# S3 File Uploader

## Amazon S3 and IAM Configuration

Author Vikas Singhal | wpwave.com

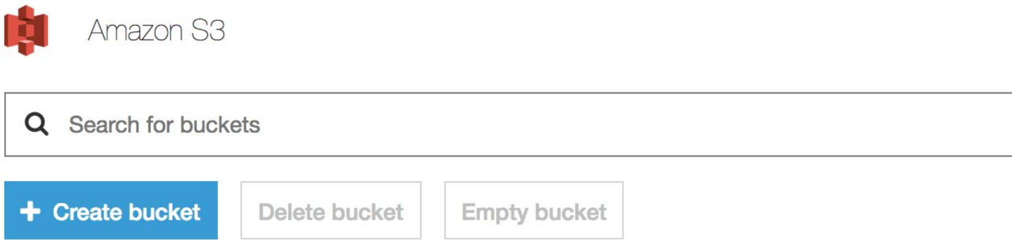
Buy Plugin at

https://codecanyon.net/item/s3-file-uploader-simply-drag-and-drop-your-files-to-upload-into-the-cloud/20824262

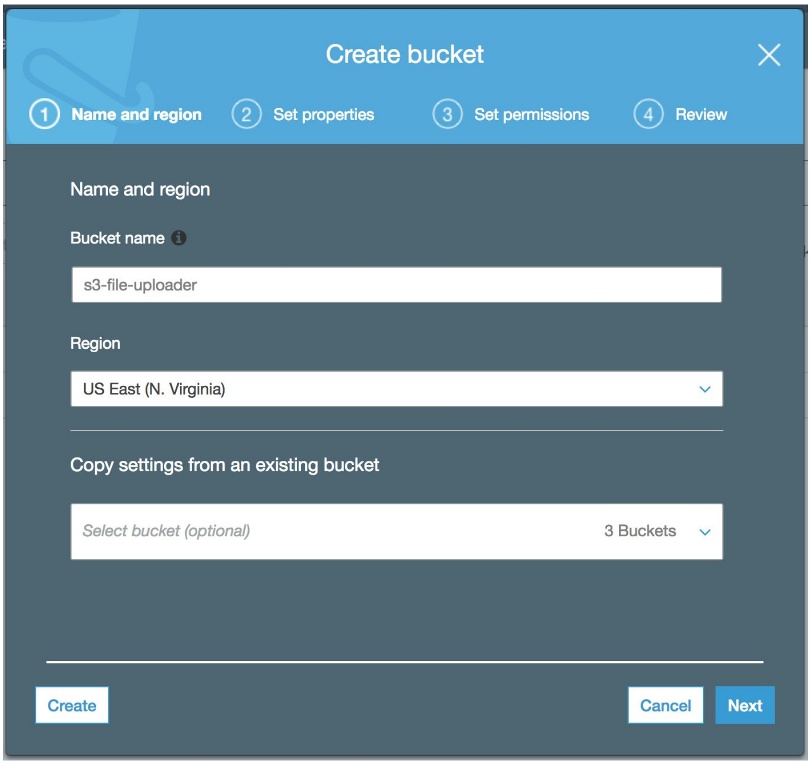
In order to use this plugin you need to create two items in your Amazon AWS Console – S3 bucket and an IAM user. Lets get started.

