

Contents

How do I get a <i>new</i> AWS S3 bucket?	2
How do I get approved to be owner of a bucket?	2
How do I create a new bucket as a bucket owner?	4
To create a Project:	4
To create a Bucket:	6
How Do I Request/Provide Access to an Existing AWS S3 Bucket?	6
How do I Access my S3 Keys or Credentials for a Bucket?	8
How do I Access my Bucket and Move/Copy Data?	9
Considerations for Copying, Moving and Accessing Data in AWS S3 Storage	10
Using ExpanDrive (Windows, MAC OS, <i>Linux</i>)	11
Using an S3 Browser Windows Client	14
To Create an S3 Browser Account and Associated Buckets	14
To Organize, Move or Copy Data to/from the AWS Bucket via the S3 Browser	15
How to Use Cyberduck	17
How to Install Cyberduck on your Mac	17
How to Install Cyberduck on your Windows PC	18
How to Connect and Transfer Files with Cyberduck	18
How to Transfer Files and Use the Browser Window	21
Other S3 Client Connectivity Options	24
How to Install and Use rclone on Linux and Windows	24
.....	26
S3FS Mounted File System Based S3 Client (Linux & MAC OS only)	26
Using AWS CLI	31

Several different applications are required to work with the DHTS AWS Object Storage:

1. **ServiceNow** <https://duke.service-now.com/sp?id=index>:
 - to request ownership privileges for bucket creation and management (see below for steps)
 - to request to change an owner on a bucket
 - to request to change the Cost Center on a bucket
 - to request to move large files of data (> 5 TB total at a time) using Starfish
2. **Duke OIT Group Manager**: to grant permission for people (owners) to setup and manage buckets via Portunus
3. **Portunus** <https://portunus-eso.duhs.duke.edu/portunus/>:
 - to create a new bucket or set of buckets (Project) assigned to a Cost Center
 - to add or remove users (read/write or read-only) to buckets
4. **Starfish**: for DHTS to move large volumes of data (>5TB) to/from Isilon File Shares, servers and object storage on behalf of a bucket owner
5. **S3-Compatible Clients or Command Line Interfaces (CLIs)**: to access, view, and manage data in your bucket or to/from your bucket

How do I get a new AWS S3 bucket?

Only individuals with permission to be bucket owners in Portunus can create new buckets.

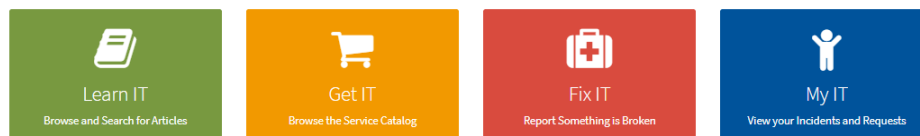
Once permission is granted, the bucket owner can use **Portunus** to create a bucket and add users to the bucket.

NOTE: A bucket owner should be the data owner (e.g., PI) or designated delegate of the data owner (e.g., Lab Manager) who is responsible for providing and removing user access to a bucket.)

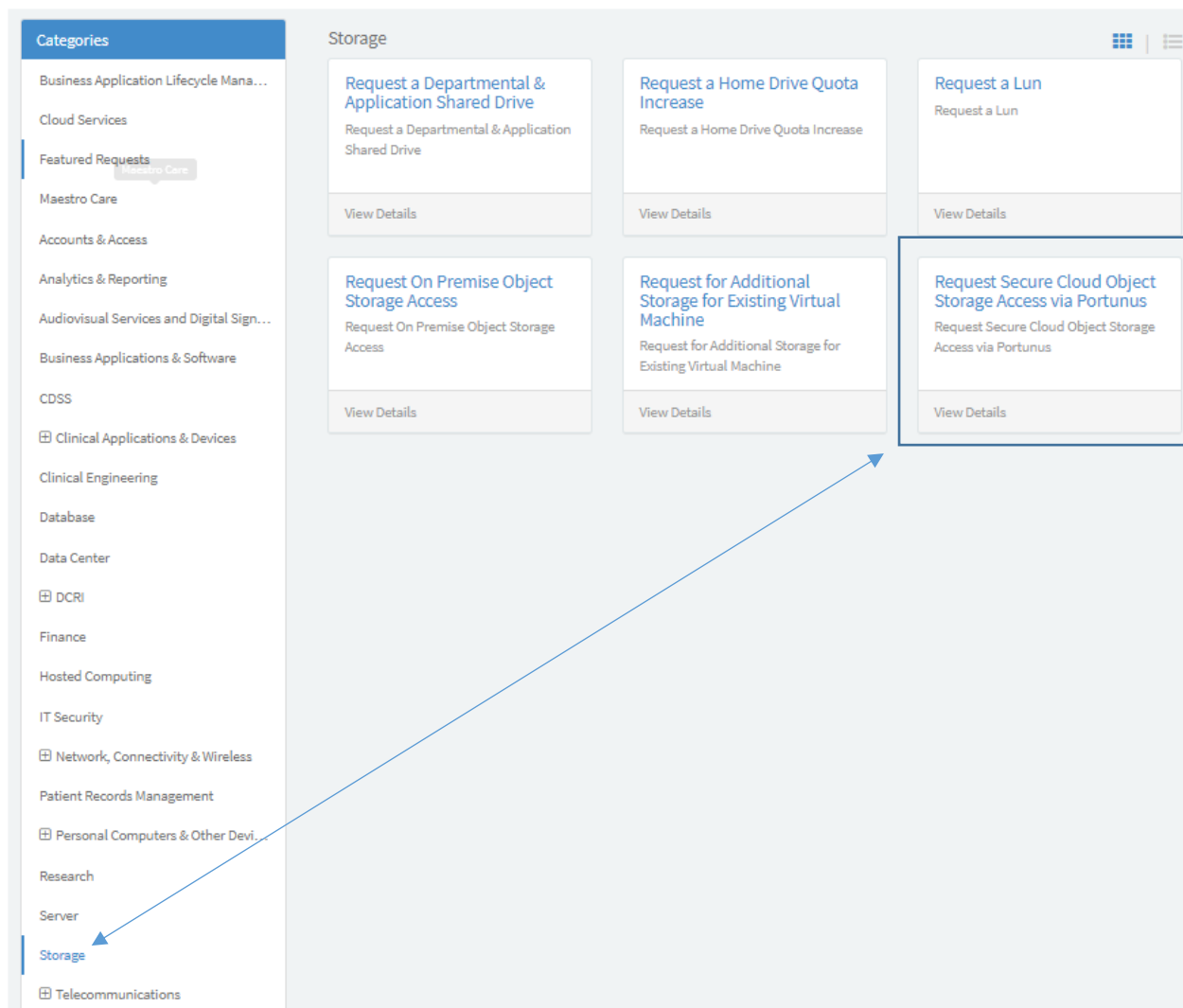
How do I get approved to be owner of a bucket?

A. If you are the **owner of a bucket** [and have not previously been provided owner privileges in Portunus],

1. Go to Duke ServiceNow: <https://duke.service-now.com/sp?id=index> and select “**Get IT**”.



2. Under **Categories** on the left side of the screen, select “**Show More**” and then “**Storage**”.
3. Then under Storage options, select the box, “**Request Secure Cloud Object Storage Access via Portunus**”.



4. In the Requester Information form, enter the [Name of the Duke user who is requesting owner rights](#).
5. Contact information for the individual and that of his/her manager should auto-populate.
6. You can change the name of the Approver, if necessary, to an appropriate person other than the individual's manager.
7. Select [Request Now](#).
8. The Approver will receive an email notification from ServiceNow to approve the request.
9. Once the request is approved, DHTS will add the Requester's Name to the Grouper group in Group Manager for owner permissions to be added in Portunus.

Request Secure Cloud Object Storage Access via Portunus

Request Secure Cloud Object Storage Access via Portunus

Requester Information

* Requested For

Duke user who needs to be added to the Grouper group.

Terri West (tth52)

* Requester's Best Contact Phone Number

+1 919 613 2611

Approver Information

* Requester's Manager

Cory Ennis (ennis009)

* Manager Contact Phone Number

+1 919 668 8284

Application Information

Will an Application depend on this Cloud Storage Bucket?

☐ Yes ☒ No

Add attachments

Request Now

Favorite Request

How do I create a new bucket as a bucket owner?

Once you are setup with bucket owner permissions in **Portunus**, you will use Portunus to:

1. First create a **Project** with an associated **Cost Center**; then
2. Create a **bucket(s)**; then
3. Add and remove **users** (by netid@duke.edu) with either Read-Only or Read/Write access to buckets; then
4. Generate **Keys** (credentials) for the bucket(s) in the Project.

NOTE: *Project and Bucket Names CANNOT be changed once they are in place.*

Instead, you would have to create a NEW bucket and/or project and move the data to the new bucket.

Projects are:

- associated to a **Cost Center**;
- a collection for one or more buckets that share the same Cost Center AND share the same set of S3 credentials unique for each user;
- analogous to "Accounts" in the S3 Browser Client or "Collections" in Cyberduck.

Buckets are:

- analogous to folders in file shares, EXCEPT anyone who has access to a bucket can see EVERYTHING in the bucket. (So if you need to limit access, you will want to create different buckets for different data.)

To create a Project:

1. Go to **Portunus** <https://portunus-eso.duhs.duke.edu/portunus/>
2. Select **Projects** in the menu at the top.

3. On the right hand side of the screen, under **New Project**, enter:

New Project

Project Owner
tlh52@duke.edu

Name
TEST-AWS

Entity
SOM

Cost Center
1571187

Department/Center Requested For
Pharmacology & Cancer

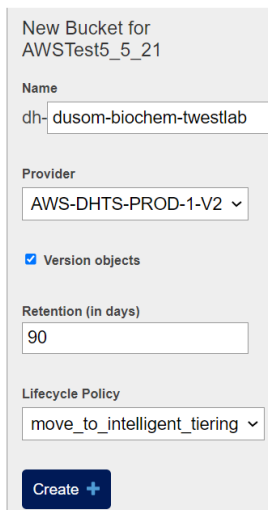
Tracking Cost Center
441-1162

Admin Group
Support-Academic DHTS

- Name** of the Project (should be something to explain the types of buckets in the project).
 - Entity**: select the appropriate area of Duke in which you reside; i.e., SOM, SON, or UNIV.
 - Cost Center** – this will be the same Cost Center for all the buckets created in that Project.
 - This will be used to bill for charges.
 - For SOM and SON, the centralized cost center will automatically populate.
 - You can look up cost centers by typing in the name or just enter the numerical value.
 - Department/Center Requested For**; e.g., Pathology, Radiology, Biology.
 - When you start typing the name, the choices will display.
 - Tracking Cost Center**: enter the numerical value
 - Will be used in reporting and should match the department/center entered
 - Admin Group**; e.g., **Device Support-Academic-DHTS** for most of the School of Medicine.
 - Allowed Storage Providers** – select **AWS-DHTS-PROD-1-V2** for AWS S3 buckets.
4. Select **Create**.

