



Search Medium



You have **2 free member-only stories left** this month. [Sign up](#) for Medium and get an extra one.

◆ Member-only story

# Sync your AWS CodeCommit repository to your GitHub repository for public code sharing or private repository backup

Paris Nakita Kejser · [Follow](#)

9 min read · Aug 27, 2022

[Listen](#)[Share](#)

# git

After starting using Amazon Web Service (AWS) for real use, I'm falling in love with the CI/CD pipeline on AWS, I feel its easier to integrate with services when you use CodeCommit for Git, CodeBuild for the building process, and CodeDeploy when you deploy your software out but you can use different services out there, but in my case, I fell good to keep all my code on AWS.

The problem can be when you will do more “*open source*” projects or just share your code with the public, I have a different repository on GitHub from legacy time and now

I want to change it from GitHub to AWS CodeCommit.

Then I need in some cases to sync my Git repository from AWS CodeCommit to a GitHub repository, and it can be many cases why I want to do this, so I should find an easy way to do that.

## **Manually sync repository from between two GIT remote repositories**

When you are starting you can do it free by running the commands manually every time you need it, its costs your time but if you see your time as free it's an easy way to start and you need to understand how it works before you are doing it automatically anyway.

First, we need to clone our primary repository

```
1 git clone --mirror https://primary_repo_url/primary_repo.git
```

git-clone-repo-as-mirror.sh hosted with ❤ by GitHub



[view raw](#)

then you need to go into your repo folder and add the new remote repository.

```
1 cd primary_repo.git
```

```
2 git remote add --mirror=fetch secondary https://secondary_repo_url/secondary_repo.git
```

git-add-remote-repo.sh hosted with ❤ by GitHub

[view raw](#)

Now you are ready to fetch out on your origin (primary) remote repository, and after it pushes everything to your secondary remote repository.

```
1 git fetch origin
```

```
2 git push secondary --all
```

git-fetch-push-all.sh hosted with ❤ by GitHub

[view raw](#)

If you do changes in your primary (origin) repository you can just re-run these 2 lines in your terminal, then it will fetch your data from the remote origin and push it to the second remote repository.

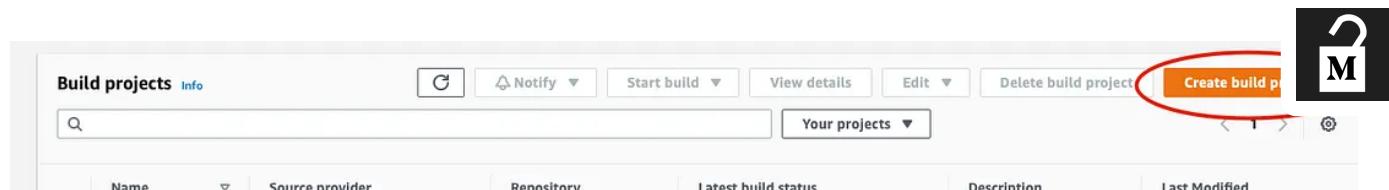
## **Use CodeBuild to sync the CodeCommit repository to the GitHub repository with Event Bridge and Lambda function**

As said before if your time does not cost you money, then it's okay to do it manually, but automatic is much better so you don't need to remember the boring stuff over and over again.

So the steps will first add your GitHub provider connection to your AWS account using CLI and after that, we want to create the AWS CodeBuild file there will do the sync between AWS CodeCommit and GitHub repositories every time there are changes on your source repositories on AWS CodeCommit using Event Bridge and a Lambda

### **Setup your AWS CodeBuild inline build spec YAML and run your sync inside the AWS Web Console manually**

Go to AWS CodeBuild in your console, create your project and prepare the configuration for our build project.



First we need to fill out the project configuration settings with a project name, if you want a build badge you can check it on, but for now we are not need this option.

**Project configuration**

**Project name**

git-repo-sync-test-project

A project name must be 2 to 255 characters. It can include the letters A-Z and a-z, the numbers 0-9, and the special characters - and \_.

**Description - optional**

**Build badge - optional**

Enable build badge

**Enable concurrent build limit - optional**

Limit the number of allowed concurrent builds for this project.

Restrict number of concurrent builds this project can start

**▶ Additional configuration**

tags



Now we need to select our source provider (AWS CodeCommit) and find the repository you want to sync from an AWS CodeCommit repository to a GitHub repository, after you have selected the repository select Branch on reference type and select your main branch in my case it called “master”

Source Add source

Source 1 - Primary

Source provider AWS CodeCommit

Repository Q X

Reference type  
Choose the source version reference type that contains your source code.

Branch

Git tag

Commit ID

Branch  
Choose a branch that contains the code to build.

master ▼

Commit ID - optional  
Choose a commit ID. This can shorten the duration of your build.

Source version Info

refs/heads/master

**d7d30074** update app file to generate the infrastructure code

► Additional configuration  
Git clone depth, Git submodules

For our environment, we want to use **Ubuntu** as the operating system and the image is **aws/codebuild/standard:6.0** for now and later our environment create a new service role, you can always create your own role and then change it but in the beginning, it's easier to let CodeBuild create this role automatically when you create the project.

## Environment

Environment image

**Managed image**  
Use an image managed by AWS CodeBuild

**Custom image**  
Specify a Docker image

Operating system

Ubuntu 

 The programming language runtimes are now included in the standard image of Ubuntu 18.04, which is recommended for new CodeBuild projects created in the console. See [Docker Images Provided by CodeBuild for details](#).

