**<https://docs.aws.amazon.com/codedeploy/latest/userguide/deployments-cross-account.html>**

**https://medium.com/@davehagler/cross-account-aws-deployments-in-jenkins-8411595ae34e**

**Cross Account:**

[**https://aws.amazon.com/premiumsupport/knowledge-center/codepipeline-deploy-cloudformation/**](https://aws.amazon.com/premiumsupport/knowledge-center/codepipeline-deploy-cloudformation/)

**TOOLS ACCOUNT(Account A)::**

**ECR Policy:**

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "Pushorpull",

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam::Account\_B:root"

},

"Action": [

"ecr:BatchCheckLayerAvailability",

"ecr:BatchGetImage",

"ecr:CompleteLayerUpload",

"ecr:DescribeImages",

"ecr:DescribeRepositories",

"ecr:GetDownloadUrlForLayer",

"ecr:GetRepositoryPolicy",

"ecr:InitiateLayerUpload",

"ecr:ListImages",

"ecr:PutImage",

"ecr:UploadLayerPart"

]

}

]

}

**S3 Bucket Policy:**

{

"Version": "2012-10-17",

"Id": "Policy1553183091390",

"Statement": [

{

"Sid": "",

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam:: Account\_B:root"

},

"Action": [

"s3:Get\*",

"s3:Put\*"

],

"Resource": "arn:aws:s3:::pipelinedeployingsafeapi-apipipelineartifactsbuck-1lwkmjyjb10l1/\*"

},

{

"Sid": "",

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam:: Account\_B:root"

},

"Action": "s3:ListBucket",

"Resource": "arn:aws:s3:::pipelinedeployingsafeapi-apipipelineartifactsbuck-1lwkmjyjb10l1"

}

]

}

**Role for CodePipeline and CodeBuild:**

**Trust Relationship:**

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Principal": {

"Service": [

"codepipeline.amazonaws.com",

"codebuild.amazonaws.com"

]

},

"Action": "sts:AssumeRole"

}

]

}

**Add Policies for CodeBuild and CodePipeline:**

**Assume policy:**

{

"Version": "2012-10-17",

"Statement": {

"Effect": "Allow",

"Action": "sts:AssumeRole",

"Resource": [

"arn:aws:iam:: Account\_B:role/\*"

]

}

}

**NON-PROD Account(Account B):**

**Cross Account Role:**

**TrustRelationship: (while creating select other aws account)**

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Principal": {

"AWS": "arn:aws:iam::Account\_A\_ID:root"

},

"Action": "sts:AssumeRole",

"Condition": {}

}

]

}

**S3-crossAccount-policy:**

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"cloudformation:\*",

"iam:PassRole"

],

"Resource": "\*"

},

{

"Effect": "Allow",

"Action": [

"s3:Get\*",

"s3:Put\*",

"s3:ListBucket"

],

"Resource": [

"arn:aws:s3:::artifacts-pipeline" //Tools Account s3 bucket name

]

}

]

}

[**ecr-ca-policy**](https://console.aws.amazon.com/iam/home?region=us-west-2#/policies/arn%3Aaws%3Aiam%3A%3A901017866019%3Apolicy%2Fecr-ca-policy)**:**

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"ecr:\*",

"cloudtrail:LookupEvents"

],

"Resource": "arn:aws:ecr:us-west-2:258102705540:repository/test-ecs-demo"

}

]

}

[**kms-cross-account-policy**](https://console.aws.amazon.com/iam/home?region=us-west-2#/policies/arn%3Aaws%3Aiam%3A%3A901017866019%3Apolicy%2Fkms-cross-account-policy)**:**

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"kms:DescribeKey",

"kms:GenerateDataKey\*",

"kms:Encrypt",

"kms:ReEncrypt\*",

"kms:Decrypt"

],

"Resource": [

"arn:aws:kms:us-west-2:258102705540:key/ea8154b1-d885-42cd-b5b9-2403575b58ed"

]

}

]

}

**Cloudformation role:**

**Trust Relation: (Cloudformation)**

[AWSCloudFormationFullAccess](https://console.aws.amazon.com/iam/home?region=us-west-2#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAWSCloudFormationFullAccess)

# How do I use CodePipeline to deploy an AWS CloudFormation stack in a different account?

*Last updated: 2019-12-09*

How can I use AWS CodePipeline to deploy an AWS CloudFormation stack in a different account?

## Short Description

To deploy an AWS CloudFormation stack in a different account, you must complete the following:

1. Create a pipeline in one account, account A. This account should include a customer managed AWS Key Management Service (AWS KMS) key, an Amazon Simple Storage Service (Amazon S3) bucket for artifacts, and an S3 bucket policy that allows access from the other account, account B.
2. In account B, configure a cross-account service role that allows the following: 1) AWS CloudFormation actions, 2) access to the S3 bucket in account A, and 3) decryption with the customer managed KMS key in account A.
3. In account A, allow a pipeline service role to assume a cross-account role (with [AssumeRole](https://docs.aws.amazon.com/STS/latest/APIReference/API_AssumeRole.html" \t "_blank)) in account B.

