Multi-Region Application Architecture Solution

Implementation Guide

Multi-Region Application Architecture Solution: Implementation Guide

Table of Contents

Deploy a reference architecture that models a serverless active/passive workload with asynchronous replication of application data and failover from a primary to a secondary AWS Region

AWS Implementation Guide

AWS Solutions Builder Team

June 2020 (last update (p. 20): January 2021)

This implementation guide discusses architectural considerations and configuration steps for deploying the Multi-Region Application Architecture solution in the Amazon Web Services (AWS) Cloud. It includes links to an AWS CloudFormation template that launches and configures the AWS services required to deploy this solution using AWS best practices for security and availability.

The guide is intended for IT infrastructure architects, administrators, and DevOps professionals who have practical experience architecting in the AWS Cloud.

Overview

Many Amazon Web Services (AWS) customers have business requirements that require expedient recovery from Regional failure with little-to-no application data loss. To help demonstrate an active/passive serverless architecture with easy failover to a backup Region, AWS offers the Multi-Region Application Architecture solution.

This solution leverages Amazon Simple Storage Service (Amazon S3) Cross-Region replication and Amazon DynamoDB Global Tables to asynchronously replicate application data between the primary and secondary AWS Regions. A sample photo-sharing web application can be deployed after the solution completes its deployment. The web application serves as a visual demonstration of the solution's back-end layers and verifies that Regional failover is working.

An application's Recovery Time Objective (RTO) and Recovery Point Objective (RPO) are important metrics when considering failover and disaster recovery scenarios. This solution allows for an RTO of a few seconds and an RPO of 15 minutes for application data stored in Amazon S3 and Amazon DynamoDB.

Cost

You are responsible for the cost of the AWS services used while running this solution. The total cost for running this solution depends on the number of users you create, the number of images you store and how many times they are accessed from the application.

The cost for running this solution with default settings in the primary US East (N. Virginia) Region and Asia Pacific (Tokyo) as the secondary Region is approximately **$10.00 per month**. This estimate includes considerations for photos stored in Amazon S3 (totaling 1GB in size) and comments stored in Amazon DynamoDB replicated to the secondary Region.

Prices are subject to change. For full details, see the pricing webpage for each AWS service you will be using in this solution.

Architecture overview

Deploying this solution builds the following environment in the AWS Cloud.

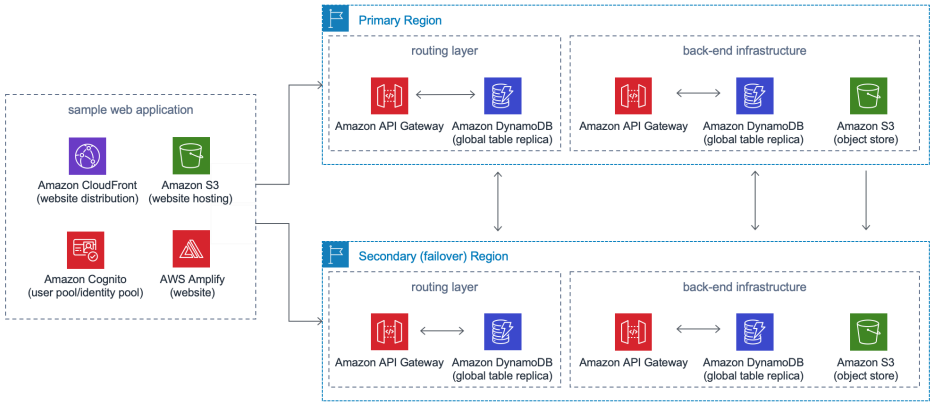

Figure 1: Multi-Region Application Architecture on AWS architecture

The AWS CloudFormation template uses AWS CloudFormation StackSets to deploy the routing layer and back-end infrastructure in both the primary and secondary (failover) AWS Regions. An optional second AWS CloudFormation template can be deployed after the solution's main template has completed deployment. This second template deploys an AWS Amplify sample web application hosted in an Amazon Simple Storage Service (Amazon S3) bucket, an Amazon CloudFront distribution to deliver the web application to users, and an Amazon Cognito user pool and identity pool to enable users to access the web application, the routing layer, and the back-end infrastructure resources.

After the web application loads, it queries the solution's routing layer for the current state of the application (`active`, `fenced`, `failover`), and configures AWS Amplify to target the solution's resources in the correct Region. The state of the application is also retrieved when the user uploads a new photo or adds a comment. Depending on the state of the application, a message may be displayed to indicate whether or not certain actions are available, or if the application must be refreshed. For more information, refer to the section called "Application states" (p. 4).

Solution components

Application states

The Multi-Region Application Architecture solution provides three potential application states. For information on changing the state, refer to *Appendix A: Changing the application state* (p. 13).