Runtime(s)

Standard 

Image

aws/codebuild/standard:6.0 

Image version

Always use the latest image for this runtime version

Environment type

Linux

Privileged

Enable this flag if you want to build Docker images or want your builds to get elevated privileges

Service role 

**New service role**  
Create a service role in your account

**Existing service role**  
Choose an existing service role from your account

Role name

codebuild-git-repo-sync-test-project-service-role

Type your service role name

► Additional configuration  
Timeout, certificate, VPC, compute type, environment variables, file systems

In Buildspec you should use **Insert build commands**, then switch to an editor so its

looks like the images below

The screenshot shows the AWS CodeBuild buildspec editor interface. At the top left, it says "Buildspec". Below that, "Build specifications" are listed: "Use a buildspec file" (unchecked) and "Insert build commands" (checked). A red arrow points from the text "looks like the images below" to the "Insert build commands" option. In the main area, "Build commands" are displayed as a YAML snippet:

```
1 version: 0.2
2
3 #env:
4 #variables:
5     # key: "value"
6     # key: "value"
7 #parameter-store:
8     # key: "value"
9     # key: "value"
10 #secrets-manager:
11     # key: secret-id:json-key:version-stage:version-id
12     # key: secret-id:json-key:version-stage:version-id
13 #exported-variables:
14     # - variable
15     # - variable
16 #git-credential-helper: yes
17 #batch:
18     #fast-fail: true
19 #build-list:
20 #build-matrix:
21 #build-graph:
22 phases:
```

A red arrow also points from the word "commands" in "Build commands" to the first line of the YAML snippet. On the right side of the editor, there is a small icon of a padlock with a keyhole.

when you have this editor ready you can paste my build spec YAML snippet into the editor

```

1  version: 0.2
2
3  env:
4      git-credential-helper: yes
5
6  phases:
7      install:
8          commands:
9              - echo "STARTING PYTHON INSTALLATION"
10             - pip install git-remote-codecommit
11
12     pre_build:
13         commands:
14             - echo CLONE DIRECTORIES
15             - mkdir /usr/bin/repo
16             - cd /usr/bin/repo
17             - git clone --mirror codecommit::{region}://{repo} {repo}
18
19     build:
20         commands:
21             - cd /usr/bin/repo/{repo}
22             - git remote add --mirror=fetch github https://<token>@github.com/{username}/{repo}
23
24     post_build:
25         commands:
26             - git fetch origin
27             - git push github --all

```



[buildspec.yml](#) hosted with ❤ by GitHub

[view raw](#)

This snippet will install the `git-remote-codecommit` python package, clone your CodeCommit repository with the `mirror` flag, then we are going into the folder and add our GitHub remote path with the token and `mirror` tag with `fetch` at the end of it we are fetching from the origin (CodeCommit) and push to GitHub repository after it.

### Add your GitHub access token to Secret Manager

Using Secret Manager will be the best way to keep your GitHub access token saved and easy to rotate if needed, if you keep your access token static it will be hard to change it automatically if you build an access token rotation system later or if something happens and you need to change it quickly.

First, let's go into Secrets Manager and create our Secret with Secrets Manager.

The screenshot shows the 'Choose secret type' step in AWS Secrets Manager. On the left, there are four steps: Step 1 (Choose secret type), Step 2 (Configure secret), Step 3 (Configure rotation - optional), and Step 4 (Review). Step 1 is currently active. The main area is titled 'Choose secret type' and contains a 'Secret type' section with five options: 'Credentials for Amazon RDS database', 'Credentials for Amazon DocumentDB database', 'Credentials for Amazon Redshift cluster', 'Credentials for other database', and 'Other type of secret'. The 'Other type of secret' option is selected and highlighted with a blue box and a red arrow pointing to it. Below this is a 'Key/value pairs' section with two tabs: 'Key/value' (selected) and 'Plaintext'. Under 'Key/value', there is a row with a 'key' field containing 'access\_token' and a 'value' field which is empty. A red arrow points to the 'access\_token' key. At the bottom, there is an 'Encryption key' section with a dropdown set to 'aws/secretsmanager' and a 'Add new key' button. On the right side, there is a large black icon with a white 'M' and a padlock symbol. At the bottom right are 'Cancel' and 'Next' buttons.

The secret type we want to pick Other type of secret to store our access token for GitHub, then the key name is access\_token, and paste your token string inside the value field on the right side, after clicking next.

Step 1  
Choose secret type

Step 2  
Configure secret

Step 3  
Configure rotation - optional

Step 4  
Review

## Configure secret

**Secret name and description** Info

**Secret name**  
A descriptive name that helps you find your secret later.  
 

Secret name must contain only alphanumeric characters and the characters /\_+=@~.

**Description - optional**  
  
Maximum 250 characters.

**Tags - optional**

**Resource permissions - optional** Info  
Add or edit a resource policy to access secrets across AWS accounts.

**Replicate secret- optional**  
Create read-only replicas of your secret in other Regions. Replica secrets incur a charge. [Learn more](#) in the User Guide.  

Now select a secret name, I pick **git/github** so I'm ready to use this path for multi GIT providers if needed, when you have done it goes to the next step.

## Configure rotation - optional

### Secret rotation

#### Configure automatic rotation - optional Info

Configure AWS Secrets Manager to rotate this secret automatically.

Automatic rotation

#### Rotation schedule Info

Schedule expression builder

Schedule expression

Time unit

Days

Days

30

#### Window duration - optional

4h

Enter the time in hours.



#### Rotation function

##### Lambda rotation function Info

Choose a Lambda function that can rotate this secret.

AwsDeploymentStack-Contin-ZipTransferArtifactPacka-MxqrtxghaVKI



[Create function](#)

[Cancel](#)

[Previous](#)

[Next](#)

We are not gonna use rotation for this right now, so let's just skip this step with click next.