## Resolution

In account A, create a customer managed key

1.    In account A, open the [AWS KMS console](https://console.aws.amazon.com/kms/).

2.    In the navigation pane, choose Customer managed keys.

3.    Choose Create key, and then choose Symmetric.

Note: In the Advanced options section, leave the origin as KMS.

4.    For Alias, enter a name for your key.

5.    (Optional) Add tags, and then choose Next.

6.    On the Define key administrative permissions page, for Key administrators, choose your AWS Identity and Access Management (IAM) user and any other users or groups that you want to serve as administrators for the key, and then choose Next.

7.    On the Define key usage permissions page, for This account, add the IAM users in the account that should have access to the key (such as the CodePipeline service role).

8.    In the Other AWS accounts section, choose Add another AWS account, and then enter the Amazon Resource Name (ARN) of the IAM role in account B.

9.    Choose Next, and then choose Finish.

10.    In the Customer managed keys section, choose the key that you just created, and then copy the ARN for that key.

Note: You must have the ARN for your key when you edit your pipeline and configure your policies.

In account A, select or create an Amazon S3 bucket as the pipeline of the SourceArtifact bucket, and then create a bucket policy

Your S3 bucket policy in account A must allow access for account B.

1.    In account A, open the [Amazon S3 console](https://console.aws.amazon.com/s3/).

2.    Choose an existing S3 bucket or create a new S3 bucket to use as the [ArtifactStore](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-codepipeline-pipeline-artifactstore.html" \t "_blank) for CodePipeline.

Note: Artifacts can include a [stack template file](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/continuous-delivery-codepipeline-cfn-artifacts.html#w2ab1c13c17c13), a [template configuration file](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/continuous-delivery-codepipeline-cfn-artifacts.html#w2ab1c13c17c15), or both. CodePipeline uses these artifacts to work with AWS CloudFormation stacks and change sets. In your template configuration file, you must specify template parameter values, a stack policy, and tags.

3.    On the Amazon S3 details page for your bucket, choose Permissions.

4.    Choose Bucket Policy.

5.    In the Bucket policy editor, enter the following code:

{

"Id": "Policy1553183091390",

"Version": "2012-10-17",

"Statement": [{

"Sid": "",

"Action": [

"s3:Get\*",

"s3:Put\*"

],

"Effect": "Allow",

"Resource": "arn:aws:s3:::codepipeline-source-artifact/\*",

"Principal": {

"AWS": [

"arn:aws:iam::ACCOUNT\_B\_NO:root"

]

}

},

{

"Sid": "",

"Action": [

"s3:ListBucket"

],

"Effect": "Allow",

"Resource": "arn:aws:s3:::codepipeline-source-artifact",

"Principal": {

"AWS": [

"arn:aws:iam::ACCOUNT\_B\_NO:root"

]

}

}

]

}

6.    In the preceding code, replace codepipeline-source-artifact with the SourceArtifact bucket name for CodePipeline.

7.    In the preceding code, replace ACCOUNT\_B\_NO with the account number of account B.

8.    Choose Save.

In account B, create a cross-account role (CROSS\_ACCOUNT\_ROLE) that allows actions related to S3 and KMS in the SourceArtifact for account A

The cross-account role policy allows the pipeline in account A to assume a role in account B. This policy also enables AWS CloudFormation actions, and access to operations related to SourceArtifact and KMS.

1.    In account B, open the [IAM console](https://console.aws.amazon.com/iam/).

2.    In the navigation pane, choose Policies.

3.    Choose Create policy.

4.    Choose the JSON view, and then enter the following policy into the code editor:

{

"Version": "2012-10-17",

"Statement": [{

"Effect": "Allow",

"Action": [

"cloudformation:\*",

"iam:PassRole"

],

"Resource": "\*"

},

{

"Effect": "Allow",

"Action": [

"s3:Get\*",

"s3:Put\*",

"s3:ListBucket"

],

"Resource": [

"arn:aws:s3:::codepipeline-source-artifact/\*"

]

}

]

}

5.    In the preceding code, replace codepipeline-source-artifact with the ArtifactStore bucket name of the pipeline.

6.    Choose Review policy, and then create the policy.

7.    In the navigation pane, choose Policies.

8.    Choose Create policy.

9.    Choose the JSON view, and then enter the following policy into the code editor:

{

"Version": "2012-10-17",

"Statement": [{

"Effect": "Allow",

"Action": [

"kms:DescribeKey",

"kms:GenerateDataKey\*",

"kms:Encrypt",

"kms:ReEncrypt\*",

"kms:Decrypt"

],

"Resource": [

"arn:aws:kms:REGION:ACCOUNT\_A\_NO:key/key-id"

]

}]

}

10.    In the preceding code, replace arn:aws:kms:REGION:ACCOUNT\_A\_NO:key/key-id with the ARN of the key that you created in the Create a customer managed key in account A section earlier.