To create a Bucket:

1. Select the [hyperlink](#) of the **Project** name.



2. On the right side of the screen, under **New Bucket** for the Project, enter:
 - a. **Name** of the bucket
 - The “dh” indicates that this bucket is an AWS S3 bucket.
 - Bucket name should include dh-**ENTITY-DEPARTMENT-LAB** or EXPERIMENT, etc.;Examples:
dh-duhs-cardiology-mriscans2000to2020 or
dh-dusom-biochem-brennanlab or
dh-dusom-cagpm-Pr00123456

When choosing a name for your bucket, keep in mind:

- Bucket names must be unique.
- Namespace and bucket names should be DNS compatible since they can appear in a DNS record. For instance, **no underscores and no capital letters**. Use **lower case letters** a-z, numbers 0-9, and **hyphens**.
- Do NOT use personal or confidential information as names; e.g., NO PHI.
- Once a bucket name has been created, it cannot be changed.

How Do I Request/Provide Access to an Existing AWS S3 Bucket?

Access to an AWS S3 bucket must be authorized by the data owner and requested of the bucket owner. This should be a process that is determined by a Department, Entity or lab.

To get access to a bucket, you should provide the data owner/bucket owner your name and NETID, and the name of the bucket to which you need access, as well as if you will need Read/Write or Read-Only access.

To provide access to a bucket, the bucket owner will:

1. Go to **Portunus** <https://portunus-eso.duhs.duke.edu/portunus/>
2. Select **Projects** in the menu at the top.



3. Select the hyperlink of the desired [Project Name](#).

Duke Portunus

Projects Providers Buckets Keys Reports

AWSTest5_5_21

PI/Owner tlh52@duke.edu
Fund Code 201070173
Status active

AWS-DHTS-PROD-1-V2

Name	Owner	URL	Handle	Oper
dh-dusom-dmpi-test	tlh52@duke.edu	https://dh-dusom-dmpi-test.s3.amazonaws.com/	arn:aws:s3:::dh-dusom-dmpi-test	
dh-dusom-oasisshowntell	tlh52@duke.edu	https://dh-dusom-oasisshowntell.s3.amazonaws.com/	arn:aws:s3:::dh-dusom-oasisshowntell	

4. On the right hand side of the screen, under **Authorize user for dh-.....**, enter the [NETID@duke.edu](#) for the **Local User** and the desired **Privilege Level** (read-write or read-only) and select **Create+**.

Duke Portunus You are logged in as tlh52@duke.edu [sign out](#)

Projects Providers **Buckets** Keys Reports

dh-dusom-dmpi-test (Project: AWSTest5_5_21)

Provider AWS-DHTS-PROD-1-V2
Bucket Name dh-dusom-dmpi-test
Owner tlh52@duke.edu
Versioning enabled
Retention Period 90 days
Access URL https://dh-dusom-dmpi-test.s3.amazonaws.com/
AWS-DHTS-PROD-1-V2 Handle arn:aws:s3:::dh-dusom-dmpi-test
Fund Code 201070173

Authorize user for dh-dusom-dmpi-test

Local User

Privilege Level
read-write (without delete) ▼

Create +

Granted Permissions for dh-dusom-dmpi-test

User	Permission	Delete
tlh52@duke.edu	owner	
lk50@duke.edu	read-write	

5. The name will display in the list of users “Granted Permissions for dh-.....” of the bucket.
6. As the bucket OWNER, if you have not already done so, select [Keys](#) in the upper menu; select [Create Keys](#).

Duke Portunus

Projects Providers Buckets **Keys** Reports

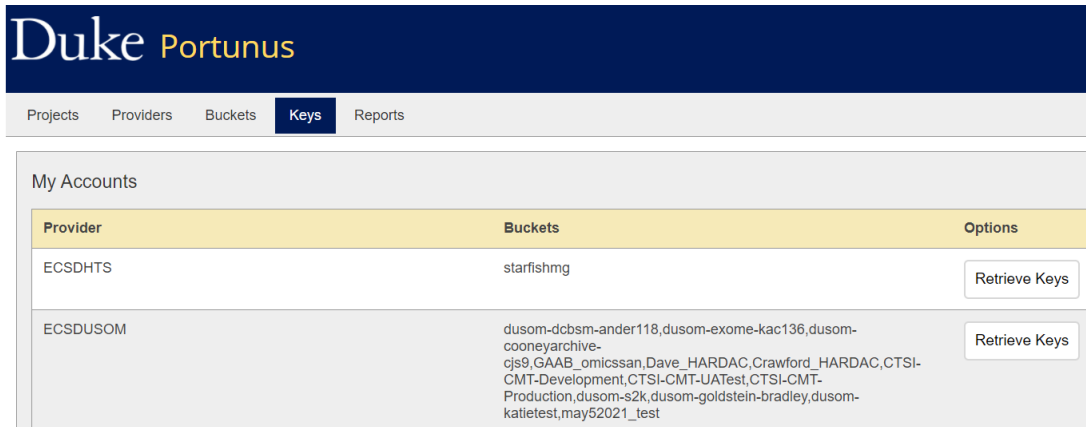
7. You will need to notify the users of the bucket when you have added them as a user.
8. Users of the bucket will be able to use [Portunus](#) to see the name(s) of their bucket(s) and to access the [Keys](#) or S3 credentials that are required for using the S3-compatible tools/clients to use the bucket(s).

How do I Access my S3 Keys or Credentials for a Bucket?

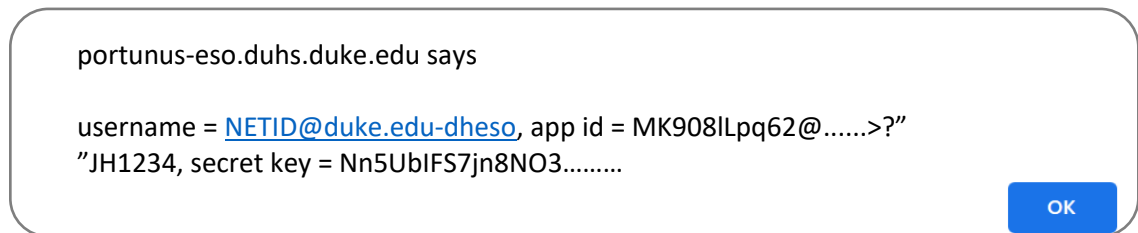
1. Go to **Portunus** <https://portunus-eso.duhs.duke.edu/portunus/>



2. Select **Keys** in the menu at the top.
3. Under **Options**, select **Retrieve Keys** for the desired **Project**. *NOTE: All buckets that are created in a Project share the same S3 credentials (keys) for each unique user.*



4. After selecting, **Retrieve Keys**, a pop-up box will display at the top of your screen that will say:



5. For AWS S3 buckets:
User ID/Access ID = **app id**
Secret Key = **secret key**

How do I Access my Bucket and Move/Copy Data?

The AWS S3 buckets use the same type of S3 protocol that the ECS on-site S3 buckets use. An S3-compatible client or command line interface (CLI) or REST APIs are required to be able to read and display what would otherwise be unintelligible unicode. There are several types of S3-compatible clients and CLIs that are available and easy to use. Below is a table of just a few options.

Administrative access on your computer is required to be able to download the clients or CLIs. If you do not have administrative access, contact your local IT support personnel or the DHTS Service Desk at 919-684-2243 or via the DukeHealth ServiceNow portal: <https://duke.service-now.com/sp?id=index>

NOTE: if working remote, you MUST be on the Duke Health VPN to access and use the S3 clients or CLI and connect to the S3 buckets.

	ExpanDrive Client	S3 Browser Client	CyberDuck Client	rClone (CLI)	AWS CLI	Globus
Windows OS	✓	✓	✓	✓	✓	✓
Mac OS	✓	NO	✓	✓	✓	✓
Linux OS	✓	NO	NO	✓	✓	✓
Cost	Free from OIT Software Library	<u>Free version:</u> limited to 2 accounts <u>Pro Version:</u> \$29.95 plus tax for lifetime license; faster	Free with ads; Remove ads w/\$10.00 donation.	Free	Free	Free; provided by DHTS
Targeted User	Novice; users who want “Explorer” or “Finder” like experience	Easy; Windows OS users who want easy to use, quick interface.	Easy; great alternative to ExpanDrive for Macs.	Power user; requires experience w/CLIs	Power user; requires experience w/CLIs	System admin; researcher working w/large datasets
Features		Allows buckets to be grouped and seen all at once in an Account (with same keys).	Allows buckets to be grouped and seen together in a Collection (with same keys). Can be used with Box.	Does not use computer’s cache to move files.	Some believe it is easier to learn than rclone. Uses threading to copy multiple files simultaneously.	Best to use in transferring/ copying large files
Speed of Use	Acceptable for small data moves (<1GB)	Faster than ExpanDrive	Faster than ExpanDrive	Faster than Clients	Faster than clients.	Faster for large files (1TB+)
Limitations	Have to “mount” each individual bucket. Uses computer’s cache.	Not available for Mac or Linux. Uses computer’s cache.	Not available for Linux. Uses computer’s cache to move files.	Harder to learn.	Harder to learn.	Currently only w/non-PHI data and on-premise S3 buckets
How to Access	https://software.duke.edu/node/635	S3 Browser Client Download	https://cyberduck.io/	https://rclone.org/downloads/	https://aws.amazon.com/cli/	Globus Login
How to Use	Mount bucket like a drive in Finder or Explorer	Create Account		Common OS shell commands in the CLI	Common OS shell commands in the CLI	Globus install & account setup. Globus GUI.

In addition, here is a link to directions on how to access the S3 bucket from Python or R. “Accessing S3 from the CLI, Python or R”: <https://sciwiki.fredhutch.org/compdemos/aws/>

Considerations for Copying, Moving and Accessing Data in AWS S3 Storage

The AWS S3 object storage uses a method called **Intelligent Tiering (IT)**. Intelligent Tiering in AWS automatically moves the data object down to less expensive storage mediums based on the last day that an object/file was accessed as follows:

1. Data Starts in AWS S3 **Frequent Access** Tier. If not accessed for 30+ days,
2. Moves to AWS S3 **Infrequent Access**. If not accessed for 60+ days,
3. Moves to AWS S3 **Archive Instant Access** (also called Glacier Instant Retrieval for data accessed 1x/quarter). If not accessed for 90+ days,
4. Moves to AWS **Glacier Flexible Retrieval**. If not accessed after 180+ days,
5. Moves to AWS **Deep Glacier**.

Prior to copying or moving any data to the cloud, there are several items that you should consider:

- A. Appropriateness of Data Storage in the Cloud
 - a. Security Requirements; e.g., HIPAA, FISMA, FERPA, etc.
 - b. Data Use Agreements
 - c. IRBs
- B. Volume and Average Size of the Data
- C. Overall Data Pipeline and Lifecycle
 - a. Why will data need to be accessed in the future?
 - b. Who will need to access the data – and to what data will they need access?
 - c. Level of familiarity with accessing and using tools for data management?
 - d. How will data be copied, moved, and accessed?
 - e. How frequently will the data be accessed?
 - f. Who will be responsible for how and for how long the data will be retained – and then deleted, if and when appropriate?
- D. Desired Organization of the Data
 - a. Naming Conventions, data labeling standards, etc.
- E. Need for Data Redundancy

NOTE: when considering copying or moving data to the AWS S3 storage, any data file (object) less than or equal to 128 KB will ALWAYS remain in the AWS S3 Frequent Access Tier – the highest cost tier for AWS S3 storage. In order to reduce costs, consider compressing and zipping related files or creating a tar.gz (tarball) in Linux.