**Rotation function**

Lambda rotation function

-

Secret that performs rotation

-

**Sample code**

Use these code samples to retrieve the secret in your application.

Java | JavaV2 | JavaScript | C# | **Python3** | Ruby | Go

```

1 # Use this code snippet in your app.
2 # If you need more information about configurations or implementing the sample code, visit the AWS docs:
3 # https://aws.amazon.com/developers/getting-started/python/
4
5 import boto3
6 import base64
7 from botocore.exceptions import ClientError
8
9
10 def get_secret():
11     secret_name = "git/github"
12     region_name = "us-east-1"
13
14     # Create a Secrets Manager client
15     session = boto3.Session()
16     client = session.client(
17         service_name='secretsmanager'.

```

[Download AWS SDK for Python](#)

Cancel Previous **Store**

On the review pages at the bottom of the pages, you can quickly copy code samples if you want to speak with the Secrets Manager from the SDK.

Secret name	Description	Last retrieved (UTC)
git/github	-	-

Secrets

Secrets

Filter secrets by name, description, tag key, tag value, or primary Region

Store a new secret

< 1 > ⌂

When it's storage you can see it in your overview for Secrets Manager, if not then click refresh one time, sometimes the can take a few seconds.

Show we are ready to demonstrate the CLI command to get it out again so we can test it's working before we add it to our AWS CodeBuild process.



```
pnk@MacBook-Pro aws-github-source-provider % aws secretsmanager get-secret-value --secret-id git/github --region us-east-1
{
  "ARN": "arn:aws:secretsmanager:us-east-1:788962042046:secret:git/github-A9DD22",
  "Name": "git/github",
  "VersionId": "29b39d89-f8f4-4949-b248-e507e7488cc8",
  "SecretString": "{\"access_token\":\"\"}",
  "VersionStages": [
    "AWSCURRENT"
  ],
  "CreatedDate": "2022-08-26T11:30:01.829000+02:00"
}
```

As you can see my SecretString is empty for my test here, in the real world it will contain my GitHub access token.

Let's take only the SecretString object out using jq in our terminal so it's easier to work with and convert or string to JSON so it's easier grep our GitHub access token out.



in my case its returns “test” I have to change my value to “test” for our access\_token so we have something to print out.

```
pnk@MacBook-Pro aws-github-source-provider % aws secretsmanager get-secret-value --secret-id git/github --region us-east-1 | jq -r ".SecretString" | jq -r ".access_token"
test
```

Now you are ready to put this command into our AWS CodeBuild process to replace our static access token.



When you now run the build in AWS CodeBuild you will meet the error AccessDeniedException, its because your role did not allow to get access from Secrets Manager

278  
279 An error occurred (AccessDeniedException) when calling the GetSecretValue operation: User: arn:aws:sts::788962042046:assumed-role/codebuild-test-project-sync-repo/AWSCodeBuild-179c1fe9-75cc-455f-9a9c-  
ffd27a81328e is not authorized to perform: secretsmanager:GetSecretValue on resource: git/github because no identity-based policy allows the secretsmanager:GetSecretValue action  
280

So let's go inside IAM and find our role and give it the needed access, and run the process one more time.

The screenshot shows the AWS IAM Roles page. On the left, there is a sidebar with 'Identity and Access Management (IAM)' selected. The main area shows a search bar with 'codebuild-test-project-sync-repo' and a result table with one item. The table has columns for 'Role name' (codebuild-test-project-sync-repo) and 'Trusted entities' (AWS Service: codebuild). There are also buttons for 'Delete' and 'Create role'.

Select your role and go to the permissions tab and click “Add permissions > Create inline policy”

The screenshot shows the 'Permissions' tab for a role. It includes tabs for 'Permissions', 'Trust relationships', 'Tags', 'Access Advisor', and 'Revoke sessions'. Below these are sections for 'Permissions policies' (with one managed policy listed) and 'Permissions boundary - (not set)'. At the top right, there are buttons for 'Simulate', 'Remove', 'Add permissions' (which is highlighted with a red arrow), 'Attach policies', and 'Create inline policy'.

When you are creating the inline policy you should select Secrets Manager services and only allow Read access to it, then select Any in this account under resources.

