

Local Machine to S3 Bucket Data Migration

Step 1 : Create an IAM user.

First step is to create an IAM user with S3FullAccess policy and programmatic access.

The screenshot shows the AWS IAM console 'Create user' wizard, Step 1: Specify user details. The 'User name' field is filled with 's3user'. Below it, a note states: 'The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , @ _ - (hyphen)'. There is an unchecked checkbox for 'Provide user access to the AWS Management Console - optional'. A blue information box at the bottom states: 'If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. [Learn more](#)'. At the bottom right are 'Cancel' and 'Next' buttons.

us-east-1 console.aws.amazon.com/iam/home?region=us-east-1#/users/create

Specify user details

Step 1
Specify user details

Step 2
Set permissions

Step 3
Review and create

User details

User name

s3user

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , @ _ - (hyphen)

☐ Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a best practice [to](#) manage their access in IAM Identity Center.

Information If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. [Learn more](#)

Cancel Next

The screenshot shows the AWS IAM console 'Create user' wizard, Step 3: Review and create. It displays a summary of the user details and permissions. The 'User details' section shows 'User name: s3user', 'Console password type: None', and 'Require password reset: No'. The 'Permissions summary' section shows two policies: 'AdministratorAccess' (AWS managed - job function, Permissions policy) and 'AmazonS3FullAccess' (AWS managed, Permissions policy). The 'Tags - optional' section shows 'No tags associated with the resource.' At the bottom are 'Cancel' and 'Next' buttons.

us-east-1 console.aws.amazon.com/iam/home?region=us-east-1#/users/create

Review and create

Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

User details

User name	Console password type	Require password reset
s3user	None	No

Permissions summary

Name	Type	Used as
AdministratorAccess	AWS managed - job function	Permissions policy
AmazonS3FullAccess	AWS managed	Permissions policy

