Multiple-account, multiple-Region AWS CloudFormation using nested Stack Deployment

Prepared by:

Nurunnubi Talukder, Solutions Architect

David Fei, Senior DevOps Engineer

1. Create IAM role for Central account and Target Account (DEV, UAT and PROD) using CloudFormation template

## **central-iam.yaml**

## **dev-iam.yaml**

prod-yaml.yaml

uat-yaml.yaml

1. Deployment IAM role cloudformation template in respected Account in Tools, Dev, UAT, PROD.
2. Create bucket policy in Tools account Bucket and access to DEV, UAT and PROD.

{

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

"Statement": [

{

"Effect": "Allow",

"Principal": {

"AWS": [

~~"arn:aws:iam::062823937524:root",~~

~~"arn:aws:iam::828852556902:root",~~

~~"arn:aws:iam::851573698321:root"~~

"arn:aws:iam::062823937524: role/ProdAccessRole",

"arn:aws:iam::828852556902: role/DevAccessRole",

"arn:aws:iam::851573698321: role/UatAccessRole"

]

},

"Action": [

"s3:GetObject",

"s3:PutObject",

"s3:PutObjectAcl"

],

"Resource": "arn:aws:s3:::duobank-testbucket1/\*"

}

]

}

~~Note: Root means, it will share to all the users under that account. You can also share to a specific role or user.~~

Note: The lambda backed custom resource will launch lambda function under CentralControlRole which will assume to ProdAccessRole/DevAccessRole/UatAccessRole. Those role will create cloudformation stacks in Prod/Dev/Uat accounts and Those roles will access the S3 bucket from Tools account. So The bucket policy should grant S3 permissions to those roles.

If you want to share S3 bucket to all users/roles in those account, your can use arn:aws:iam::<account\_num>:root. But it is nice to only share to some certain roles for the security perspective.

Graphical user interface, text, application, email

Description automatically generated

1. Create bucket and create the folder accordingly in Tools Account:

Graphical user interface, text, application, email

Description automatically generated

1. Create the cross-account-helper folder and update the python package zip file to the folder.

<https://github.com/aws-quickstart/quickstart-examples/tree/main/samples/cloudformation-cross-account/functions/packages/CfnStackAssumeRole>

The cross-account helper lambda function will extend the cloudformation function to create stack in another account. That’s why we are using custom resource.

Graphical user interface, text, application, email

Description automatically generated

1. Create folder lambda\_source and update lambda code

Graphical user interface, application

Description automatically generated

1. Create folder and update all CFN template except the cross-account-dev.yaml

Graphical user interface, application

Description automatically generated

Please note that we added all parameters from main\_dev\_parameters.json to cross-account-dev.yaml to support custom resource. ~~No need seprate file for customer resources~~ The current solution doesn’t support separate json parameter file. All the parameters has to be in the cross-accound-dev.yaml.

1. Deploy the sources from Tools account to Dev Account using the below command:

aws cloudformation create-stack --stack-name duobank-iam-stack --template-body file://cross-account-Dev.yaml --region ca-central-1 --capabilities CAPABILITY\_NAMED\_IAM

1. Check nested stack from Tools (Central Account) and check all the resources in Dev Account

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Reference:

<https://aws.amazon.com/blogs/infrastructure-and-automation/multiple-account-multiple-region-aws-cloudformation/>