Using ExpanDrive (Windows, MAC OS, Linux)

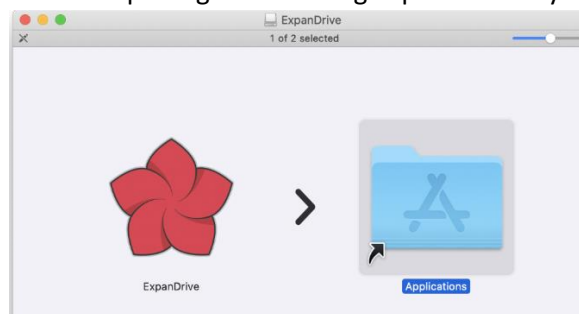
ExpanDrive is a Mac/Linux/Windows client that maps S3 storage to a drive letter in Explorer and can be used with Macs in Finder. ExpanDrive can be used to copy, move or download datasets <2-3 GB in size. (Larger files should leverage a NATIVE S3 command line such as rClone).

ExpanDrive can be run on:

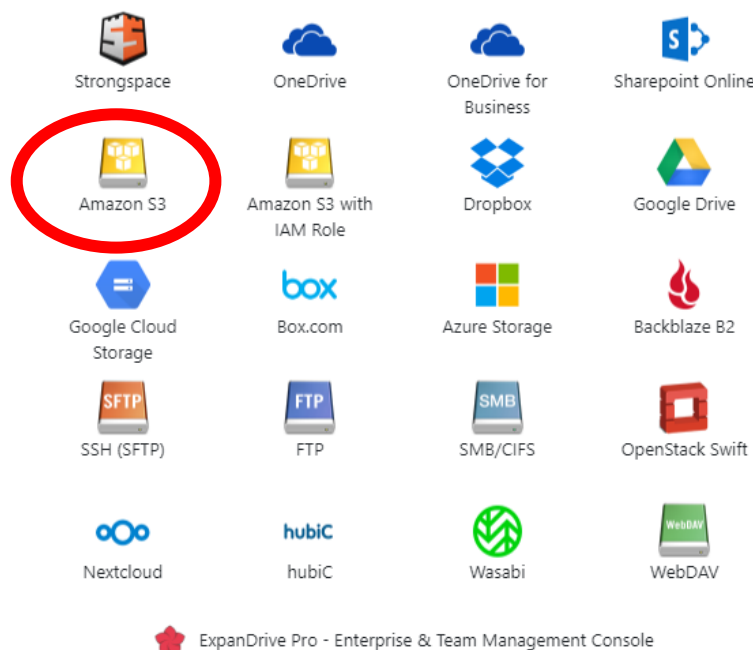
- Windows 7 – Windows 10
- macOS x 10.10 or newer
- Linux – all modern 64 bit desktop Linux environments

To Download and Setup ExpanDrive to Connect to the S3 Object Storage


1. Go to the Duke OIT Software Library to <https://software.duke.edu/node/635>, search on ExpanDrive, add it to your Cart, and checkout.
2. On a Mac, click on the .dmg installation package – then drag ExpanDrive to your Applications folder to install.



3. On a Mac, select the zip file to decompress ExpanDrive and then double click the ExpanDrive to run it.
4. From the **ExpanDrive** window, click ADD.
5. To Select Drive Type, click on the yellow icon for **Amazon S3**.



6. In the data entry screen, enter the following (as an example):



Server:

Access Key:

Secret Key: ☐ Use MFA

Custom Region:

Nickname:

Bucket:




Drive Letter:

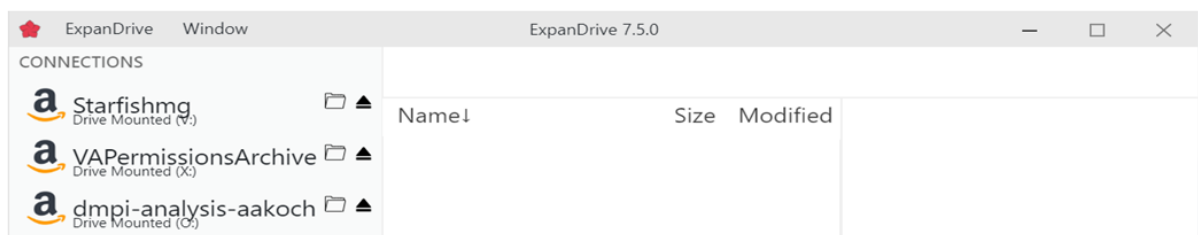
☒ Automatically Mount as Drive
☐ Read Only

- a. **Server:** s3.amazonaws.com
- b. **Access Key ID:** the [app id](#) under [Keys](#) in [Portunus](#)
- c. **Secret Key:** provided under [Keys](#) in [Portunus](#)
- d. **Nickname:** what you want to name your S3 bucket for your use (usually same name as Project)
- e. **Bucket:** enter the BUCKET name; e.g., *dh-dusom-dmpi-test*
- f. **Drive Letter (for Windows):** select one not currently in use on the computer to which you are mapping the bucket
- g. **Save**

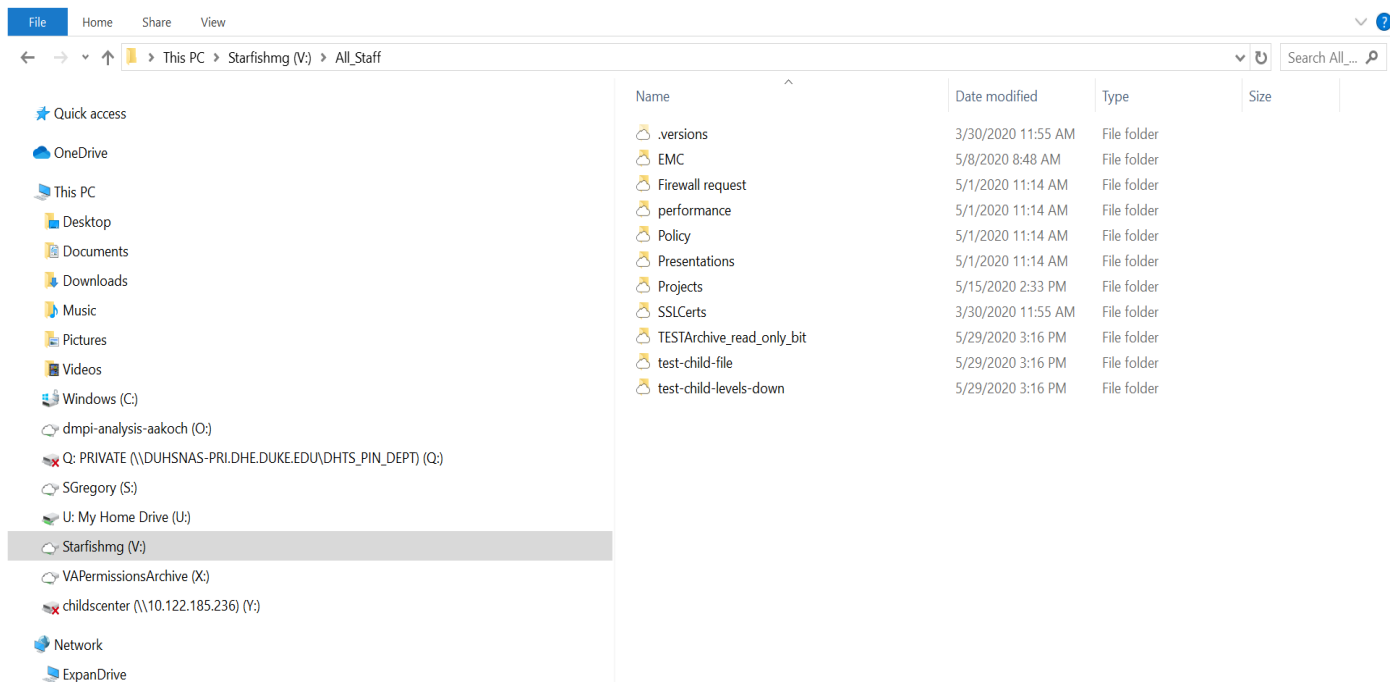
Once you have the ExpanDrive client installed, you can “pin it” to your taskbar for easy access.

To See Data Files in the S3 Bucket

1. Open ExpanDrive on your computer.
2. Select the  to eject the drive and then  once again to refresh/remount the drive. (Refreshes the drive/view.)
3. Select the  to reveal the contents in the drive.



4. You will see the directories/drives mapped to your computer and the contents of the S3 bucket.
5. From here you can search files, drag and drop to move files, etc. to/from the S3.
 - Object versioning is enabled at the bucket level.
 - Versions can be viewed by right clicking the red icon previous version.
 - Folders can be created inside of a bucket by right clicking on the bucket drive, select “New”, Folder.



To connect/map to other S3 buckets using ExpanDrive

Repeat the steps above.

To Copy Files to the S3 Object Storage Using ExpanDrive

You can copy files that are saved to your computer to the S3 object storage bucket just like you would to any other drive/folder such as drag and drop, Send to; Copy to. (See below for other client connectivity options like rclone.)