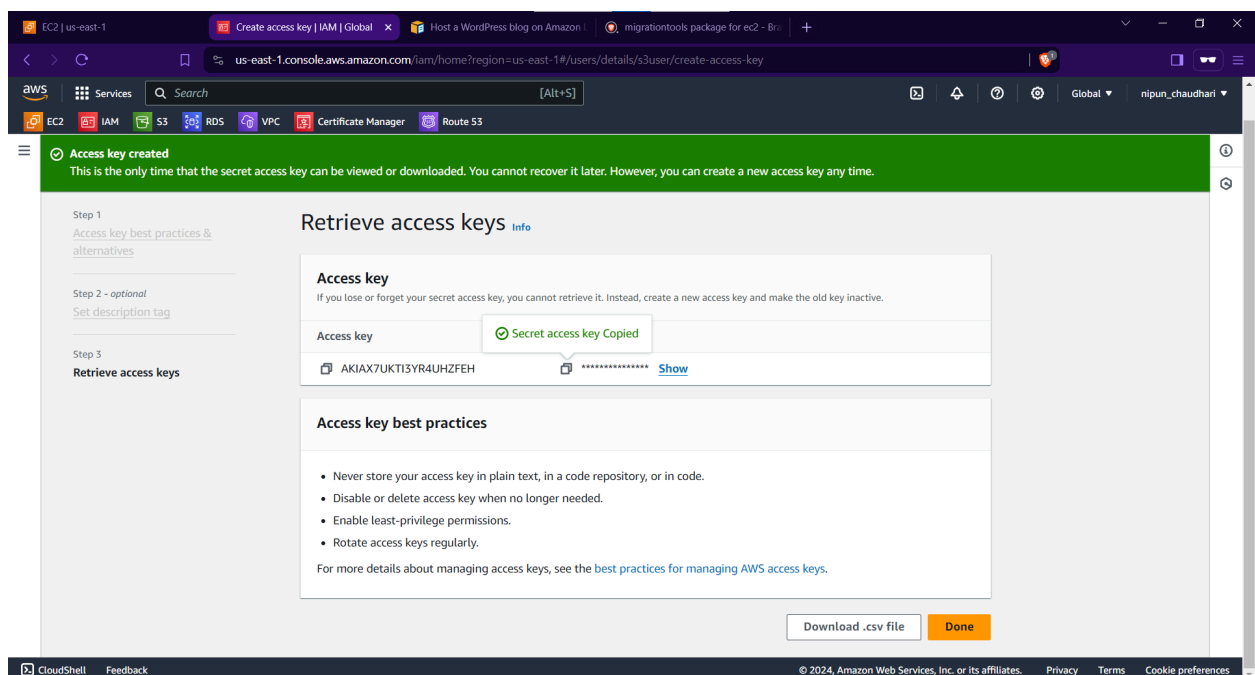
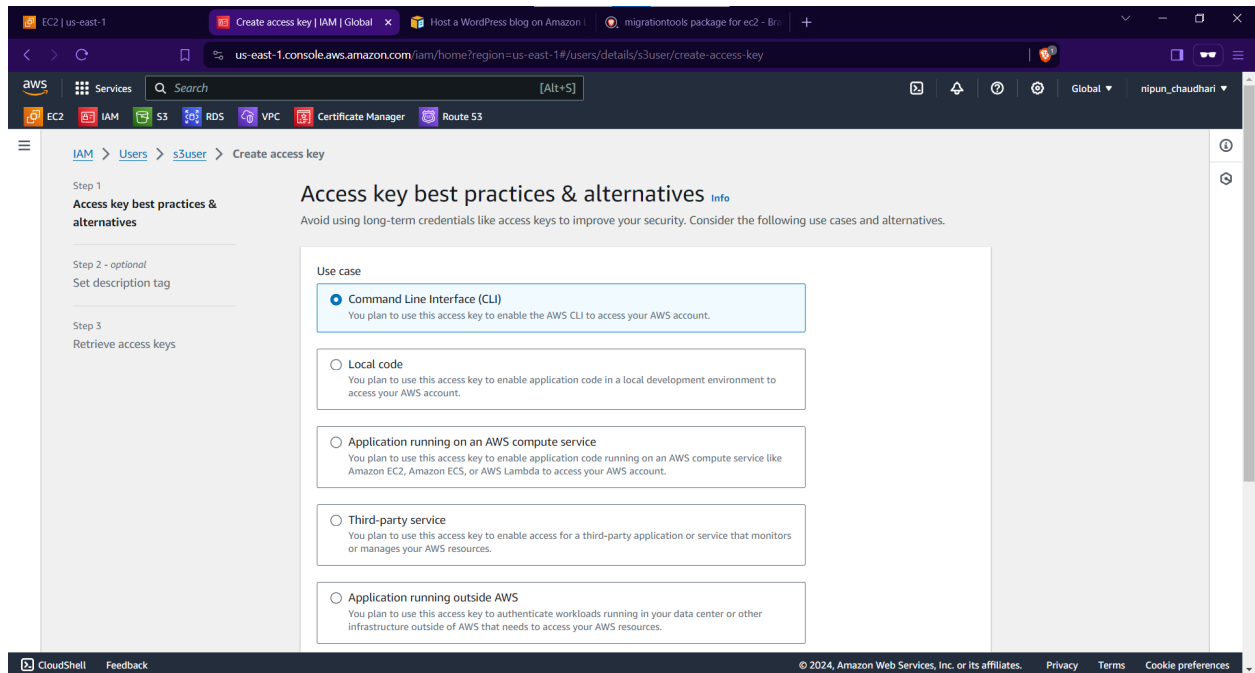
Tags - optional

Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

Cancel Next

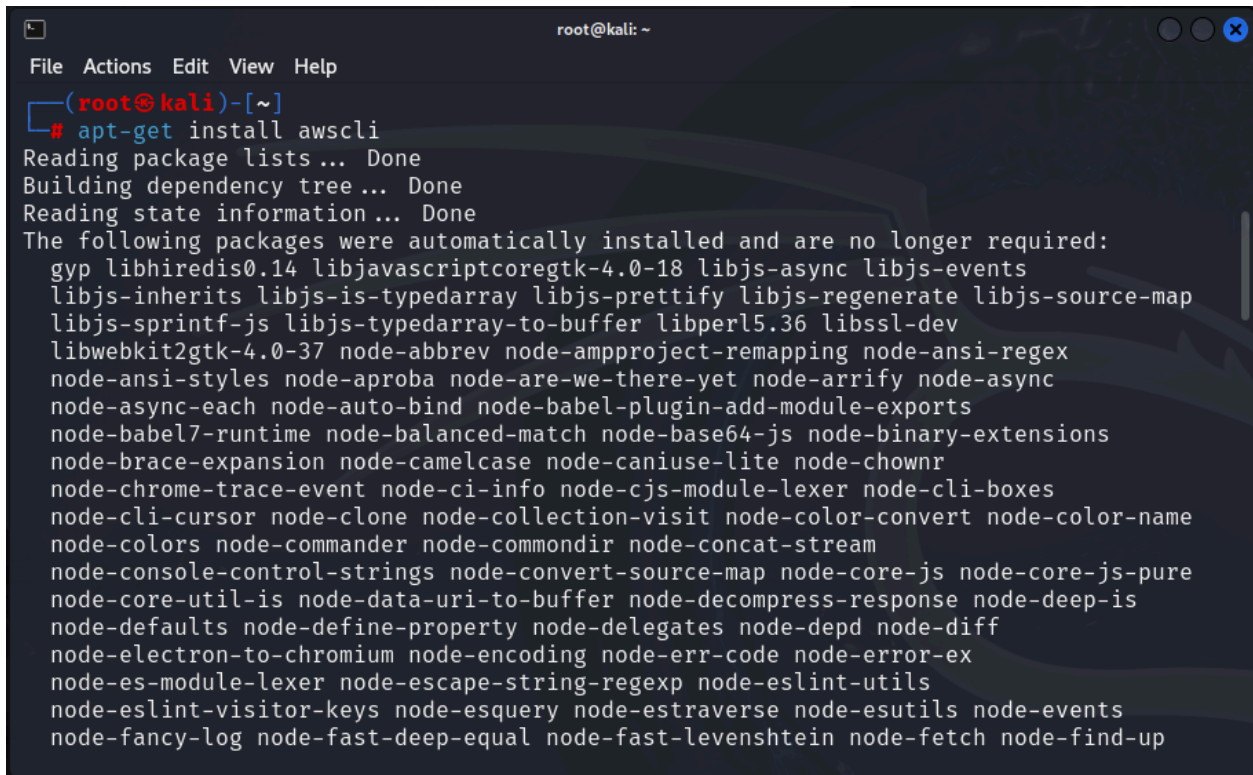
Create an access key for the user, we will use this key to get access to an AWS account.



Step 2 : Get the access of an AWS account of your local machine.

To get the programmatic access of an AWS account, we need to install awscli client on our local system using following command.