## Create policy

1 2

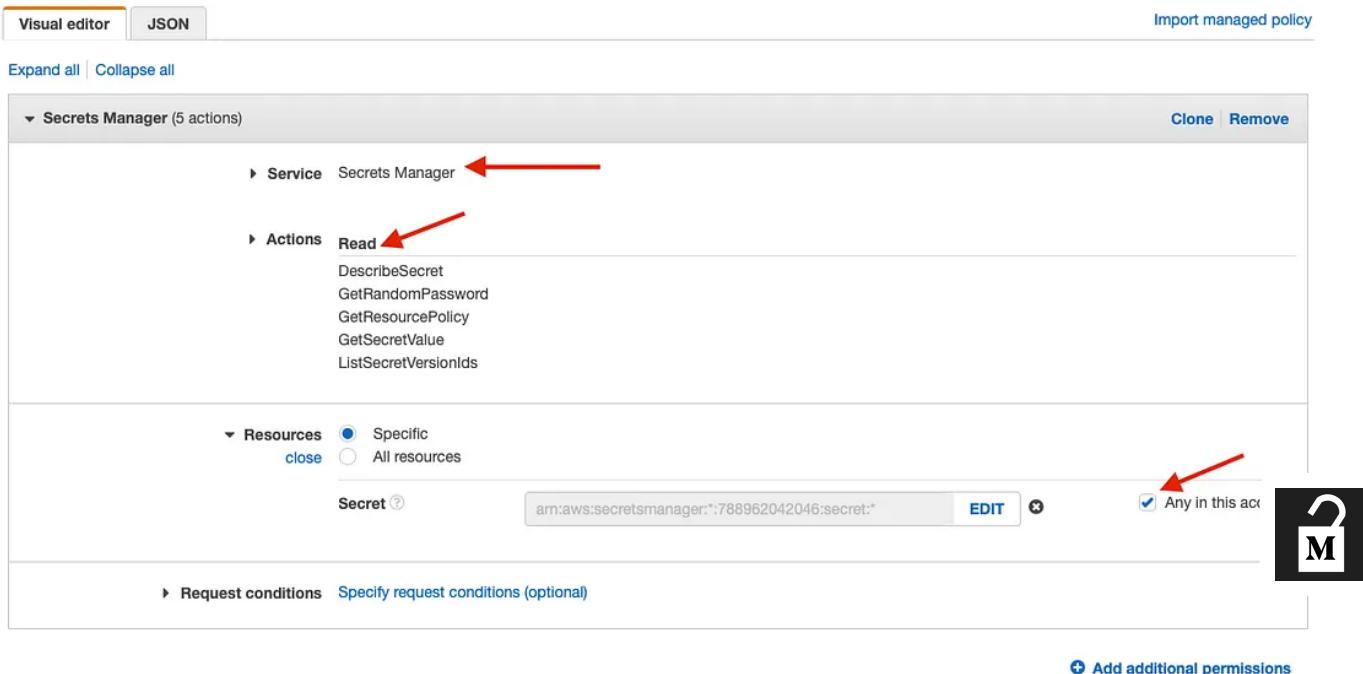
A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

[Visual editor](#) [JSON](#) [Import managed policy](#)

[Expand all](#) | [Collapse all](#)

<b>Secrets Manager (5 actions)</b>		<a href="#">Clone</a>   <a href="#">Remove</a>
<b>Service</b> Secrets Manager 		
<b>Actions</b> <b>Read</b> 		
DescribeSecret GetRandomPassword GetResourcePolicy GetSecretValue ListSecretVersionIds		
<b>Resources</b> <input checked="" type="radio"/> Specific <a href="#">close</a> <input type="radio"/> All resources		
Secret 		arn:aws:secretsmanager::788962042046:secret:  <span style="border: 1px solid #ccc; padding: 2px;">Edit</span>  <input checked="" type="checkbox"/> Any in this account 
<b>Request conditions</b> <a href="#">Specify request conditions (optional)</a>		

[+ Add additional permissions](#)



Pick a simple name so you can find it for other roles there need reading from Secrets Manager, I name it secrets-manager-read in my case.

## Review policy

Before you create this policy, provide the required information and review this policy.

Name\*  

Maximum 128 characters. Use alphanumeric and '+,-,.,@,\_' characters.

Summary			
Filter <input type="text" value="Filter"/>			
Service	Access level	Resource	Request condition
Allow (1 of 333 services) <a href="#">Show remaining 332</a>			
Secrets Manager	Full: Read	Multiple	None



When you finished and have added the inline policy it should be showing with the already existing role.

Permissions policies (2)		
You can attach up to 10 managed policies.		
<input type="button" value="Simulate"/> <input type="button" value="Remove"/> <input type="button" value="Add permissions"/>		
<input type="text"/> Filter policies by property or policy name and press enter		
Policy name	Type	Description
<input type="checkbox"/> <input type="button" value="CodeBuildBasePolicy-test-project-sync-repo-us-east-1"/>	Customer managed	Policy used in trust relationship
<input type="checkbox"/> <input type="button" value="secrets-manager-read"/>	Customer inline	

After you run the AWS CodeBuild again, you will see its success and everything should work as you expect when you trigger this build manually.

```

272 [Container] 2022/08/26 10:02:35 Phase complete: PRE_BUILD State: SUCCEEDED
273 [Container] 2022/08/26 10:02:35 Phase context status code: Message:
274 [Container] 2022/08/26 10:02:35 Entering phase BUILD
275 [Container] 2022/08/26 10:02:35 Running command cd /usr/bin/repo/datasink-cloud
276
277 [Container] 2022/08/26 10:02:35 Running command git remote add --mirror=fetch github https://$AWS_secretsmanager_get-secret-value --secret-id git/github | jq -r ".SecretString" | jq -r ".access_token"@github.com/pa...onalsdk/g.../datasink-cloud.git
278
279 [Container] 2022/08/26 10:02:37 Phase complete: BUILD State: SUCCEEDED
280 [Container] 2022/08/26 10:02:37 Phase context status code: Message:
281 [Container] 2022/08/26 10:02:37 Entering phase POST_BUILD
282 [Container] 2022/08/26 10:02:37 Running command git fetch origin
283
284 [Container] 2022/08/26 10:02:37 Running command git push github --all
285 Everything up-to-date
286
287 [Container] 2022/08/26 10:02:37 Phase complete: POST_BUILD State: SUCCEEDED
288 [Container] 2022/08/26 10:02:37 Phase context status code: Message:
289