Open Explorer to view bucket contents mapped to the drive. *You will see clouds on files and folder you create.*

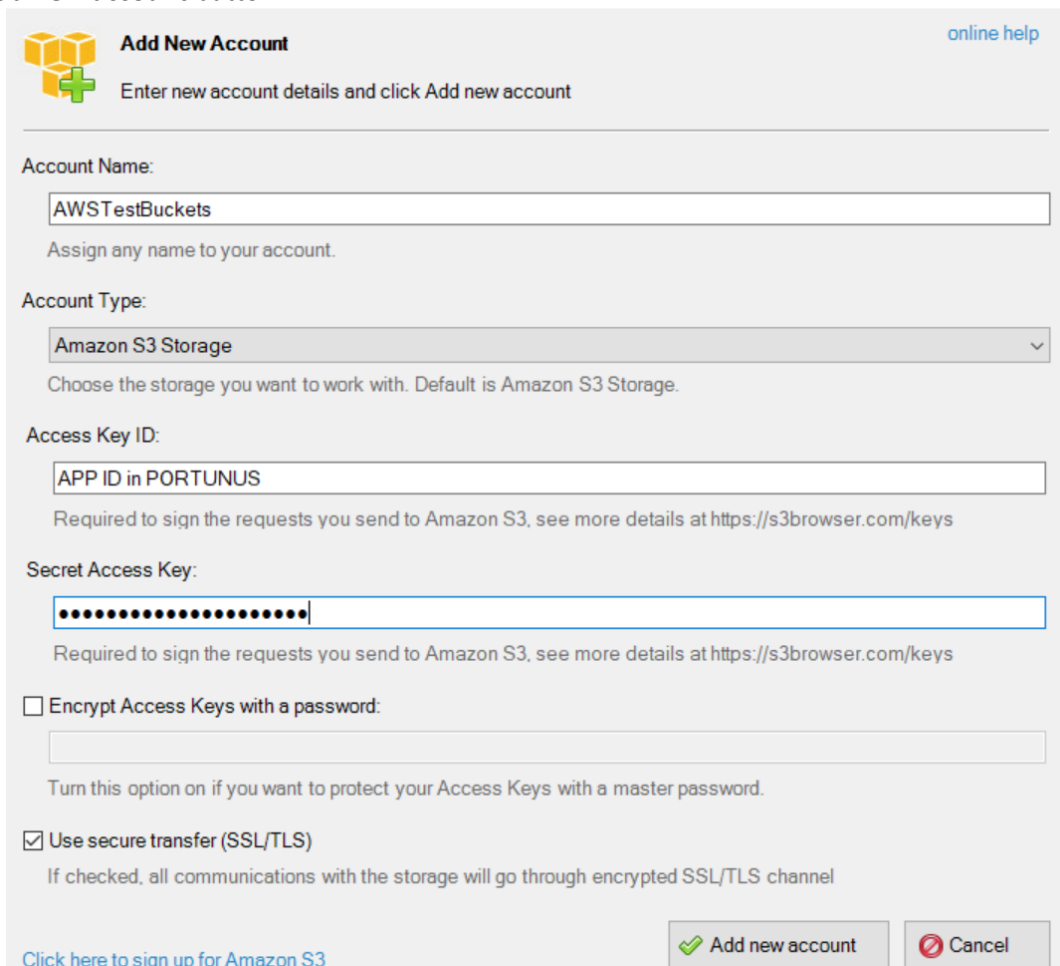
Using an S3 Browser Windows Client

S3 Browser is a Windows client which can be used to push and pull data on S3 compatible storage such as ECS. You can download and install S3 Browser from <https://s3browser.com/>. In order to setup and use S3 Browser with appropriate performance and with more than 2 accounts, you will need to pay for the Pro version. Highly recommended!

To access Object Storage via S3 Browser you need to create an Account that will supply the credentials to the storage provider. Then you create the link to the bucket you want to access.

To Create an S3 Browser Account and Associated Buckets

1. Open S3 Browser, in the top menu, click **Accounts, Add new account**
 - a. Enter **Account Name:** Enter name of account; best practice is to be same as [Project name](#) in Portunus.
 - b. **Account Type:** [Amazon S3 Storage](#)
 - c. **Access Key ID:** use the [AppID](#) provided (clicking on [Retrieve Keys & Keys](#) tab in [Portunus](#))
 - d. **Secret Key:** use the info provided on the [Retrieve Keys](#) tab in Portunus
2. Click **Add new account** button



Add New Account [online help](#)

Enter new account details and click Add new account

Account Name:
AWSTestBuckets
Assign any name to your account.

Account Type:
Amazon S3 Storage
Choose the storage you want to work with. Default is Amazon S3 Storage.

Access Key ID:
APP ID in PORTUNUS
Required to sign the requests you send to Amazon S3, see more details at <https://s3browser.com/keys>

Secret Access Key:
.....
Required to sign the requests you send to Amazon S3, see more details at <https://s3browser.com/keys>

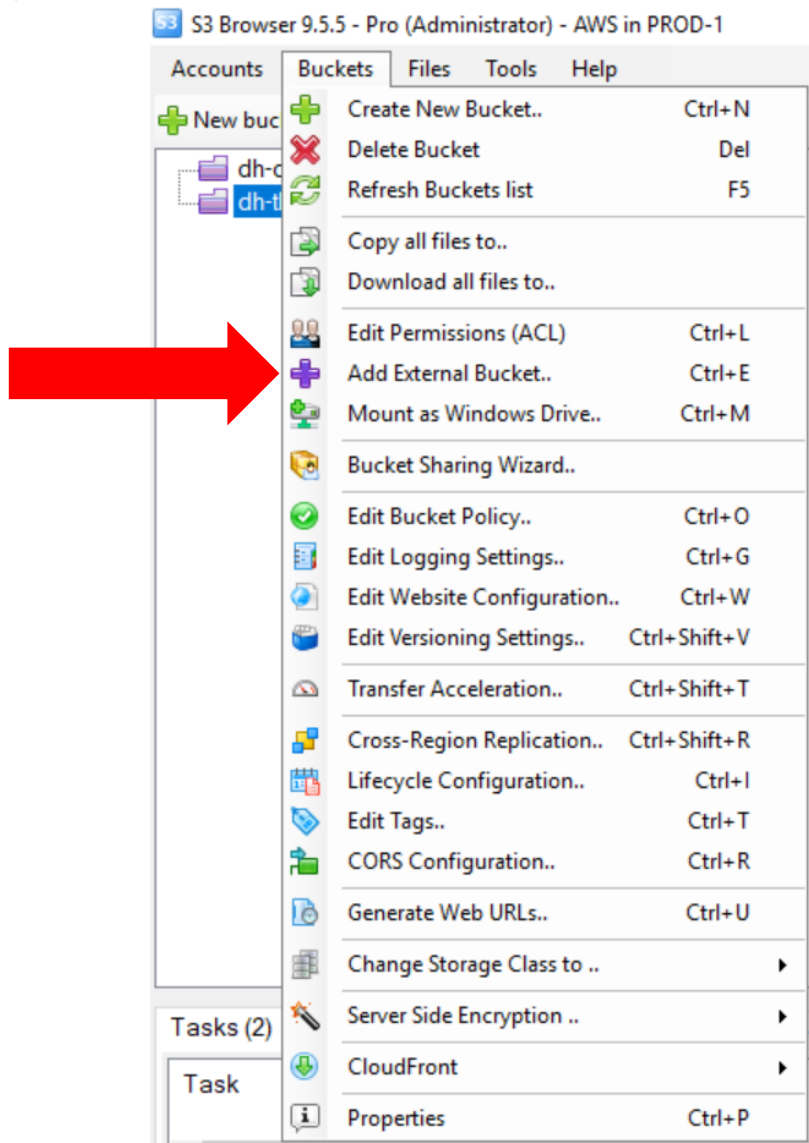
☐ Encrypt Access Keys with a password:
Turn this option on if you want to protect your Access Keys with a master password.

☒ Use secure transfer (SSL/TLS)
If checked, all communications with the storage will go through encrypted SSL/TLS channel

[Click here to sign up for Amazon S3](#)

3. When asked to add an [External Bucket](#), select yes.
4. Enter the AWS S3 [external bucket name](#); e.g., dh-thwest123.
5. To add more AWS S3 buckets, go to [Buckets](#) in the main menu.

6. Click [Buckets, Add External Bucket](#).

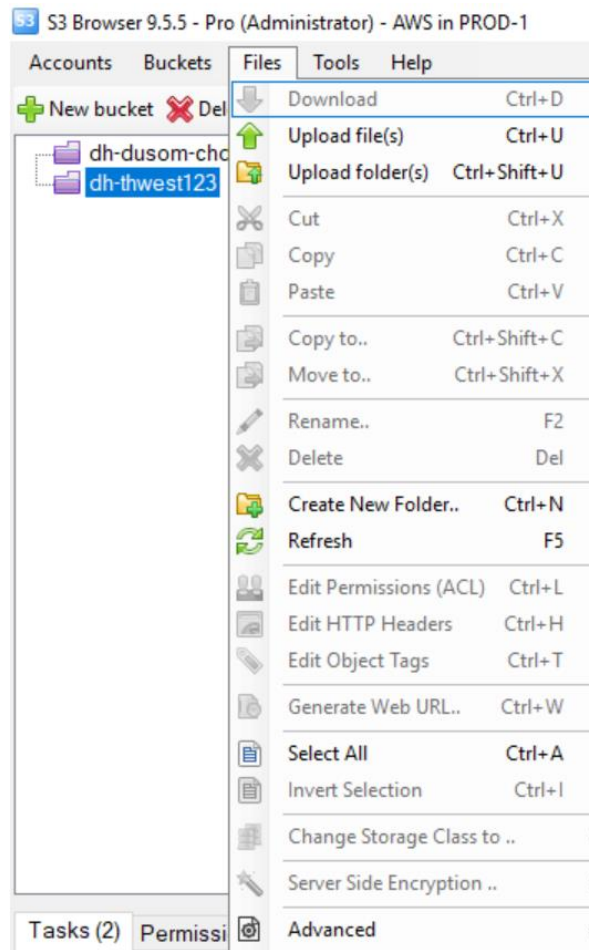


7. Use the bucket Name provided on the [Buckets](#) tab in [Portunus](#).

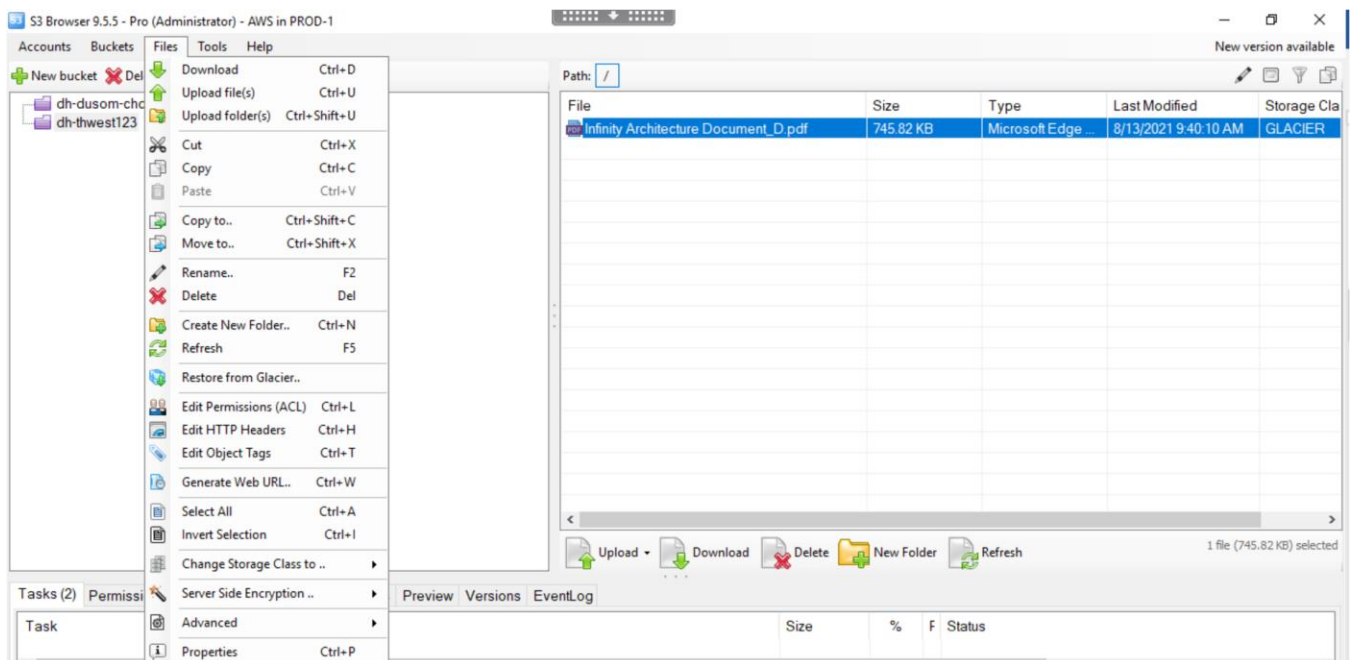
Do NOT create new buckets via the S3 Browser!!!