## Create a S3 Bucket

1. Navigate to https://s3.console.aws.amazon.com/s3/home and click on **Create Bucket**.



2. Now enter the bucket name and click on Create. [No need to set anything else at this point]



Name and region

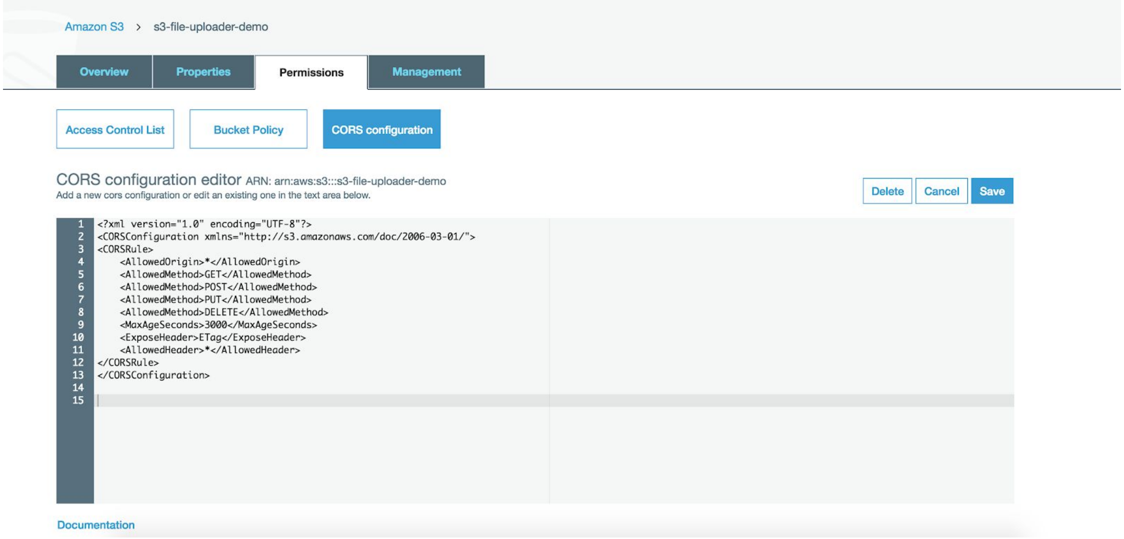
Bucket name: s3-file-uploader

Region: US East (N. Virginia)

Copy settings from an existing bucket

Select bucket (optional) 3 Buckets

3. Once the bucket is created, select the bucket and you will see following options, go to Permissions > CORS Configuration tab and enter this text.



CORS configuration editor ARN: arn:aws:s3:::s3-file-uploader-demo
Add a new cors configuration or edit an existing one in the text area below.

```
<?xml version="1.0" encoding="UTF-8"?>
<CORSConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<CORSRule>
    <AllowedOrigin>*</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
    <MaxAgeSeconds>3000</MaxAgeSeconds>
    <ExposeHeader>ETag</ExposeHeader>
    <AllowedHeader>*</AllowedHeader>
</CORSRule>
</CORSConfiguration>
```

Text to enter:

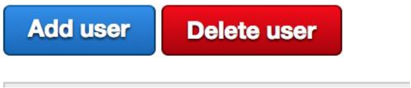
```
<?xml version="1.0" encoding="UTF-8"?>
<CORSConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<CORSRule>
    <AllowedOrigin>*</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
    <MaxAgeSeconds>3000</MaxAgeSeconds>
    <ExposeHeader>ETag</ExposeHeader>
    <AllowedHeader>*</AllowedHeader>
</CORSRule>
</CORSConfiguration>
```

4. At this point your bucket is created successfully.

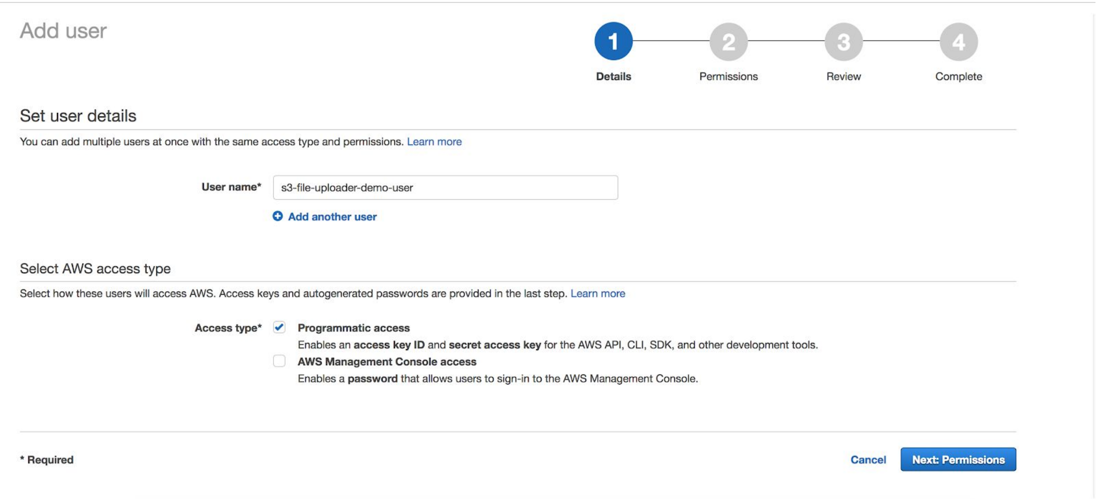
## Create an IAM User

We will now create a user who will be allowed to access the bucket so that our script can start working.