11.    Choose Review policy, and then create the policy.

12.    In the navigation pane, choose Roles.

13.    Choose Create role.

14.    Choose Another AWS account.

15.    For Account ID, enter the account ID of account A.

16.    Choose Next: Permissions, and then complete the steps to create the role.

17.    Attach the cross-account role policy and KMS key policy to the role that you created.

In account A, add the AssumeRole permission to the CodePipeline service role

Your policy allows the CodePipeline service role in account A to assume the cross-account role that you previously created in account B.

1.    In account A, open the [IAM console](https://console.aws.amazon.com/iam/).

2.    In the navigation pane, choose Roles.

3.    Choose the IAM service role that you're using for CodePipeline.

4.    Choose Add inline policy.

5.    Choose the JSON view, and then enter the following policy into the code editor:

{

"Version": "2012-10-17",

"Statement": {

"Effect": "Allow",

"Action": "sts:AssumeRole",

"Resource": [

"arn:aws:iam::ACCOUNT\_B\_NO:role/\*"

]

}

}

6.    In the preceding code, replace ACCOUNT\_B\_NO with the number of account B.

7.    Choose Review policy, and then create the policy.

In account B, create a role for the AWS CloudFormation stack (CFN\_STACK\_ROLE)

This [service role](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-iam-servicerole.html) is directly configured on the AWS CloudFormation stack in account B, and must include the permissions for the services deployed by the stack.

1.    In account B, open the [IAM console](https://console.aws.amazon.com/iam/).

2.    In the navigation pane, choose Roles.

3.    Create a role for AWS CloudFormation to use when launching services on your behalf.

4.    Apply [permissions](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-iam-servicerole.html) to your role based on your needs.

Important: Be sure that your trust policy is for AWS CloudFormation, and that your role has permissions to access services deployed by the stack.

In account A, update the CodePipeline configuration

Note: You can't use the CodePipeline console to create or edit a pipeline that uses resources associated with another account. However, you can use the console to create the general structure of the pipeline. Then, you can use the AWS Command Line Interface (AWS CLI) to edit the pipeline and add the resources associated with the other account. Or, you can update a current pipeline with the resources for the new pipeline.

1.    In the AWS CLI, run the following command to get the pipeline JSON structure:

aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json

2.    In your local pipeline.json file, confirm that the encryptionKey ID contains the ID with the ARN of the key that you created earlier.

3.    In the pipeline.json file, update the AWS CloudFormation [action configuration](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/continuous-delivery-codepipeline-action-reference.html).

Note: The RoleArn inside the action configuration JSON structure for your pipeline is the role for the AWS CloudFormation stack (CFN\_STACK\_ROLE). The roleArn outside the action configuration JSON structure is the cross-account role that the pipeline assumes to operate an AWS CloudFormation stack (CROSS\_ACCOUNT\_ROLE).

4.    Verify that the role is updated for both RoleArn and roleArn.

In the following code example, RoleArn is the role passed to AWS CloudFormation to launch the stack. CodePipeline uses roleArn to operate an AWS CloudFormation stack.

{

"name": "Prod\_Deploy",

"actions": [{

"inputArtifacts": [{

"name": "MyApp"

}],

"name": "test-cfn-x",

"actionTypeId": {

"category": "Deploy",

"owner": "AWS",

"version": "1",

"provider": "CloudFormation"

},

"outputArtifacts": [],

"configuration": {

"ActionMode": "CHANGE\_SET\_REPLACE",

"ChangeSetName": "test",

"RoleArn": "ARN\_FOR\_CFN\_STACK\_ROLE",

"Capabilities": "CAPABILITY\_IAM",

"StackName": "test-cfn-sam",

"TemplatePath": "MyApp::template.yaml"

},

"roleArn": "ARN\_FOR\_CROSS\_ACCOUNT\_ROLE",

"runOrder": 1

}]

}

5.    Remove the metadata configuration from the pipeline.json file:

"metadata": {

"pipelineArn": "arn:aws:codepipeline:REGION:ACC:my\_test",

"updated": 1551216777.183,

"created": 1551207202.964

}

Important: To align with proper JSON formatting, remove the comma before the metadata section.

6.    (Optional) To create a pipeline and update the JSON structure, run the following command to update the pipeline with the new configuration file:

aws codepipeline update-pipeline --cli-input-json file://pipeline.json

7.    (Optional) To use a current pipeline and update the JSON structure, run the following command to create a new pipeline:

aws codepipeline create-pipeline --cli-input-json file://pipeline.json

Note: In your pipeline.json file, change the name of your new pipeline.