To Organize, Move or Copy Data to/from the AWS Bucket via the S3 Browser

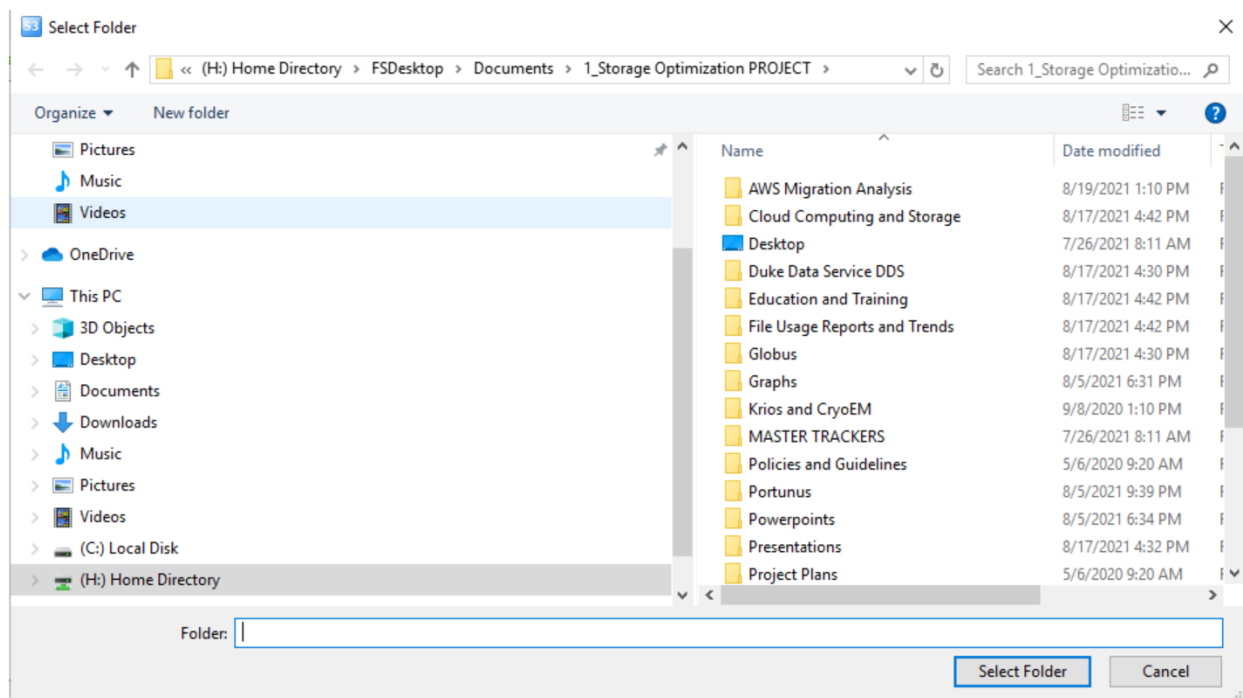
1. Select the desired [Bucket](#) and in the top menu, select [Files](#).
2. From here you can:
 - [Create a New Folder](#)
 - Upload file(s) from Explorer, Finder, another mapped device



3. To **download a file** from the AWS S3 bucket, select the desired file in the S3 Browser; select **Files** in the menu and **Download**.



4. Select the location and desired **Folder** to which to download the file.



Additional Information on the S3 Browser can be found here: <https://www.nakivo.com/blog/overview-of-amazon-s3-browser-windows/>

How to Use Cyberduck



Cyberduck is a popular free and open source FTP client for Mac and Windows since 2003. Cyberduck takes a unique and simple approach to file transfers.

[Cyberduck's feature list](#) includes:

- Secure file transfers with SFTP and FTP-SSL.
- Easy file management with cut and paste as well as drag and drop.
- File previews so you can view the files before you download them.
- Password-less authentication with [ssh keys](#).

How to Install Cyberduck on your Mac

1. Browse to the [Cyberduck website](#).
2. Click the **Download Cyberduck for Mac** button to obtain the current version of the installer.
3. After the download is complete and the zip file finishes extracting, drag the Cyberduck application to your **Applications** folder. (NOTE: you may need to allow your system access to the application by changing your security settings for your Mac.)
4. Double-click the Cyberduck icon to start the program.

How to Install Cyberduck on your Windows PC

1. Browse to the [Cyberduck website](#).
2. Click the **Download Cyberduck for Windows** button to obtain the current version of the installer.
3. After the download is complete, double-click the downloaded executable file and follow the prompts (the defaults will suit just about everyone).
4. Select the Cyberduck program from your Start menu to start the program.

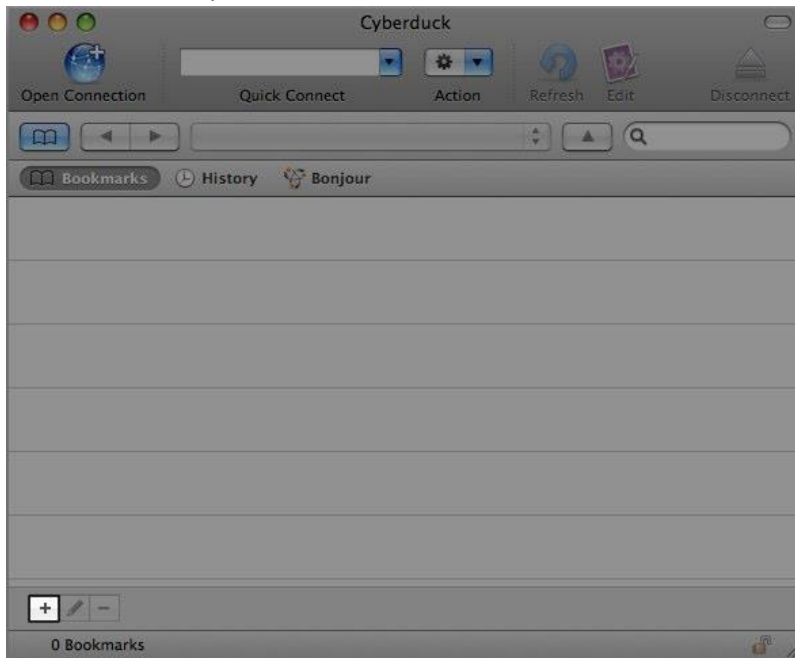
How to Connect and Transfer Files with Cyberduck

YOU MUST BE ON THE DHTS VPN IF YOU ARE REMOTE.

Once Cyberduck is installed, you can create a bookmark to your AWS Bucket.

NOTE: after installing Cyberduck, you may need to make modifications to ALLOW it to be accessed via your computer. If so, your operating system will ask you to Update Security Preferences – to unlock them and allow the Cyberduck application.

1. Click on the "+" symbol at the bottom left corner to create a bookmark.



2. Click the top bar and select "**Amazon S3**" as the connection type

s3.amazonaws.com - S3

Amazon S3

Nickname: s3.amazonaws.com - S3

URL: https://s3.amazonaws.com

Server: s3.amazonaws.com Port: 443

Access Key ID: Access Key ID

☐ Anonymous Login

SSH Private Key: None

Client Certificate: None

▼ More Options

Path:

Web URL: http:///

Download Folder: Downloads

Transfer Files: Open multiple connections

Timezone: UTC

Encoding: Default

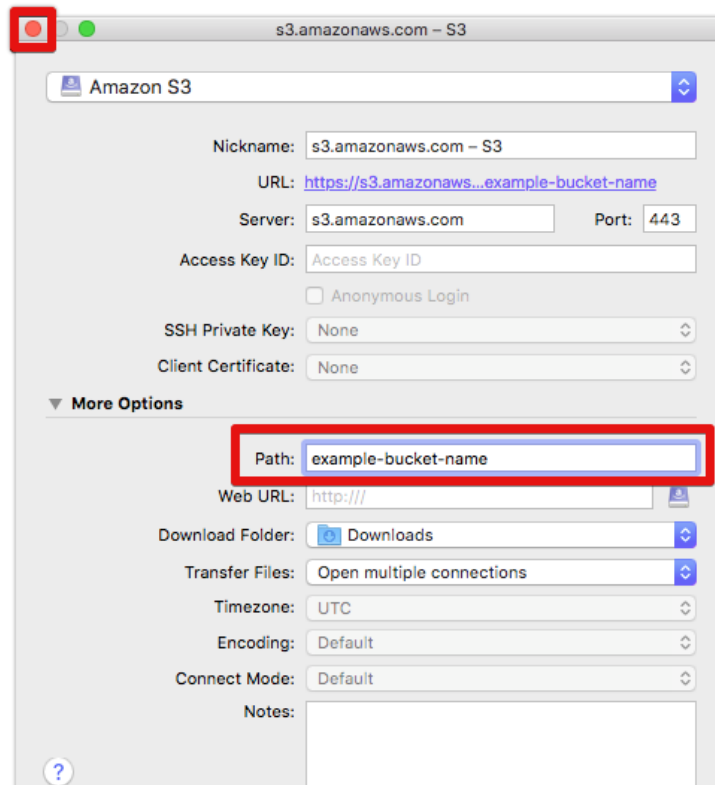
Connect Mode: Default

Notes:

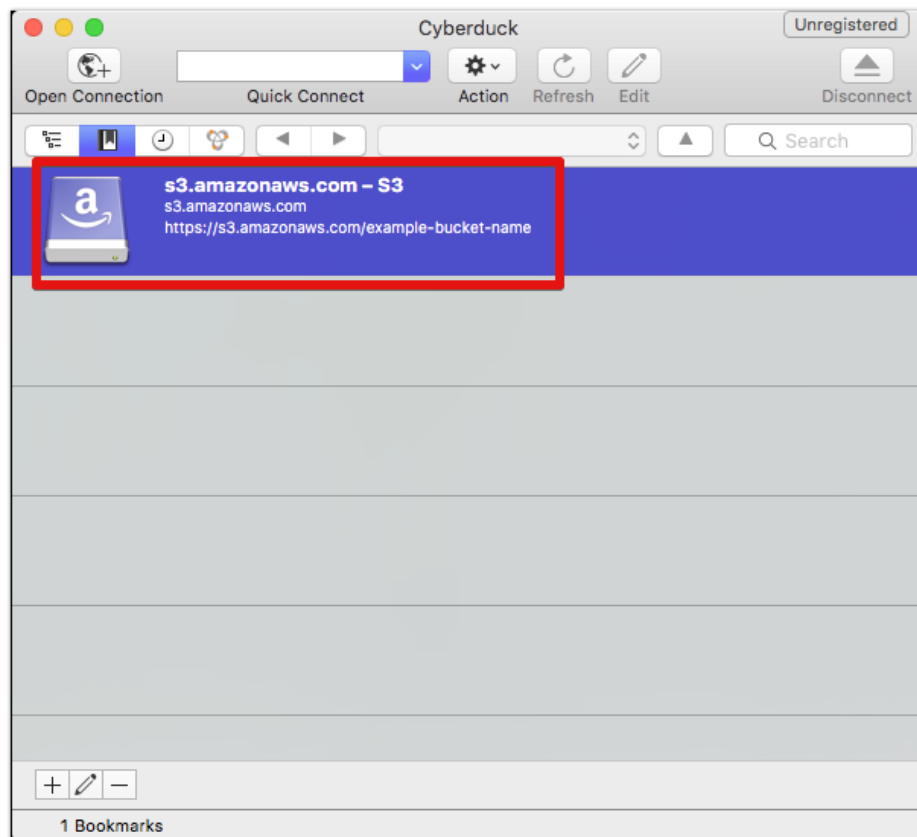
3. Server: should auto populate to s3.amazonaws.com.
4. Access Key ID: **APP ID** (provided to you via [Portunus](#))

*NOTE: This is not the User ID that is used for the ECS bucket.
This is the 2nd key that is in the key pop-up in Portunus.*

5. Secret Access Key: **Secret Access Key** (provided to you via [Portunus](#))
6. Path: **Bucket Name**; e.g., **dh-dusom-cellbio-twtest**
7. Select **Connect**.
8. Change "Path" to the **bucket name** (also in Portunus; e.g., dh-dusom-cellbio-twtest) and click the red "X" button in the top left corner to save the bookmark.

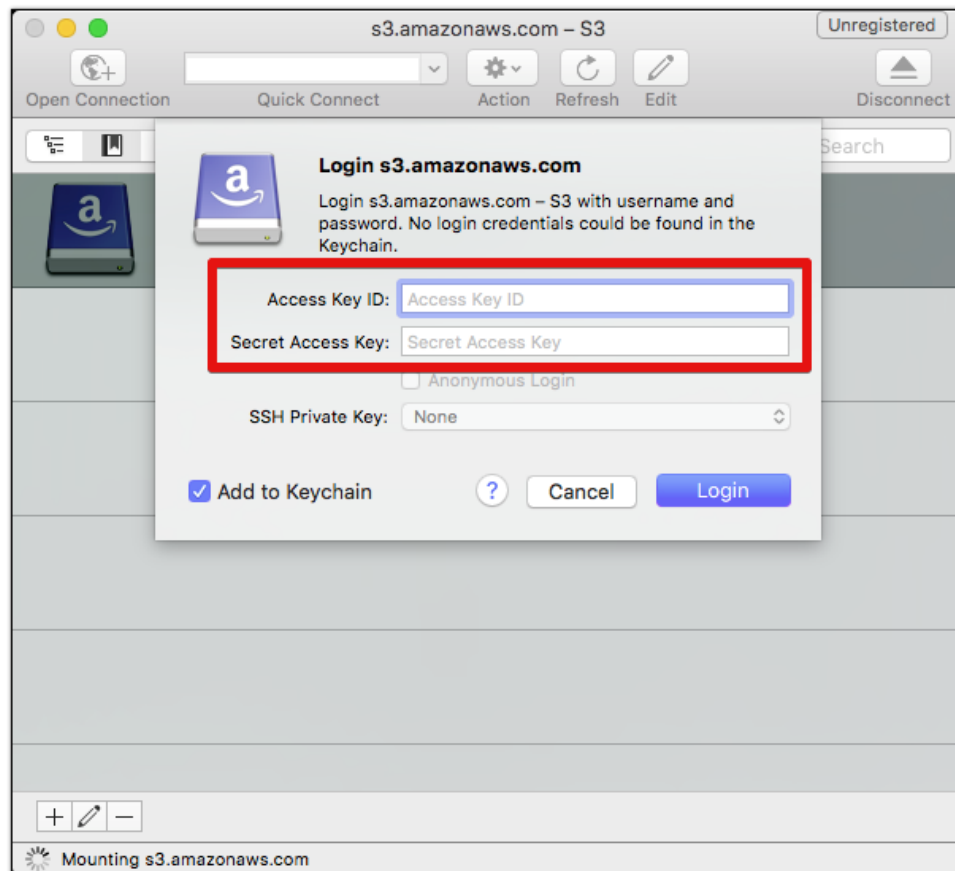


9. Double click on the bookmark to open a connection.



10. Input the Access Key ID and Secret Access Key then click "Login." (your credentials are the ones in Portunus)

- Access Key ID: [APP ID](#) (provided to you via [Portunus](#))
- Secret Access Key: [Secret Access Key](#) (provided to you via [Portunus](#))



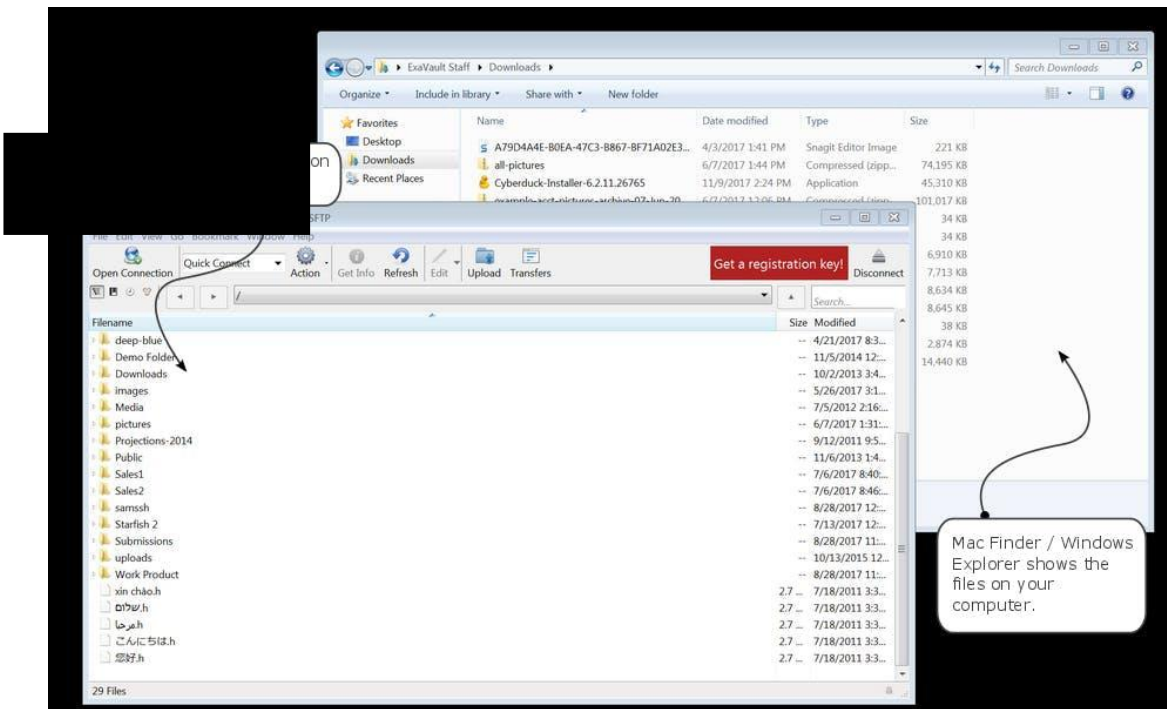
- Cyberduck Homepage - <https://cyberduck.io/>
- Cyberduck Help Documents - <https://trac.cyberduck.io/wiki/help/en>

You'll know you are connected when you see a list of files in the Cyberduck Browser window.

How to Transfer Files and Use the Browser Window

The Browser window is the main part of the application window in Cyberduck.

The folders and files that you see in the Browser window are like in a Finder (Mac) or Explorer (PC) window. Transferring files with Cyberduck can be done by dragging files from one window to another.

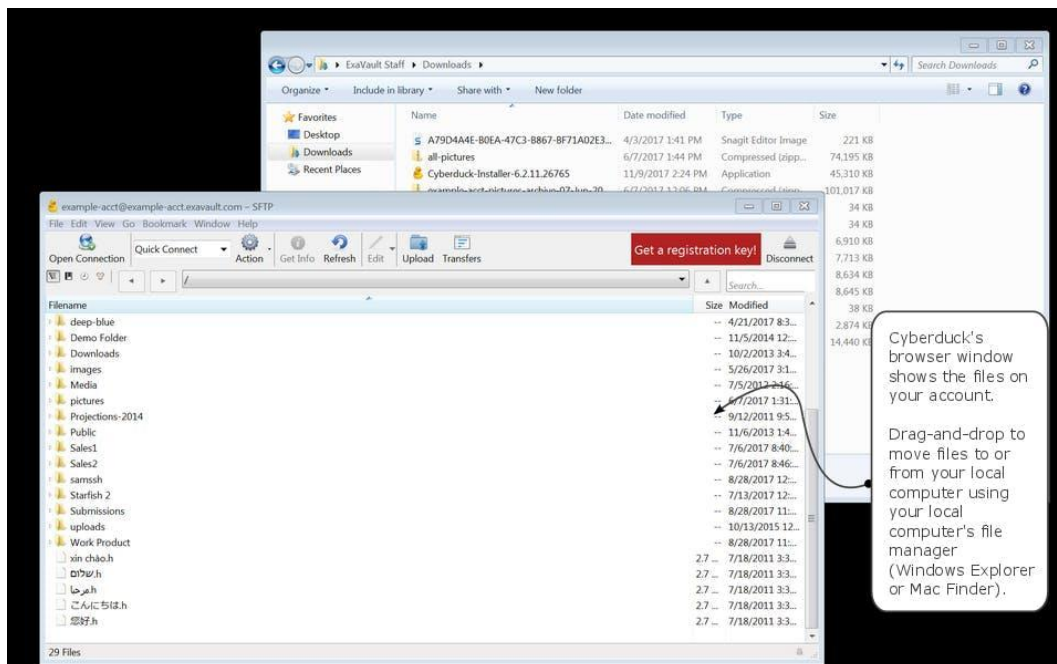


Mac:

- Finder will show the files on your computer.
- Cyberduck's Browser window shows the files in your S3 bucket.

Windows:

- Windows Explorer will show the files on your computer.
- Cyberduck's Browser window shows the files in your S3 bucket.