apt-get install awscli

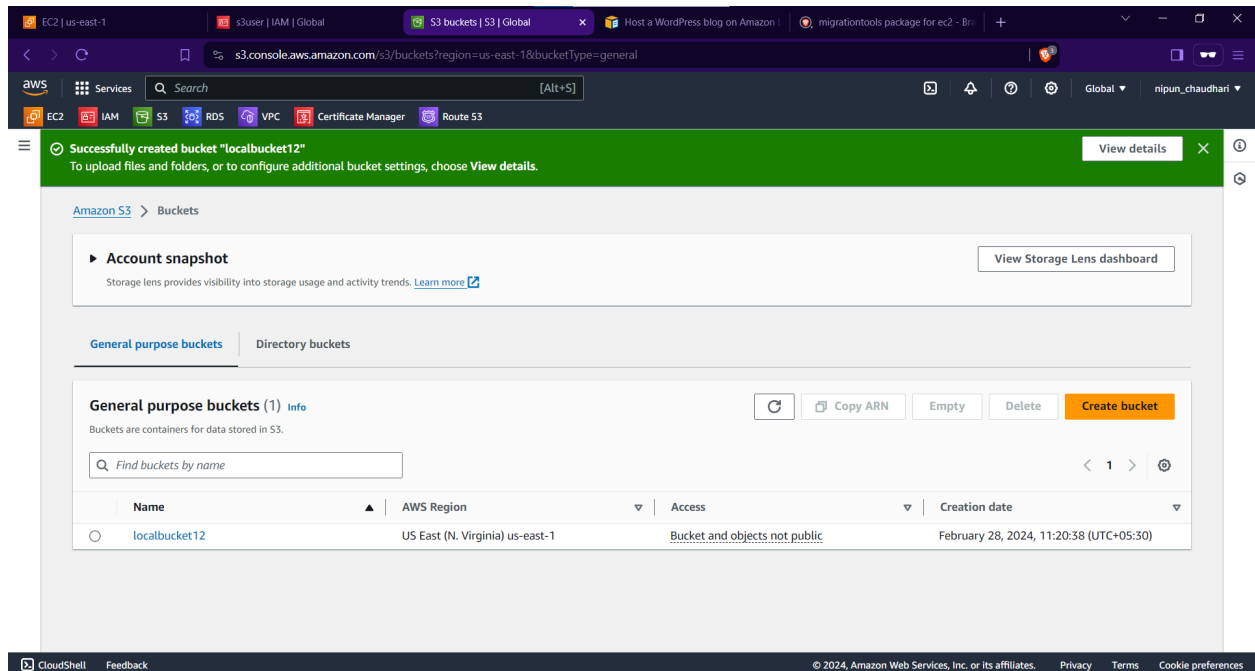
A terminal window titled 'root@kali: ~' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(root@kali)-[~]'. The command '# apt-get install awscli' has been entered. The output shows the package lists being read, the dependency tree being built, and state information being read. It then lists a large number of packages that were automatically installed and are no longer required, including gyp, libhiredis0.14, libjavascriptcoregtk-4.0-18, libjs-async, libjs-events, libjs-inherits, libjs-is-typedarray, libjs-prettify, libjs-regenerate, libjs-source-map, libjs-sprintf-js, libjs-typedarray-to-buffer, libperl5.36, libssl-dev, libwebkit2gtk-4.0-37, node-abbrev, node-ampproject-remapping, node-ansi-regex, node-ansi-styles, node-aproba, node-are-we-there-yet, node-arrify, node-async, node-async-each, node-auto-bind, node-babel-plugin-add-module-exports, node-babel7-runtime, node-balanced-match, node-base64-js, node-binary-extensions, node-brace-expansion, node-camelcase, node-caniuse-lite, node-chownr, node-chrome-trace-event, node-ci-info, node-cjs-module-lexer, node-cli-boxes, node-cli-cursor, node-clone, node-collection-visit, node-color-convert, node-color-name, node-colors, node-commander, node-commandir, node-concat-stream, node-console-control-strings, node-convert-source-map, node-core-js, node-core-js-pure, node-core-util-is, node-data-uri-to-buffer, node-decompress-response, node-deep-is, node-defaults, node-define-property, node-delegates, node-depd, node-diff, node-electron-to-chromium, node-encoding, node-err-code, node-error-ex, node-es-module-lexer, node-escape-string-regexp, node-eslint-utils, node-eslint-visitor-keys, node-esquery, node-estraverse, node-esutils, node-events, node-fancy-log, node-fast-deep-equal, node-fast-levenshtein, node-fetch, and node-find-up.

```
root@kali: ~  
File Actions Edit View Help  
(root@kali)-[~]  
# apt-get install awscli  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following packages were automatically installed and are no longer required:  
gyp libhiredis0.14 libjavascriptcoregtk-4.0-18 libjs-async libjs-events  
libjs-inherits libjs-is-typedarray libjs-prettify libjs-regenerate libjs-source-map  
libjs-sprintf-js libjs-typedarray-to-buffer libperl5.36 libssl-dev  
libwebkit2gtk-4.0-37 node-abbrev node-ampproject-remapping node-ansi-regex  
node-ansi-styles node-aproba node-are-we-there-yet node-arrify node-async  
node-async-each node-auto-bind node-babel-plugin-add-module-exports  
node-babel7-runtime node-balanced-match node-base64-js node-binary-extensions  
node-brace-expansion node-camelcase node-caniuse-lite node-chownr  
node-chrome-trace-event node-ci-info node-cjs-module-lexer node-cli-boxes  
node-cli-cursor node-clone node-collection-visit node-color-convert node-color-name  
node-colors node-commander node-commandir node-concat-stream  
node-console-control-strings node-convert-source-map node-core-js node-core-js-pure  
node-core-util-is node-data-uri-to-buffer node-decompress-response node-deep-is  
node-defaults node-define-property node-delegates node-depd node-diff  
node-electron-to-chromium node-encoding node-err-code node-error-ex  
node-es-module-lexer node-escape-string-regexp node-eslint-utils  
node-eslint-visitor-keys node-esquery node-estraverse node-esutils node-events  
node-fancy-log node-fast-deep-equal node-fast-levenshtein node-fetch node-find-up
```

Step 3 : Create a S3 bucket

In order to migrate data from a local machine to S3 bucket, we need to create a S3 bucket. This can be done through AWS S3 service console.

To create S3 bucket, we specify the bucket name and the region where we want to create that bucket.