```

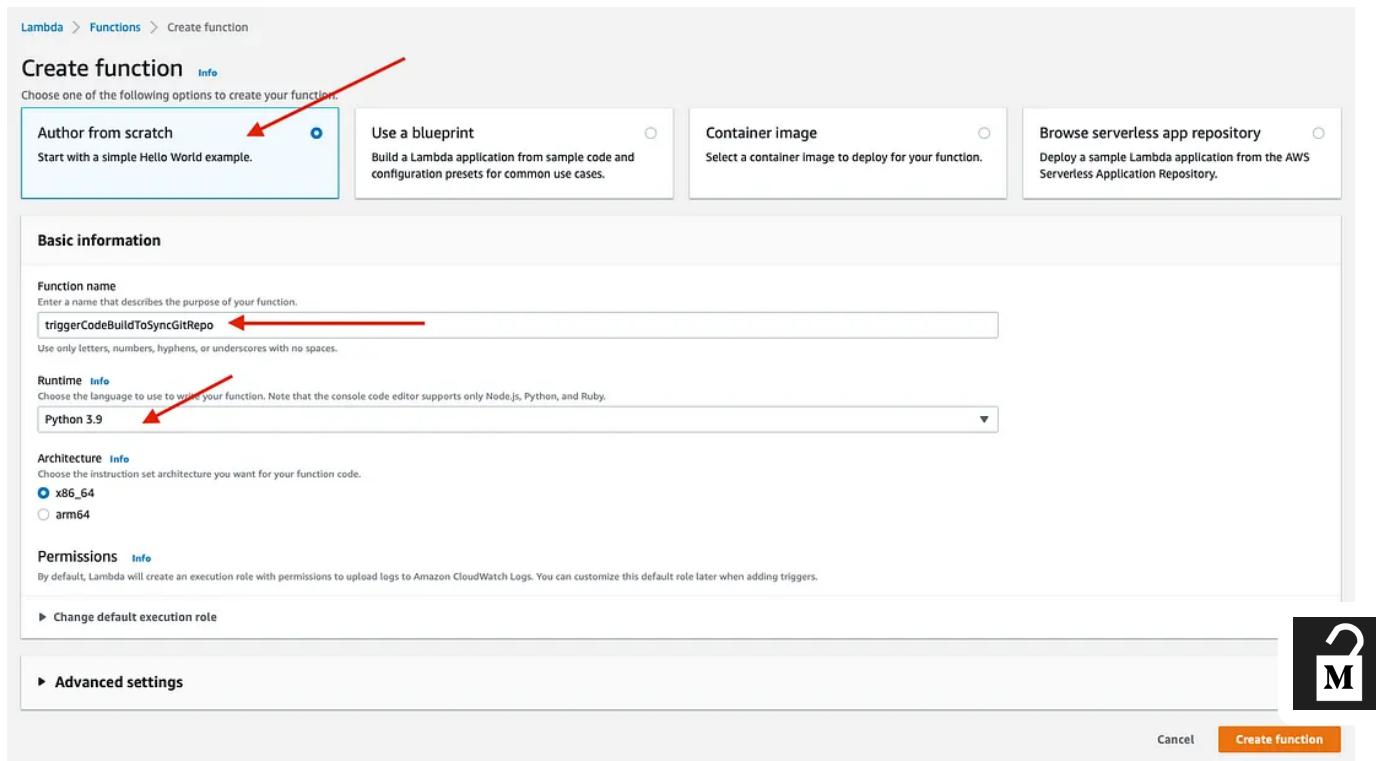


## Finished your automatic using Event Bridge and Lambda to sync automatically

What we will do in this section is create a trigger on every event from the AWS CodeCommit repository and send the event a Lambda function there will start our AWS CodeBuild process to sync between AWS CodeCommit and your GitHub repository.

It will be very simple, and we will don't focus on SNS and SQS in this area, I think it's something you should challenge yourself with after you know how these steps are working.

Start to create your Lambda function there can trigger our AWS CodeBuild from our trigger, so go to Lambda > Functions > Create function



When you create the function, we will create it from scratch and use the Python 3.9 runtime, I have picked a meaningful name so it's easier to find after.

```

import json
def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda!')
    }

```

Now you should end up with your Lambda function screen, now we need to allow this

function to start our AWS CodeBuild by going to Configuration > Permissions > Execution Role here we need to edit this Execution Role by clicking on the name

It's open in a new tab and we need to “Create inline policy” for this role

Select CodeBuild as a service and with actions you pick Write > StartBuild to allow this role to start builds and then allow all resources in this account.

Give it a meaningful name eg. codebuild-start-job so it's easy to reuse for other role policy.

### Review policy

Before you create this policy, provide the required information and review this policy.

The screenshot shows the AWS IAM 'Create New Policy' page. The 'Name\*' field contains 'codebuild-start-job' with a red arrow pointing to it. Below the field is a note: 'Maximum 128 characters. Use alphanumeric and '+,-,\_,@-' characters.' The policy document table has three columns: Service, Access level, Resource, and Request condition. Under 'Service', 'CodeBuild' is selected. Under 'Access level', 'Limited: Write' is chosen. Under 'Resource', the ARN is 'arn:aws:codebuild:::788962042046:project/'. The 'Request condition' column is empty.

Service	Access level	Resource	Request condition
CodeBuild	Limited: Write	arn:aws:codebuild:::788962042046:project/	None

Then you end up with 2 permissions policies for our execute role there are attached to our Lambda function

The screenshot shows the 'Permissions' tab for a Lambda function. It displays two attached policies: 'AWSLambdaBasicExecutionRole' (Customer managed) and 'codebuild-start-job' (Customer inline). The 'codebuild-start-job' policy is highlighted with a red box. Other tabs visible include 'Trust relationships', 'Tags', 'Access Advisor', and 'Revoke sessions'. A large 'Edit' button with a lock icon is prominent on the right.