Active State: This state indicates the application is operating as normal. When the web application is loaded, it will target back-end resources in the primary Region. Photo uploads and comments are enabled. Uploaded photos are automatically replicated asynchronously to the secondary Region's object store S3 bucket using Amazon Simple Storage Service (Amazon S3) Cross-Region replication.

Failover State: This state indicates the primary Region is unavailable. When the web application is loaded, it will target back-end resources in the secondary Region. Photo uploads and comments will remain available, but photos uploaded while the application is in a failover state will not be replicated to the primary Region's object store S3 bucket.

Fenced State: This state indicates that the application is in a read-only mode. When the web application is loaded, it will target back-end resources in the primary Region. You cannot upload new photos or post new comments but existing photos and comments are available to view.

Routing layer

The routing layer deploys an Amazon API Gateway with proxy integration to an Amazon DynamoDB global table the primary and secondary Regions. The DynamoDB global table will store the application's state. When the front-end layer's web application is loaded, it requests the application's state from the API Gateway in the primary Region. If the primary Region is unavailable, the web application will query the API Gateway in the secondary Region. When the current state is received, the application will load the necessary configuration settings in AWS Amplify, and point to resources in the primary or secondary Region. Access to the routing layer's API methods is secured with AWS Identity and Access Management (IAM). The sample web application deploys an Amazon Cognito identity pool and grants the authenticated user role permission to invoke the API.

Back-end infrastructure

Object store

An Amazon S3 bucket is deployed in both the primary and secondary Regions. Amazon S3 Cross-Region replication is configured to asynchronously replicate objects from the primary Region's object store S3 bucket to the secondary Region's object store S3 bucket.

The sample web application stores photos uploaded by users in the object store S3 bucket. When the sample web application is in an active state and configured to use resources in the primary Region, photos uploaded by users are stored in the primary Region's object store S3 bucket. They are then asynchronously replicated to the secondary Region

> **Note**
> This solution demonstrates an active/passive architecture that allows for failover to a backup Region. Objects are replicated from the primary Region to the secondary Region only.

When the application is in a **failover** state and a photo is uploaded by a user, that photo will be stored in the secondary Region's object store S3 bucket and will not be replicated to the primary Region.

API Gateway and DynamoDB global table

This solution deploys an Amazon API Gateway with proxy integration to an Amazon DynamoDB global table in the primary and secondary Regions. Access to the API Gateway is controlled with IAM.

The sample web application uses the API Gateway and DynamoDB to store and retrieve comments made on uploaded photos. When a photo is selected in the sample application, a `GET` request is created to retrieve the comments for the selected photo. When comments are added in the sample application, a `POST` request is made to store the comment.

Sample web application

This solution provides an optional sample web application that is created if you choose to deploy the `multi-region-application-architecture-demo-ui` AWS CloudFormation template. This application creates a React with AWS Amplify sample photo-sharing web application that can be used as a visual demonstration of this solution's back-end infrastructure, which verifies that regional failover is working. This web application is deployed as a static website hosted in an Amazon S3 bucket. To help reduce latency and improve security, this solution includes an Amazon CloudFront distribution with an origin access identity. This is a special CloudFront user that helps restrict access to the website bucket contents.

This web application supports user sign-in, image uploads and sharing, and commenting on images. Amazon Cognito authenticates users to enable access to invoke the solution's APIs and authorize uploading/viewing images stored in the solution's object store S3 bucket. Images are uploaded to the Amazon S3 bucket in the primary Region and replicated to the secondary Region using Amazon S3 Cross-Region replication. Photo comments are stored in an Amazon DynamoDB global table. New users may only be added to the user pool using the Amazon Cognito console. New user sign-ups from the web application are not allowed.

Note
This solution doesn't restrict access to photos using Amazon Cognito. All uploaded photos will be visible to all users of the application.

For more information on deploying the sample web application, refer to Appendix C (p. 15).

Design considerations

AWS CloudFormation StackSets

This solution uses AWS CloudFormation StackSets to deploy the routing layer and back-end infrastructure in both the primary and secondary AWS Regions. When deploying this solution, an additional CloudFormation stack launches in each Region. These additional stacks are the StackSets instances that are managed by this solution. The deployment creates the `AWSCloudFormationStackSetExecutionRole`, which is assumed by CloudFormation and contains the necessary permissions to manage the solution's resources created by the AWS CloudFormation StackSets.

Cross-Region replication

This solution leverages Amazon Simple Storage Service (Amazon S3) Cross-Region replication to asynchronously replicate photos uploaded in the Amazon S3 object store bucket in the primary Region to the Amazon S3 object store bucket in the secondary Region.