Step 4 : Move data from local machine to S3 bucket.

Final step is to move the data into S3 bucket. To do this, first log-in into the AWS account using the Access key and Secret key we created for our IAM user.

Use following command to get the access of an AWS account.

aws configure

It will prompt for access key and secret key, provide required credentials to log-in into the account. Select the region and output format as per your requirement.

```
root@kali: ~  
File Actions Edit View Help  
(root@kali)-[~]  
# aws configure  
AWS Access Key ID [None]: AKIA7UKTI3YR4UHZFEH  
AWS Secret Access Key [None]: t7afUWlhKm3sCk6YNWsPUaf9k8jJnFRSFHl+eia9  
Default region name [None]:  
Default output format [None]: json  
  
(root@kali)-[~]  
#
```

Create a folder and create a file in that folder. Next we navigate to the folder we created. We have a file in this folder to upload this file to S3 bucket use following command.

aws s3 sync . s3://bucketname

Replace the **bucketname** with your own bucket name.

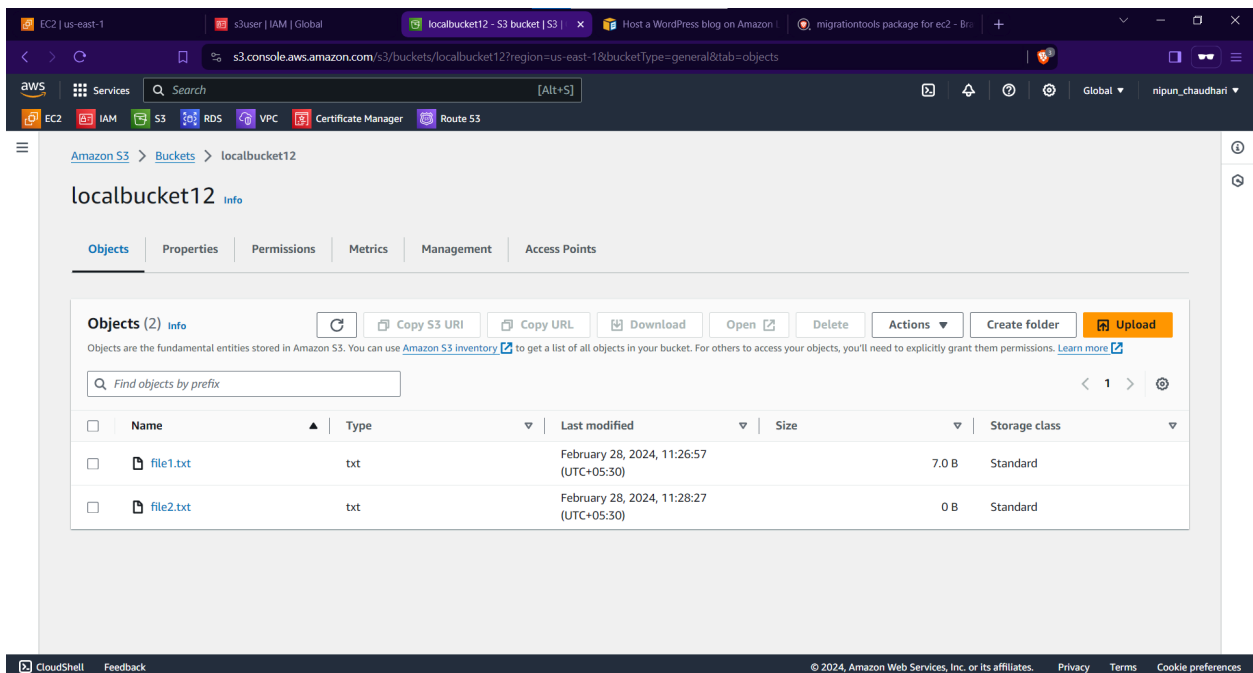
```
root@kali: ~/upload
File Actions Edit View Help
(root@kali)~[~/upload]
# cd file2.txt
cd: no such file or directory: file2.txt

(root@kali)~[~/upload]
# touch file2.txt

(root@kali)~[~/upload]
# aws s3 sync . s3://localbucket12
upload: ./file2.txt to s3://localbucket12/file2.txt

(root@kali)~[~/upload]
#
```

Check whether the files are uploaded into the bucket or not.



Here the uploading is done successfully.