1. Navigate to IAM Users section in the Amazon AWS console - https://console.aws.amazon.com/iam/home and click on Add User.



2. Enter the user name and access type as “Programmatic access” and click Next.



Add user

Set user details

You can add multiple users at once with the same access type and permissions. Learn more

**User name*** s3-file-uploader-demo-user

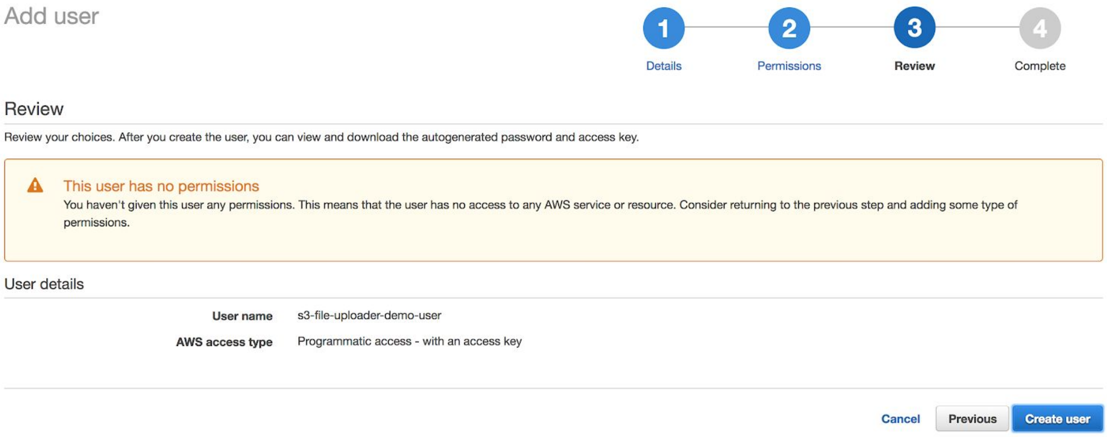
Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. Learn more

**Access type*** ☑ **Programmatic access**
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.
□ **AWS Management Console access**
Enables a **password** that allows users to sign-in to the AWS Management Console.

* **Required**

3. There is no need to add any permission at this point, just click next and then Create User.



Add user

Review

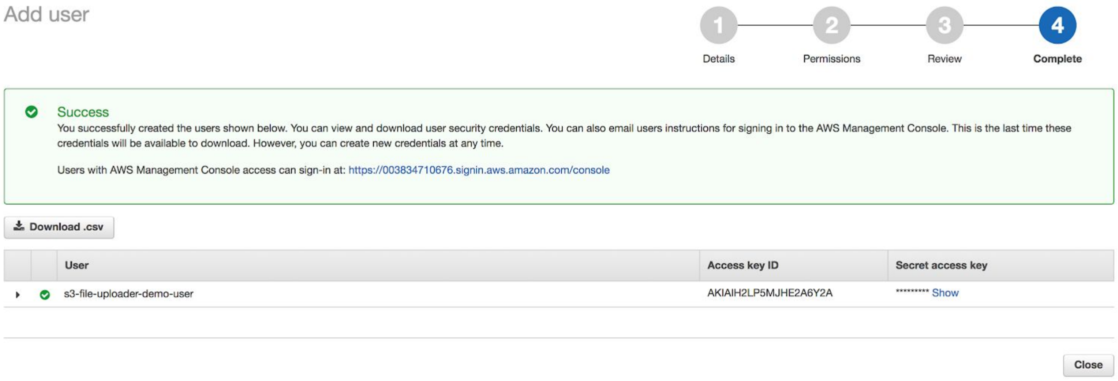
Review your choices. After you create the user, you can view and download the autogenerated password and access key.

This user has no permissions
You haven't given this user any permissions. This means that the user has no access to any AWS service or resource. Consider returning to the previous step and adding some type of permissions.

User details

| | |
|---|---|
| **User name** | s3-file-uploader-demo-user |
| **AWS access type** | Programmatic access - with an access key |

4. In the last step you will be shown an Access Key ID and Secret access key, copy and paste these values in a text file (and download the CSV for future reference).



