucket

When you use sync command, it will recursively copies only the new or updated files from the source directory to the destination.

The following will sync the files from backup directory in local machine to the tgsbucket.

$ aws s3 sync backup s3://tgsbucket

upload: backup/docker.sh to s3://tgsbucket/docker.sh

upload: backup/address.txt to s3://tgsbucket/address.txt

upload: backup/display.py to s3://tgsbucket/display.py

upload: backup/getdata.php to s3://tgsbucket/getdata.php

If you want to sync it to a subfolder called backup on the S3 bucket, then include the folder name in the s3 bucket as shown below.

$ aws s3 sync backup s3://tgsbucket/backup

upload: backup/docker.sh to s3://tgsbucket/backup/docker.sh

upload: backup/address.txt to s3://tgsbucket/backup/address.txt

upload: backup/display.py to s3://tgsbucket/backup/display.py

upload: backup/getdata.php to s3://tgsbucket/backup/getdata.php

Once you do the sync once, if you run the command immediately again, it will not do anything, as there is no new or updated files on the local backup directory.

$ aws s3 sync backup s3://tgsbucket/backup

$

Let us create a new file on the local machine for testing.

echo "New file" > backup/newfile.txt

Now when you execute the sync, it will sync only this new file to the S3 bucket.

$ aws s3 sync backup s3://tgsbucket/backup

upload: backup/newfile.txt to s3://tgsbucket/backup/newfile.txt

25. Sync File from S3 bucket to Local

This is reverse of the previous example. Here, we are syncing the files from the S3 bucket to the local machine.

$ aws s3 sync s3://tgsbucket/backup /tmp/backup

download: s3://tgsbucket/backup/docker.sh to ../../tmp/backup/docker.sh

download: s3://tgsbucket/backup/display.py to ../../tmp/backup/display.py

download: s3://tgsbucket/backup/newfile.txt to ../../tmp/backup/newfile.txt

download: s3://tgsbucket/backup/getdata.php to ../../tmp/backup/getdata.php

download: s3://tgsbucket/backup/address.txt to ../../tmp/backup/address.txt

26. Sync Files from one S3 Bucket to Another S3 Bucket

The following example syncs the files from one tgsbucket to backup-bucket

$ aws s3 sync s3://tgsbucket s3://backup-bucket

copy: s3://tgsbucket/backup/newfile.txt to s3://backup-bucket/backup/newfile.txt

copy: s3://tgsbucket/backup/display.py to s3://backup-bucket/backup/display.py

copy: s3://tgsbucket/backup/docker.sh to s3://backup-bucket/backup/docker.sh

copy: s3://tgsbucket/backup/address.txt to s3://backup-bucket/backup/address.txt

copy: s3://tgsbucket/backup/getdata.php to s3://backup-bucket/backup/getdata.php

27. Set S3 bucket as a website

You can also make S3 bucket to host a static website as shown below. For this, you need to specify both the index and error document.

aws s3 website s3://tgsbucket/ --index-document index.html --error-document error.html

This bucket is in us-east-1 region. So, once you’ve done the above, you can access the tgsbucket as a website using the following URL: http://tgsbucket.s3-website-us-east-1.amazonaws.com/

For this to work properly, make sure public access is set on this S3 bucket, as this acts as a website now.

28. Presign URL of S3 Object for Temporary Access

When you presign a URL for an S3 file, anyone who was given this URL can retrieve the S3 file with a HTTP GET request.

For example, if you want to give access to the dnsrecords.txt file to someone temporarily, presign this specific S3 object as shown below.

$ aws s3 presign s3://tgsbucket/dnsrecords.txt

https://tgsbucket.s3.amazonaws.com/error.txt?AWSAccessKeyId=AAAAAAAAAAAAAAAAAAAA&Expires=1111111111&Signature=ooooooooooo%2Babcdefghijlimmm%3A

The output of the above command will be a HTTPS url, which you can hand it out someone who should be able to download the dnsrecords.txt file from your S3 bucket.

The above URL will be valid by default for 3600 seconds (1 hour).

If you want to specify a short expirty time, use the following expires-in option. The following will create a presigned URL that is valid only for 1 minute.

–expires-in (integer) Number of seconds until the pre-signed URL expires. Default is 3600 seconds.

$ aws s3 presign s3://tgsbucket/dnsrecords.txt --expires-in 60

https://tgsbucket.s3.amazonaws.com/error.txt?AWSAccessKeyId=AAAAAAAAAAAAAAAAAAAA&Expires=1111111111&Signature=ooooooooooo%2Babcdefghijlimmm%3A

If someone tries to access the URL after the expiry time, they’ll see the following AccessDenied message.

<Error>

<Code>AccessDenied</Code>

<Message>Request has expired</Message>

<Expires>2019-04-07T11:38:12Z</Expires>

<ServerTime>2019-04-07T11:38:21Z</ServerTime>

<RequestId>1111111111111111</RequestId>

<HostId>

mmmmmmmmmm/ggggggggg

</HostId>

</Error>