Now go back to your Lambda function code editor and add the needed code to start the AWS function from GitHub trigger

[More from Paris Nakatakejser](https://parisnakatakejser.medium.com.sync-your-aws-codecommit-repos...)

```
v10.14.0
default -> 12.13.0 (-> v12.13.0)
iojs -> N/A (default)
unstable -> N/A (default)
node -> stable (-> v16.14.0) (default)
stable -> 16.14 (-> v16.14.0) (default)
lts/* -> lts/gallium (-> v16.14.0)
lts/argon -> v4.9.1 (-> N/A)
lts/boron -> v6.17.1 (-> N/A)
lts/carbon -> v8.17.0 (-> N/A)
lts/dubnium -> v10.24.1 (-> N/A)
lts/oxygum -> v12.22.10 (-> N/A)
```



Paris Nakita Kejser in DevOps Engineer, Software Architect and Software Developer



## How To Install NVM (Node Version Manager) on macOS with Homebrew

Node Version Manager (NVM) is used to develop NodeJS applications in multi versions, if you install NodeJS native on your system you will...

◆ · 2 min read · Mar 1, 2022

👏 287

💬 8

↗ +



# Monitoring Kubernetes

sitc



Paris Nakita Kejser in DevOps Engineer, Software Architect and Software Developer

**Lambda > Add trigger**

## Add trigger

### Trigger configuration

**CodeCommit** aws developer-tools git

**Repository name**  
Select the repository to add a trigger to.

**Trigger name**  
Provide a name for the trigger that will invoke this function.

**Events**  
Choose one or more events to listen for. If you choose "All repository events", you cannot choose other event types.

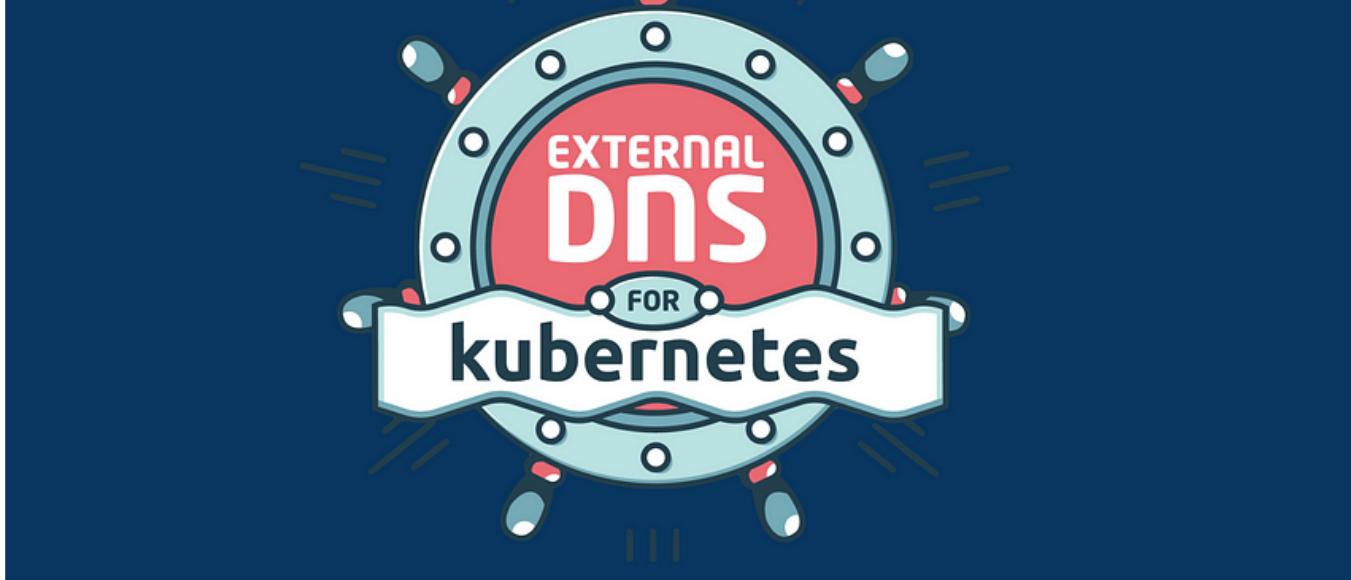
**Branch names**  
This trigger will be configured for all repository branches and tags by default. For a more specific configuration, choose up to 10 branches. If you choose "All branches", you cannot choose specific branches.

**Custom data - optional**  
Custom data is additional contextual information used to distinguish this trigger from other triggers that run for the same event, refer to external resources, or group triggers from different repositories. For example, you could include the channel ID # for a chat room used by your team to collaborate on development.

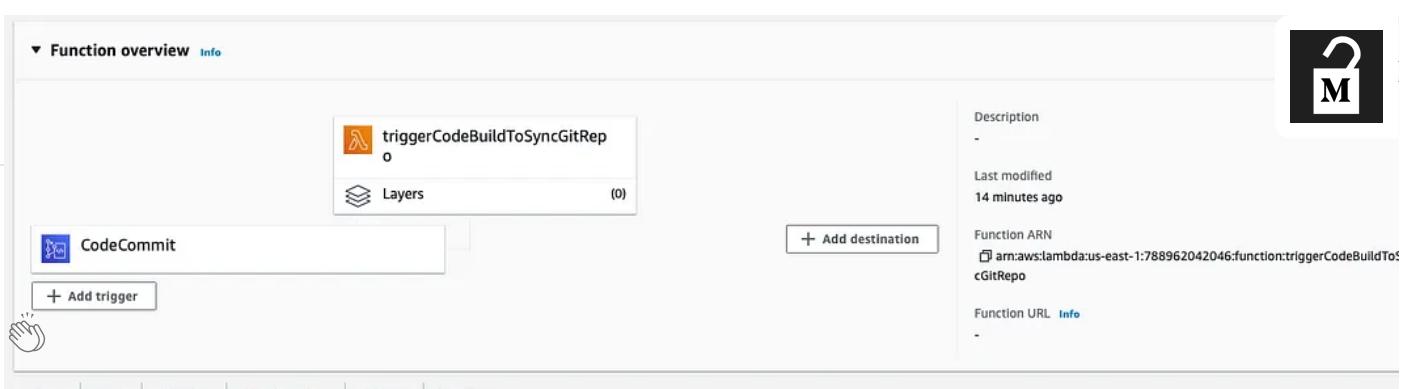
**Lambda will add the necessary permissions for AWS CodeCommit to invoke your Lambda function from this trigger.**  
[Learn more about the Lambda permissions model.](#)