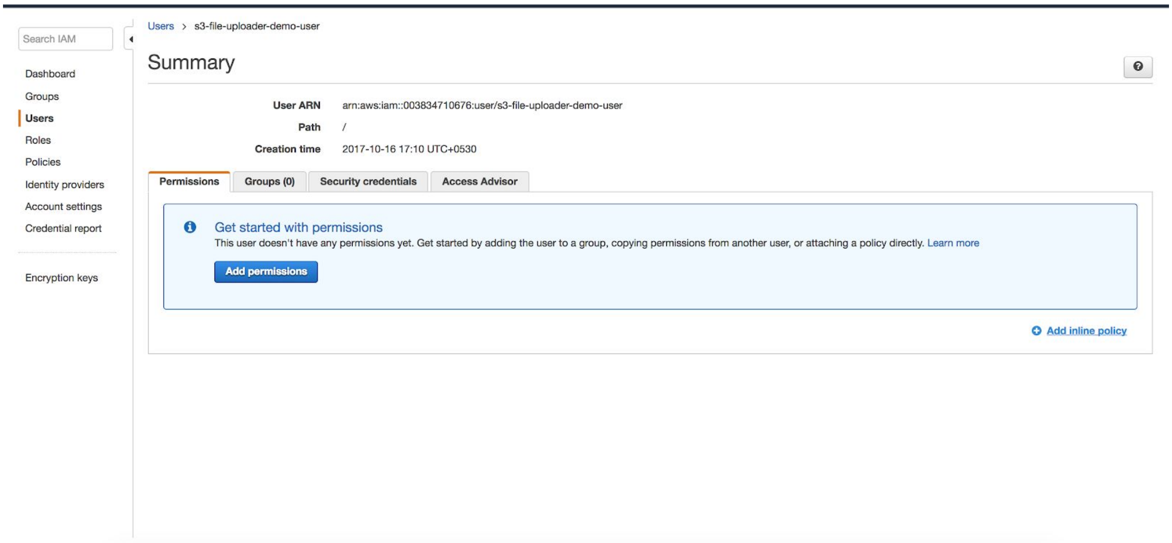
Add user

Success
You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: https://003834710676.signin.aws.amazon.com/console

| | | User | Access key ID | Secret access key |
|---|---|---|---|---|
| ▸ | ✓ | s3-file-uploader-demo-user | AKIAIH2LP5MJHE2A6Y2A | ********* Show |

5. Once the user is created, you can edit its properties to add access to our S3 bucket created in previous section.



Summary

User ARN arn:aws:iam::003834710676:user/s3-file-uploader-demo-user

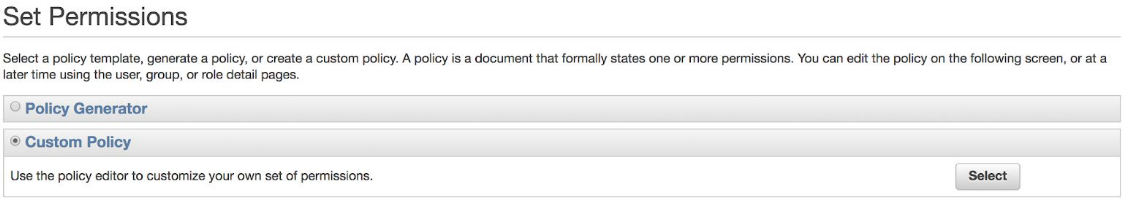
Path /

Creation time 2017-10-16 17:10 UTC+0530

Get started with permissions

This user doesn't have any permissions yet. Get started by adding the user to a group, copying permissions from another user, or attaching a policy directly. Learn more