Note
Photo replication is currently one-way so objects uploaded in the secondary Region's S3 bucket is not replicated to the primary Region's S3 bucket. When the application is in a *failover* state and a photo is uploaded by a user, that photo will be stored in the secondary Region's object store S3 bucket and will not be replicated to the primary Region.

Amazon DynamoDB global tables

This solution leverages Amazon DynamoDB global tables to asynchronously replicate the state of the application used in the routing layer and the photo comments in the back-end infrastructure. The `CreateGlobalTable` API is called to create a replication relationship between the DynamoDB tables in each Region.

RTO/RPO

This solution allows for a Recovery Time Objective (RTO) of a few seconds and a Recovery Point Objective (RPO) of 15 minutes. Profiles for new confirmed users in the primary Region's user pool are replicated to the secondary Region every five minutes. Amazon S3 Cross-Region replication is leveraged to replicate the majority of objects uploaded in 15 minutes. Added comments are stored in a DynamoDB global table. In the global table, a newly written item is propagated to all replica tables within seconds. For more information about RTO and RPO, refer to Disaster Recovery Objectives in the Reliability Pillar of the *AWS Well-Architected Framework*.

Failover from the primary Region to the secondary Region requires updating the application's state, which is stored in an item in a DynamoDB global table. Once the application's state has been updated, the web application will target resources in the other Region as soon as the page is refreshed. For more information about changing the application state, refer to Appendix A (p. 13).

Solution updates

If you have previously deployed this solution, you need to uninstall the previous version first before installing the latest version. For uninstall instructions, refer to Appendix C (p. 15).

AWS CloudFormation templates

This solution uses AWS CloudFormation to automate the deployment of the Multi-Region Application Architecture solution in the AWS Cloud. It includes the following AWS CloudFormation template, which you can download before deployment:

multi-region-application-architecture.template: Use this template to launch the solution and all associated components. The default configuration deploys Amazon Simple Storage Service (Amazon S3) buckets, Amazon DynamoDB tables, AWS Lambda functions, Amazon API Gateway, AWS Identity and Access Management (IAM) roles and policies, an Amazon CloudFront distribution, Amazon Cognito user pools, and AWS CloudFormation StackSets, but you can also customize the template based on your specific needs.

multi-region-application-architecture-demo-ui.template: Use this optional template to launch the sample web application and all associated components. The default configuration deploys an Amazon S3 bucket to store the sample application's source files, IAM roles and policies, an Amazon CloudFront distribution, and an Amazon Cognito user pool and identity pool, but you can also customize the template based on your specific needs.

Automated deployment

Before you launch the automated deployment, review the architecture, configuration, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy Multi-Region Application Architecture into your account.

If you have previously deployed this solution, uninstall the previous version first before installing the latest version. For uninstall instructions, refer to Appendix C (p. 15).

Time to deploy: Approximately 5 minutes

Launch the stack

This automated AWS CloudFormation template deploys the Multi-Region Application Architecture solution.

> **Note**
> You are responsible for the cost of the AWS services used while running this solution. Refer to the Cost (p. 2) section for more details. For full details, refer to the pricing webpage for each AWS service you will be using in this solution.

1. Sign in to the AWS Management Console and use the button below to launch the `multi-region-application-architecture` AWS CloudFormation template.

   Launch Solution

   You can also download the template as a starting point for your own implementation.
2. The template is launched in the US East (N. Virginia) Region by default. To launch this solution in a different AWS Region, use the Region selector in the console navigation bar.
3. On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**.
4. On the **Specify stack details** page, assign a name to your solution stack.
5. Under **Parameters**, review the parameters for the template, and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
Secondary Region	*<Requires Input>*	The secondary AWS Region that will be used in the event of an application failover. **Note** The primary Region will default to the Region where the stack is deployed.

6. Choose **Next**.

7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should receive a **CREATE_COMPLETE** status in approximately five minutes.

   Note
   This solution includes multiple AWS CloudFormation `custom-resource` AWS Lambda functions, which run only during initial configuration or when resources are updated or deleted.
   When running this solution, these functions are inactive. However, do not delete these functions as they are necessary to manage associated resources.

Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model can reduce your operational burden as AWS operates, manages, and controls the components from the host operating system and virtualization layer down to the physical security of the facilities in which the services operate. For more information about security on AWS, visit the AWS Security Center.

IAM roles

AWS Identity and Access Management (IAM) roles enable customers to assign granular access policies and permissions to services and users on the AWS Cloud. This solution creates IAM roles that grants the AWS Lambda functions access to the other AWS services used in this solution.