Now you should have the trigger listed on your Lambda function Triggers page

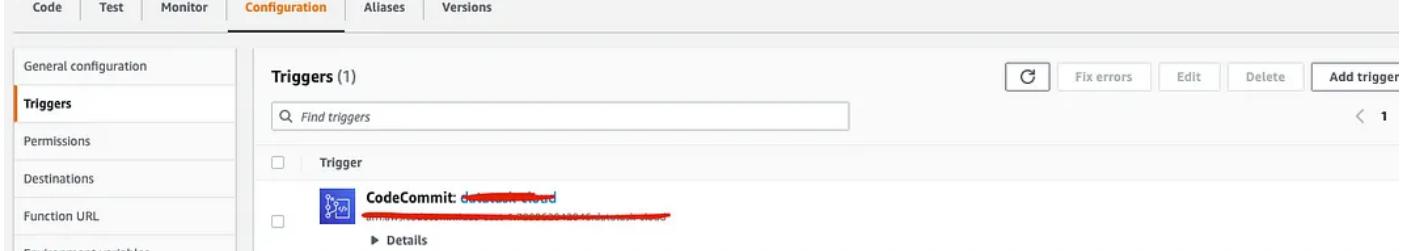




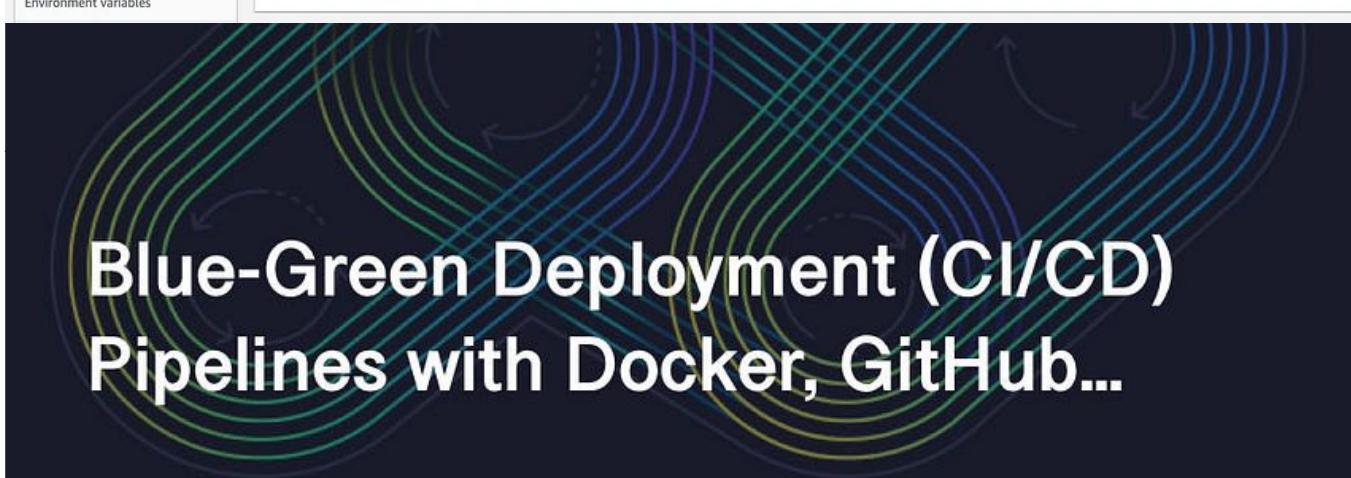
 Paris Nakita Kejser in DevOps Engineer, Software Architect and Software Developing



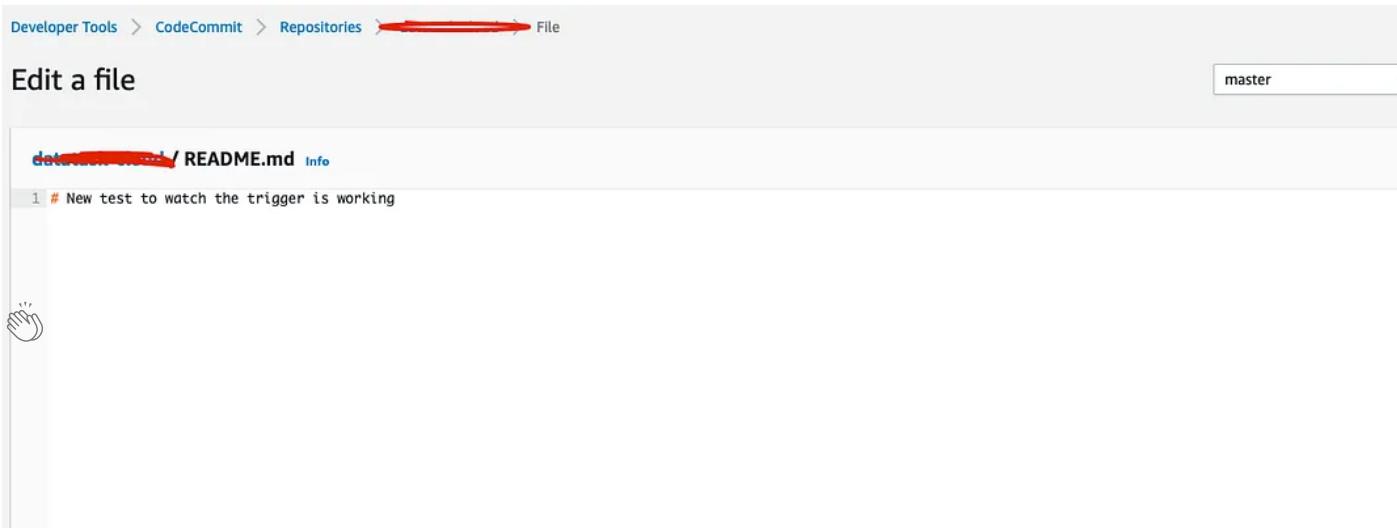
A screenshot of the AWS Lambda function configuration page. The function name is "triggerCodeBuildToSyncGitRep". It has no layers. It is triggered by "CodeCommit". There is a "Layers" section, a "Description" field (empty), and a "Last modified" field showing "14 minutes ago". The "Function ARN" is listed as "arn:aws:lambda:us-east-1:788962042046:function:triggerCodeBuildToSyncGitRepo". The "Function URL" is listed as "Function URL".