To Upload Files:

1. Select file(s) in Finder / Windows Explorer.
2. Drag the selected file(s) to the Browser window in Cyberduck.

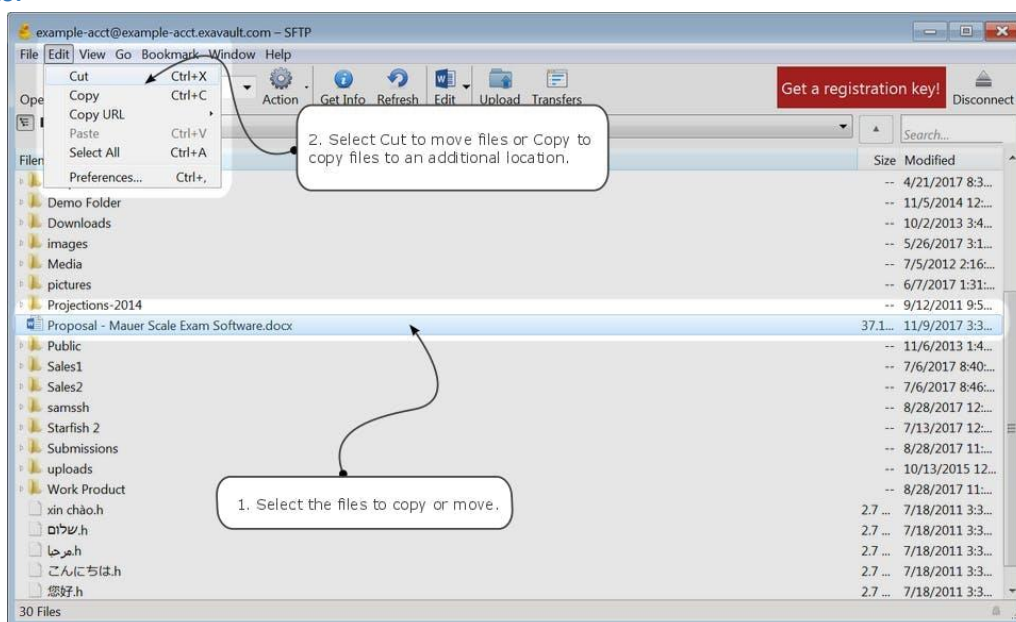
To Download Files:

1. Select file(s) in Cyberduck's Browser window.
2. Drag the selected file(s) to the Finder / Windows Explorer window.

To Create Folders in a Bucket:

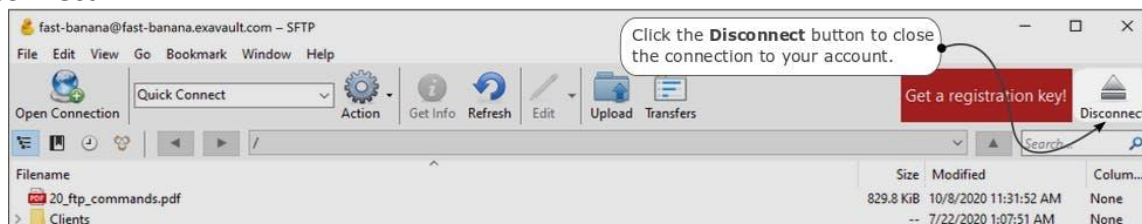
1. Click the **File** menu.
2. Select **New Folder**.
3. Enter the name of your new folder. *Avoid using spaces in folder names.*

To Move Files:



1. Select the file(s) to move.
2. Click the **Edit** menu and select **Cut**.
3. Double-click on the folder you want to move the files to.
4. Click the **Edit** menu and select **Paste**.

To Disconnect



Other S3 Client Connectivity Options

Applications that interact with object storage via the S3 RESTful API are the preferred client access method. Application developers can use the S3 software development kit (SDK) to develop snippets of code to read, write, copy, move, sync, etc. *This is considered a native S3 client and provides the best performance over other S3 browsers which require a file system and/or drive mapping to address buckets to move data.*

Native S3 Browsers:

- Rclone
- S3cmd
- AWS cli

Browsers listed above do not interact directly with applications and are used to move data in an optimized 1Chunk size, multipart upload, thread count, local disk caching and aging. Performance is ~**30%-40%** greater than *mounted* file system based S3 client browsers.

How to Install and Use rclone on Linux and Windows

See <https://rclone.org/docs/> for details

1. Download rclone from <https://rclone.org/downloads/>
2. To install rclone on **Linux/macOS/BSD** systems:
 - a. Run **curl https://rclone.org/install.sh | sudo bash**
 - b. Run **rclone config** to setup.
 - c. Should get confirmation message that "rclone vx.xx has successfully installed"
3. Run **rclone config** at the command prompt by typing: [root@localhost ~]# **rclone config**
2018/12/14 06:59:18 NOTICE: Config file "/root/.config/rclone/rclone.conf" not found - using defaults
4. To install rclone on a **Windows** machine:
 - a. Create a directory off the **c:\rclone**.
 - b. **Download** the appropriate Windows files such as for AMD64 bit.
 - c. **Extract** the files into c:\rclone.
 - d. **Go to Command Line to change directories: C:\Users\BrysonC>cd**
 - e. Type **C:\>cd rclone** to change to rclone.
 - f. Type **c:\rclone>dir** to ensure that the executable file is there.
 - g. Type **C:\rclone config** to configure rclone and then follow the prompts to complete the config.

Green is the category to configure; Red is the selection to enter in the steps below.

5. If No remotes found - make a new one. **n/s/q> n) New Remote**
6. Enter the **name>** of the new remote and enter **Project Name** from **Portunus**.
7. Type of storage to configure. **name> S3**
8. **Choose your S3 provider.**
Choose a number from below, or type in your own value
1 / Amazon Web Services (AWS) S3
 \ "AWS"
provider> 1
9. **Get AWS credentials** from runtime (environment variables or EC2/ECS metadata if no env vars).

env_auth> 1 / Enter AWS credentials in the next step
 \ "false"

10. Enter AWS Access Key ID. **access_key_id>** enter APP ID from **Portunus**

11. Enter AWS Secret Access Key (password) **secret_access_key>** Enter secret key from **Portunus**

12. Enter Region to connect to. **region> 1** / for us-east-1

13. Enter Endpoint for S3 API. Leave blank for AWS (s3.amazonaws.com). **endpoint>**

14. Enter Location constraint - must match the Region. **location_constraint> 1**

15. Enter the Canned ACL used when creating buckets and storing or copying objects. **acl> 1**/ Owner gets FULL_CONTROL. No one else has access rights (default).\ "private"

16. Server-side encryption algorithm: **server_side_encryption>1** for none

17. Storage Class: **storage_class> 8** for Intelligent Tiering

18. Edit advanced config? (y/n) **n**

19. Remote config y/e/d> **y** (if it is correct below) or **e** to Edit this remote if you need to correct it

```
-----  
[s3]  
type = s3  
provider = AWS  
env_auth = false  
access_key_id = app ID from Portunus  
secret_access_key = ZpPIF2sfH7ARs3kL9bBx63rq2y5PxBH (example from Portunus)  
region = us-east-1  
endpoint =  
location_constraint =  
acl = private  
server_side_encryption =  
storage_class = 8  
-----
```

19. Current remotes: **e/n/d/r/c/s/q> q**) Quit config

Name	Type
====	====
s3	s3

20. [root@localhost ~]#

Important options are:

- **--config=FILE** (default FILE=.rclone.conf)
 - Specifies the rclone configuration file to use. Only necessary if the desired config file is not called "~/.rclone.conf" (the default name given to the config file).
- **--transfers=N** (default N=4)
 - Number of file transfers to be run in parallel. Increasing this may increase the speed of a large transfer, as long as the network and remote storage system can handle it.
- **--drive-chunk-size=SIZE** (default SIZE=8192)
 - The chunk size for a transfer in kilobytes; must be a power of 2 and at least 256. Each chunk is buffered in memory prior to the transfer, so increasing this increases memory usage.
- **--drive-formats** (docx, pdf, txt, etc.)
 - Sets the format used when exporting files. For example, the option '--drive-formats pdf' will automatically convert the chosen file(s) to PDF format.

rClone Example Commands

This command string will copy contents from the folder **ISO** into an S3 bucket called **bk-rc84**.

```
rclone --v copy /home/rc84/ISO S3:bk-rc84
```

Another example defining additional options to increase throughput and bandwidth:

```
rclone copy --v --transfers=30 --checkers=16 --s3-upload-concurrency 20 --s3-chunk-size=25M --s3-disable-checksum "C:\Users\rc84\Downloads\ISO\ubuntu-18.04.1-desktop-amd64.iso" S3:"bk-rc84"
```

To mount the drive in **Windows**, leave the command window open. The drive will appear in Windows Explorer.

```
rclone mount --vv --vfs-cache-mode writes --daemon --no-checksum --no-modtime --vfs-cache-max-age 0h1m0s --transfers=30 --checkers=16 --s3-upload-concurrency 20 --s3-chunk-size=25M S3:bk-rc84 l:
```

NOTE: The recent version of rclone wants the ability to create buckets so, commands to copy may fail before starting the upload and present with an AccessDenied error message. If this happens, try this:

Provide the Flag :

```
--s3-no-check-bucket
```

If set, don't attempt to check the bucket exists or create it

the upload should work as intended.

For example, this command runs just fine:

```
"rclone copy .\2020-Stuff.7z mf229AWS:dh-dusom-bucketimages -P --s3-no-check-bucket --checksum"
```

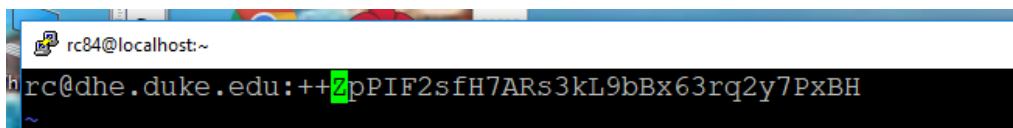
S3FS Mounted File System Based S3 Client (Linux & MAC OS only)

S3fs is a [Fuse](#) file system that allows you to mount an object storage S3 bucket as a local file system. It stores files natively and transparently in S3 (i.e., you can use other programs to access the same files).

Maximum object size that s3fs can handle depends on Amazon S3; e.g., up to 5 GB when using single PUT API; up to 5 TB is supported when Multipart Upload API is used.

To Install S3FS with Linux

1. Install the S3Fs package Linux build **yum install s3fs-fuse**.
2. Create a passwd-s3fs file to store the s3 ID and secret key in your home directory.
3. Create in your home directory vi .passwd-s3fs
4. On the top line of the file, place your credentials. *The ones in the example have been modified for viewing purposes.*
5. Save the file which is a hidden file in your home folder.