6. Click on Add Inline Policy and then select Custom Policy.



Set Permissions

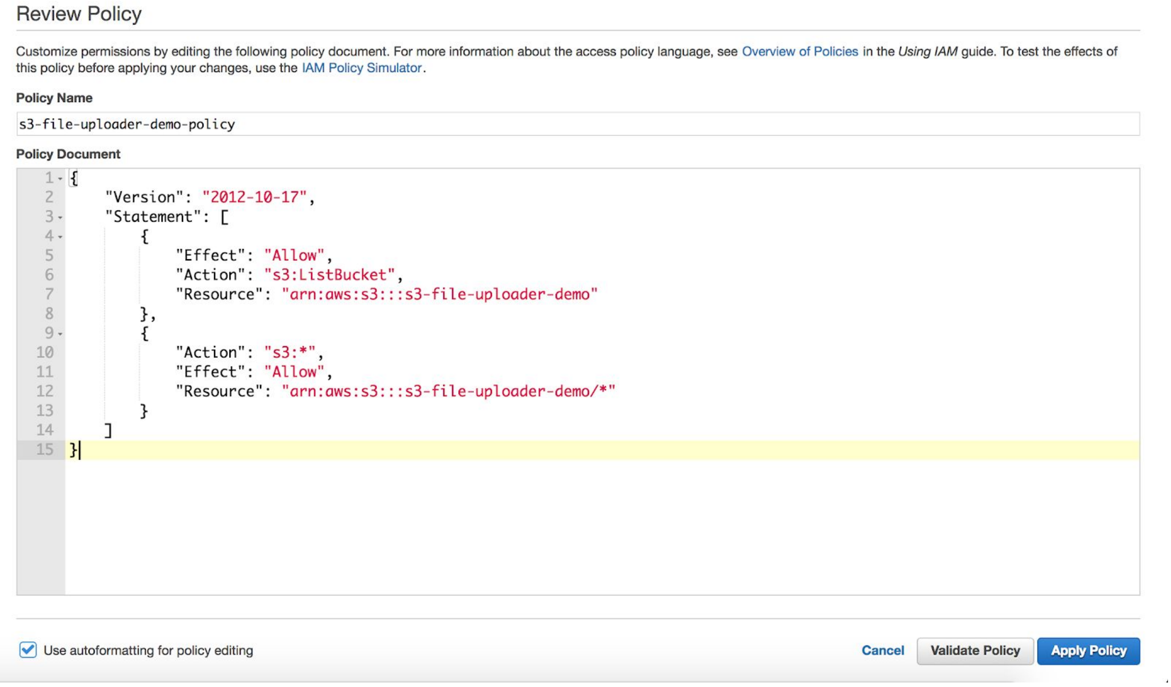
Select a policy template, generate a policy, or create a custom policy. A policy is a document that formally states one or more permissions. You can edit the policy on the following screen, or at a later time using the user, group, or role detail pages.

○ Policy Generator

● Custom Policy

Use the policy editor to customize your own set of permissions.

7. Paste the following text in the policy document area.



Review Policy

Customize permissions by editing the following policy document. For more information about the access policy language, see Overview of Policies in the *Using IAM* guide. To test the effects of this policy before applying your changes, use the IAM Policy Simulator.

Policy Name

s3-file-uploader-demo-policy

Policy Document

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::s3-file-uploader-demo"
        },
        {
            "Action": "s3:*",
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::s3-file-uploader-demo/*"
        }
    ]
}
```

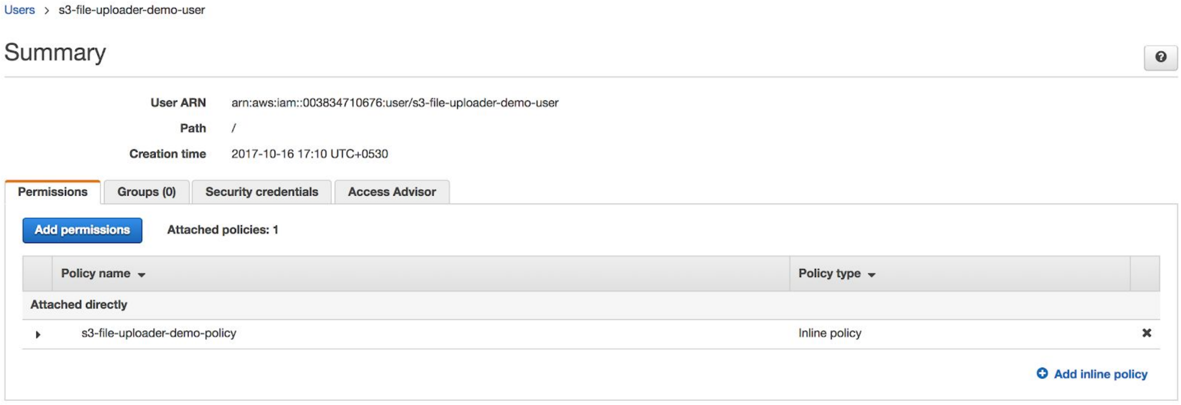
☑ Use autoformatting for policy editing

Notice the bucket name in the policy document, you need to replace it with the S3 bucket name you have used in previous section, rest of the details will be same.

Text to paste:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::s3-file-uploader-demo"
        },
        {
            "Action": "s3:*",
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::s3-file-uploader-demo/*"
        }
    ]
}
```