A screenshot of the AWS Lambda triggers configuration page. Under "General configuration", the "Triggers" tab is selected. It shows one trigger named "CodeCommit: [REDACTED]". The "Find triggers" search bar contains "[REDACTED]". There are buttons for "Edit", "Delete", and "Add trigger".



 Dmit in DevOps.dev



Developer Tools > CodeCommit > Repositories  File

Edit a file

master

data... / README.md 

```
1 # New test to watch the trigger is working
```



Commit changes to master

File: datatask-cloud/README.md

Author name 

Email address

Commit message - optional  
A default commit message will be used if you do not provide one.



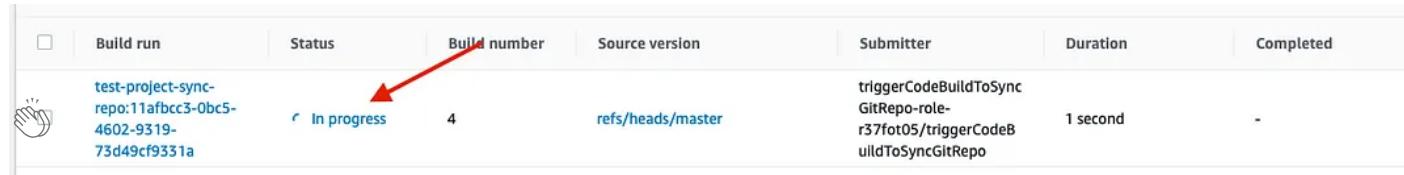
Cancel **Commit changes**

10.0.0.0/16

 **headintheclouds** in AWS Tip  
Fill out with the Author name, email, and a commit message after it clicks on “Commit changes” go to your AWS CodeBuild and your project then you can see the build status go in

**Deploy WordPress and MySQL with Terraform on AWS EC2 Like a Pro**

Prerequisites:



Build run	Status	Build number	Source version	Submitter	Duration	Completed
 test-project-sync-repo:11afbcc3-0bc5-4602-9319-73d49cf9331a	In progress	4	refs/heads/master	triggerCodeBuildToSyncGitRepo-role-r37f0t05/triggerCodeBuildToSyncGitRepo	1 second	-

## Lists



### Staff Picks

302 stories · 62 saves





CodeCircuit in CodeX



## 5 New DevOps Tools Expected to Make a Huge Impact in 2023

Enhance your DevOps collaboration with the latest and most innovative tools to shake up the industry in 2023.

◆ · 6 min read · Mar 25

192

7



```

commit ffcf2c01b7ef612893529cef188cc1961ed64521 (HEAD -> master, origin/master, origin/bors/staging, origin/HEAD)
Merge: fc991bf81 5159211da
Author: iohk-bors[bot] <43231472+iohk-bors[bot]@users.noreply.github.com>
Date:   Tue Nov 8 17:44:34 2022 +0000

Merge #4563

4563: New p2p topology file format r=coot a=coot

Fixes #4559.

Co-authored-by: Marcin Szamotulski <coot@coot.me>
Co-authored-by: olgahryniuk <67585499+olgahryniuk@users.noreply.github.com>

commit fc991bf814891a9349f22cf278632d39b04d4628
Merge: 5633d1c05 5cd94d372
Author: iohk-bors[bot] <43231472+iohk-bors[bot]@users.noreply.github.com>
Date:   Tue Nov 8 13:07:58 2022 +0000

Merge #4613

4613: Update building-the-node-using-nix.md r=CarlosLopezDeLara a=CarlosLopezDeLara

Build the cardano-node executable. No default configuration.

Co-authored-by: CarlosLopezDeLara <carlos.lopezdelara@iohk.io>

commit 5159211da7a644686a973e4fb316b64ebblaa34c
Author: olgahryniuk <67585499+olgahryniuk@users.noreply.github.com>
Date:   Tue Nov 8 13:25:10 2022 +0200

```



Jacob Bennett in Level Up Coding

## Use Git like a senior engineer

Git is a powerful tool that feels great to use when you know how to use it.

◆ · 4 min read · Nov 15, 2022

👏 4.6K

💬 51



George Baidoo Jr.

## CI/CD Pipeline: Deploy a Simple Application to an AWS EC2 Instance Via CodeDeploy

In this project I will be showing you how to deploy an application via a CI/CD Pipeline, to an AWS EC2 Instance via CodeDeploy & a AWS...

◆ · 8 min read · Dec 29, 2022

👏 16

💬





 headintheclouds in AWS Tip

## Full Stack DevOps Environment Setup Bash Script for Ubuntu, CentOS and MacOS

Introduction



◆ · 8 min read · Jan 25

 149

 6



See more recommendations