6. Once saved, change permissions on the file **Chmod 600 .passwd-s3fs**
7. Create an empty directory of your choosing to mount the S3 bucket. *Example* **mkdir s3fs-mnt**.
8. Create an additional directory for your local data caching **mkdir s3mnt-cache**
9. Use the following mount command *changing the bucket name and local path* to reflect yours.

```
s3fs bk-rc84 /home/rc84/s3mnt -o url=https://s3-ecs.dhe.duke.edu -o use_path_request_style -o
use_cache=/home/rc84/s3mnt_cache -o uid=1000 -o gid=1000 -o umask=022 allow_other
mp_umask=022
```

10. To verify the mount was successful, type **df -k**. In the example below, you see the last entries file system *s3fs* mounted to */home/rc84/s3mnt*.

```
[rc84@localhost ~]$ df -k
Filesystem      1K-blocks    Used   Available Use% Mounted on
/dev/sda3        39517336 6202264   33315072  16% /
devtmpfs         921600      0     921600    0% /dev
tmpfs            932640      0     932640    0% /dev/shm
tmpfs            932640    9820     922820    2% /run
tmpfs            932640      0     932640    0% /sys/fs/cgroup
/dev/sda1        303780 126056   177724   42% /boot
tmpfs            186532      0     186532    0% /run/user/1000
tmpfs            186532      0     186532    0% /run/user/0
s3fs             274877906944 0 274877906944  0% /home/rc84/s3mnt
[rc84@localhost ~]$
```

11. To unmount the file system, type **fusermount -u /home/rc84/s3mnt**

S3FS Options

1. **cipher_suites=AESGCM** is only relevant when using an HTTPS endpoint. By default, secure connections to IBM COS use the AES256-SHA cipher suite. Using an AESGCM suite instead greatly reduces the CPU overhead on your client machine, caused by the TLS crypto functions, while offering the same level of cryptographic security.
2. **kernel_cache** enables the kernel buffer cache on your s3fs mount point. This means that objects will only be read once by s3fs, as repetitive reading of the same file can be served from the kernel's buffer cache. The kernel buffer cache will only use free memory which is not in use by other processes. This option is not recommend if you expect bucket objects to be overwritten from another process/machine while the bucket is mounted, and your use-case requires live accessing the most up-to-date content.
3. **max_background=1000** improves s3fs concurrent file reading performance. By default, FUSE supports file read requests of up to 128 KB. If larger, the kernel will split the request into smaller sub-requests and lets s3fs process them asynchronously. The max_background option sets the global maximum number of concurrent asynchronous requests. Default is 12, but setting it to an arbitrary high value (1000) prevents read requests from being blocked, even when reading a large number of files simultaneously.
4. **max_stat_cache_size=100000** reduces the number of redundant HTTP HEAD requests sent by s3fs and reduces the time it takes to list a directory or retrieve file attributes. Typical file system usage makes frequent access to a file's metadata via a stat() call which maps to HEAD request on the object storage system. s3fs default caches attributes (metadata) up to 1000 objects. Each cached entry takes up to 0.5 KB of memory. Ideally, you want the cache to be able to hold the metadata for all of the objects in your bucket. However, you may want to consider the memory usage of this caching. Setting it to 100000 takes no more than $0.5 \text{ KB} * 100000 = 50 \text{ MB}$.
5. **multipart_size=52** sets the maximum size of requests and responses sent and received from the COS server, in MB scale. s3fs default is 10 MB. Increasing this value increases the throughput (MB/s) per HTTP connection; however, the latency for the first byte served from the file will increase respectively. Therefore, if your use-case only reads a small amount of data from each file, you probably do not want to increase this value. Furthermore, for large objects (say, over 50 MB) throughput increases if this value is small enough to allow the file to be fetched concurrently using multiple requests. COS best practices suggest requests that are multiples of 4 MB, and thus **recommends 52 (MB)**.
6. **parallel_count=30** sets the maximum number of requests sent concurrently to COS, per single file read/write operation. Default is 5. For very large objects, you can get more throughput by increasing this value. As with the previous option, keep this value low if you only read a small amount of data of each file.
7. **multireq_max=30** When listing a directory, an object metadata request (HEAD) is sent per each object in the listing (unless the metadata is found in cache). This option limits the number of concurrent such requests sent to COS, for a single directory listing operation. Default is 20. This value must be greater or equal to the parallel_count option above.
8. **dbglevel=warn** sets the debug level to warn instead of the default (crit) for logging messages to /var/log/syslog.

How to Install S3FS on MAC OS using Homebrew

1. Install Homebrew - <http://brew.sh/>

```
$ /bin/bash -c "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
```

2. Use Homebrew to install s3fs + dependencies via

```
$ brew install s3fs
```

<https://github.com/s3fs-fuse/s3fs-fuse/wiki>

```
brew install homebrew/fuse/s3fs
```

3. Create the file ~/.passwd-s3fs (one line file [accessKey:secretKey])

```
echo $ACCESS_KEY:$SECRET_KEY > ~/.passwd-s3fs
```

4. Create the directory where the S3 bucket will be mounted.

```
mkdir -p ~/home/user/s3fsmnt
chmod 600 ~/.passwd-s3fs
```

If running OS X 10.11.5 you may need to run the following command `brew cask install osxfuse`

5. Mount the bucket using same example for Linux install above.

Modified time

Modified time is stored as metadata on the object as **X-Amz-Meta-Mtime** as floating point since the epoch accurate to 1 ns.

If modification time needs to be updated, rclone will attempt to perform a server side copy to update the modification if the object can be copied in a single part. If the object is >5Gb or is in Glacier or Glacier Deep Archive storage, the object will be uploaded rather than copied.

Reading this from the object takes an additional **HEAD** request; metadata isn't returned in object listings.

Reducing Costs

Avoiding HEAD requests to read the modification time

By default rclone will use the modification time of objects stored in S3 for syncing. This is stored in object metadata which unfortunately takes an extra HEAD request to read which can be expensive (in time and money).

The modification time is used by default for all operations that require checking the time a file was last updated. It allows rclone to treat the remote more like a true filesystem, but it is inefficient on S3 because it requires an extra API call to retrieve the metadata.

Extra API calls can be avoided when syncing (using **rclone sync** or **rclone copy**) in a few ways, each with its own tradeoffs.

- **--size-only**
 - Only checks the size of files.
 - Uses no extra transactions.
 - If the file doesn't change size then rclone won't detect it has changed.
 - **rclone sync --size-only /path/to/source s3:bucket**
- **--checksum**
 - Checks the size and MD5 checksum of files.
 - Uses no extra transactions.
 - The most accurate detection of changes possible.

- Will cause the source to read an MD5 checksum which, if it is a local disk, will cause lots of disk activity.
 - If source and destination are both S3 this is the **recommended** flag to use for maximum efficiency.
 - `rclone sync --checksum /path/to/source s3:bucket`
- `--update --use-server-modtime`
 - Uses no extra transactions.
 - Modification time becomes the time the object was uploaded.
 - For many operations, this is sufficient to determine if it needs uploading.
 - Using `--update` along with `--use-server-modtime`, avoids the extra API call and uploads files whose local modification time is newer than the time it was last uploaded.
 - Files created with timestamps in the past will be missed by the sync.
 - `rclone sync --update --use-server-modtime /path/to/source s3:bucket`

These flags can and should be used in combination with `--fast-list` - see below.

If using `rclone mount` or any command using the VFS (eg `rclone serve`) commands, consider using the VFS flag `--no-modtime` which will stop rclone reading the modification time for every object. You could also use `--use-server-modtime` if you are happy with the modification times of the objects being the time of upload.

Avoiding GET requests to read directory listings

Rclone's default directory traversal is to process each directory individually. This takes one API call per directory. Using the `--fast-list` flag will read all info about the objects into memory first using a smaller number of API calls (one per 1000 objects). See [rclone docs](#) for more details.

```
rclone sync --fast-list --checksum /path/to/source s3:bucket
```

`--fast-list` trades off API transactions for memory use. As a rough guide rclone uses 1k of memory per object stored, so using `--fast-list` on a sync of a million objects will use roughly 1 GB of RAM.

If you are only copying a small number of files into a big repository, then using `--no-traverse` is a good idea. This finds objects directly instead of through directory listings. You can do a "top-up" sync very cheaply by using `--max-age` and `--no-traverse` to copy only recent files, eg

```
rclone copy --min-age 24h --no-traverse /path/to/source s3:bucket
```

You would do a full `rclone sync` less often. Note that `--fast-list` isn't required in the top-up sync.

Avoiding HEAD requests after PUT

By default rclone will HEAD every object it uploads. It does this to check that the object was uploaded correctly. You can disable this with the `--s3-no-head` option. Setting this flag increases the chance for undetected upload failures.

Hashes

For small objects which weren't uploaded as multipart uploads (objects sized below `--s3-upload-cutoff` if uploaded with rclone) rclone uses the `ETag` header as an MD5 checksum.

For objects that were uploaded as multipart uploads or with server side encryption (SSE-AWS or SSE-C), the `ETag` header is no longer the MD5 sum of the data, so rclone adds an additional piece of metadata `X-Amz-Meta-Md5chsum` which is a base64 encoded MD5 hash (in same format required for `Content-MD5`).

For large objects, calculating this hash can take time, so this hash can be disabled with `--s3-disable-checksum`. This will mean that these objects do not have an MD5 checksum.

Reading this from the object takes an additional `HEAD` request; metadata isn't returned in object listings.

Cleanup

If you run `rclone cleanup s3:bucket` it will remove all pending multipart uploads older than 24 hours. You can use the `-i` flag to see exactly what it will do. If you want more control over the expiry date, then run `rclone backend cleanup s3:bucket -o max-age=1h` to expire all uploads older than one hour. You can use `rclone backend list-multipart-uploads s3:bucket` to see the pending multipart uploads.

Restricted filename characters

S3 allows any valid UTF-8 string as a key. Invalid UTF-8 bytes will be [replaced](#), as they can't be used in XML. The following characters are replaced since these are problematic when dealing with the REST API:

Character	Value	Replacement
NUL	0x00	□
/	0x2F	/

Multipart uploads

rclone supports multipart uploads with S3 which means that it can upload files bigger than 5GB. Note that files uploaded *both* with multipart upload *and* through crypt remotes do not have MD5 sums.

rclone switches from single part uploads to multipart uploads at the point specified by `--s3-upload-cutoff`. This can be a maximum of 5GB and a minimum of 0 (ie always upload multipart files).

Chunk sizes used in the multipart upload are specified by `--s3-chunk-size` and the number of chunks uploaded concurrently is specified by `--s3-upload-concurrency`.

Multipart uploads will use `--transfers` * `--s3-upload-concurrency` * `--s3-chunk-size` extra memory. Single part uploads do not use extra memory.

Single part transfers can be faster than multipart transfers - or slower - depending on your latency from S3 - the more latency, the more likely single part transfers will be faster.

Increasing `--s3-upload-concurrency` will increase throughput (8 is a sensible value) and increasing `--s3-chunk-size` also increases throughput (16M is sensible). Increasing either will use more memory. The default values are high enough to gain most of the possible performance without using too much memory.

Glacier and Glacier Deep Archive

If rclone tries to access data from the Glacier storage class you will see an error like below.

```
2017/09/11 19:07:43 Failed to sync: failed to open source object: Object in GLACIER,
restore first: path/to/file
```

In this case, you need to [restore](#) the object(s) in question before using rclone. rclone only speaks the S3 API; it does not speak the Glacier Vault API, so rclone cannot directly access Glacier Vaults.

Using AWS CLI

1. Install AWS CLI Shell.
2. Configure with the same type of setup as rclone in the command prompt.

<https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-using.html>