8. Click on Apply policy and we are done!



User ARN arn:aws:iam::003834710676:user/s3-file-uploader-demo-user

Path /

Creation time 2017-10-16 17:10 UTC+0530

Attached policies: 1

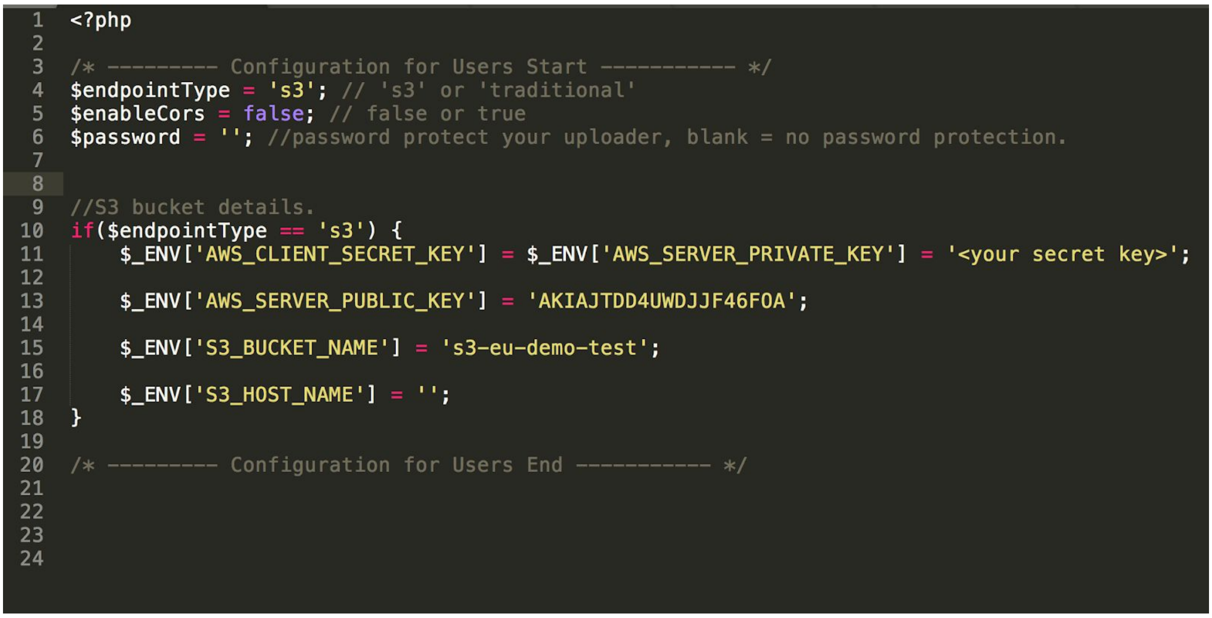
| Policy name | Policy type |
|---|---|
| Attached directly | |
| s3-file-uploader-demo-policy | Inline policy |

## Configure the Script

In this last section, you need to configure the script with the access details you obtained in last two sections.

1. **config.php** – Enter the Secret access key in the line number 11, Access key ID in line number 13 and the bucket name in line number 15. Leave S3 host name as empty.

If you want to protect your uploader script with a password, enter it in line number 6 or else leave it empty.



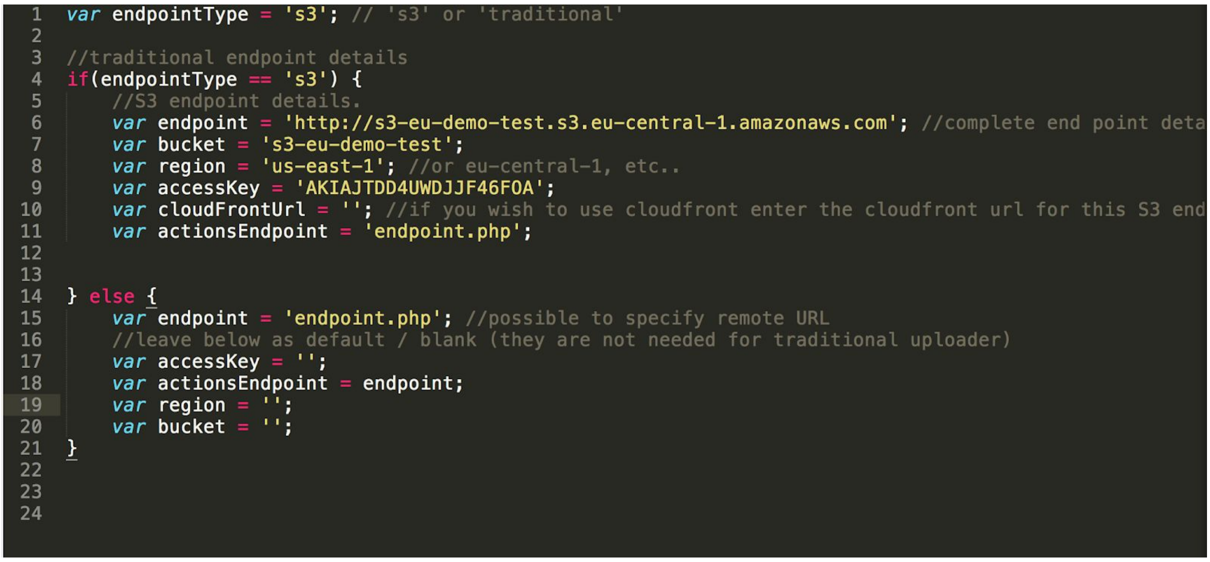
```php
<?php

/* --------- Configuration for Users Start ----------- */
$endpointType = 's3'; // 's3' or 'traditional'
$enableCors = false; // false or true
$password = ''; //password protect your uploader, blank = no password protection.


//S3 bucket details.
if($endpointType == 's3') {
    $_ENV['AWS_CLIENT_SECRET_KEY'] = $_ENV['AWS_SERVER_PRIVATE_KEY'] = '<your secret key>';

    $_ENV['AWS_SERVER_PUBLIC_KEY'] = 'AKIAJTDD4UWDJJF46FOA';

    $_ENV['S3_BUCKET_NAME'] = 's3-eu-demo-test';

    $_ENV['S3_HOST_NAME'] = '';
}

/* --------- Configuration for Users End ----------- */
```

2. **config.js** – This file is used by browser to determine your S3 details.

Enter the details as :

- endpoint - full endpoint URL to your bucket.
- bucket - just the bucket name
- region - where is your bucket hosted within S3.
- accessKey - which you obtained during IAM user creation

(Optional) If you are using Amazon Cloudfront for CDN, you can enter the domain name (with https://) here.



```javascript
var endpointType = 's3'; // 's3' or 'traditional'

//traditional endpoint details
if(endpointType == 's3') {
    //S3 endpoint details.
    var endpoint = 'http://s3-eu-demo-test.s3.eu-central-1.amazonaws.com'; //complete end point deta
    var bucket = 's3-eu-demo-test';
    var region = 'us-east-1'; //or eu-central-1, etc..
    var accessKey = 'AKIAJTDD4UWDJJF46FOA';
    var cloudFrontUrl = ''; //if you wish to use cloudfront enter the cloudfront url for this S3 end
    var actionsEndpoint = 'endpoint.php';


} else {
    var endpoint = 'endpoint.php'; //possible to specify remote URL
    //leave below as default / blank (they are not needed for traditional uploader)
    var accessKey = '';
    var actionsEndpoint = endpoint;
    var region = '';
    var bucket = '';
}
```

The process of configuring your script and Amazon AWS is completed now. You should simply upload the script folder to your server and enjoy your file uploader!

**Traditional Server**

If you don’t want to use S3 for uploads and use your own server to store files, change the **endpointType** to **traditional** in both config.php and config.js. Rest of the options are self explainatory.

File will be stored under vendor/fineuploader/php-traditional-server/files.