Amazon CloudFront

This solution deploys a static website hosted in an Amazon S3 bucket. To help reduce latency and improve security, this solution includes an Amazon CloudFront distribution with an origin access identity, which is a special CloudFront user that helps restrict access to the solution's website bucket contents. For more information, see Restricting Access to Amazon S3 Content by Using an Origin Access Identity.

Amazon API Gateway

IAM controls the access to invoke this solution's Amazon API Gateway. The template for the sample application deploys an Amazon Cognito identity pool with an authorized role that allows access to the APIs.

This solution's API Gateway is configured to allow all origins (an asterisk '*' is the value for the `Access-Control-Allow-Origin` header).

Photo-sharing demo application

When a user has signed into the photo-sharing application, they will have access to view all photos that have been added by all users. The application does not allow for private photo uploads or for only sharing photos with specific users. Currently, fine-grained access to Amazon S3 objects using Amazon Cognito is not configured for this demo application.

Additional resources

- AWS Lambda
- Amazon Simple Storage Service
- Amazon CloudWatch
- Amazon DynamoDB
- Amazon API Gateway
- AWS CloudFormation
- AWS Identity and Access Management
- Amazon CloudFront
- AWS Amplify

Appendix A: Changing the application state

The Multi-Region Application Architecture solution's application state is stored in the Amazon DynamoDB `AppConfigTable` table, created by the the section called "Routing layer" (p. 4). When the solution is deployed, an `AppId` for the sample photo-sharing application is generated by an AWS Lambda-backed Custom Resource and is used as the key for the corresponding item in the `AppConfigTable`. Updating the `state` attribute of this item in the `AppConfigTable` will change the application's state. Accepted values are `active`, `fenced`, and `failover`. Attributes are case sensitive.

As an alternative to manually editing the item in DynamoDB, the routing layer will also deploy a Lambda `AppStateUpdaterLambda` function. To change the state of the application using this function, invoke it with an event payload similar to the below. The values for `SampleApplicationAppId` and `AppConfigGlobalTableName` can be found in the **Outputs** tab of the solution's main CloudFormation stack.

```
{
  "AppId": "SampleApplicationAppId",
  "TableName": "AppConfigGlobalTableName",
  "NewState": "active/fenced/failover"
}
```

Appendix B: Deploying the sample web application

To deploy the sample web application, launch the optional CloudFormation template after the solution's main stack has finished deploying.

Sign in to the AWS Management Console and use the button below to launch the `multi-region-application-architecture-demo-ui` AWS CloudFormation template. The template is launched in the US East (N. Virginia) Region by default. To launch the solution in a different AWS Region, use the region selector in the console navigation bar. The sample web application must be deployed in the Primary region (the Region in which you deployed the solution's main stack).

You can also download the template as a starting point for your own implementation.

Under **Parameters**, review the parameters for the template and modify them as necessary. This solution uses the following default values.

Parameter	Default	Description
User Name	*<Requires Input>*	Create a user name for the Cognito user that will be created.
Email Address	*<Requires Input>*	Enter an email address for the Cognito user that will be created.
Solution Stack Name	*<Requires Input>*	Enter the name of the main stack used to deploy the solution's resources.

After the sample web application is deployed, an email is sent to the email address set in the CloudFormation parameter. This email includes a link to the sample web application and a temporary password that must be reset upon initial log in. The link to the sample web application is also provided in the **Outputs** tab of the CloudFormation stack (labeled `ConsoleUrl`).

Appendix C: Uninstall the solution

You can uninstall the Multi-Region Application Architecture using the AWS Management Console or the AWS Command Line Interface (AWS CLI). However, you must manually delete the Amazon Simple Storage Service (Amazon S3) buckets and CloudWatch Logs created by this solution. If you deployed the optional sample web application template, you must delete that stack prior to deleting the solution's main stack because CloudFormation Exports are used to share values from the solution's main stack to the sample web application stack.

If you have previously deployed this solution, uninstall the previous first before installing the latest version. The Amazon S3 buckets and CloudWatch Logs can also be deleted since the updated version of the solution does not access the previous version's resources.

> **Note**
> AWS CloudFormation StackSets are automatically deleted when you uninstall the solution's stack.

Using the AWS Management Console

1. Sign in to the AWS CloudFormation console.
2. On the **Stacks** page, select the solution stack.
3. Choose **Delete**.

Using AWS Command Line Interface

Verify that the AWS Command Line Interface (AWS CLI) is available in your environment. For installation instructions, refer to What Is the AWS Command Line Interface in the *AWS CLI User Guide*. After confirming the AWS CLI is available, run the following command.

```
$ aws cloudformation delete-stack --stack-name <cloudformation-stack-name>
```

Replace *<cloudformation-stack-name>* with the name of your CloudFormation stack.

Deleting the Amazon S3 buckets

This solution is configured to retain the Amazon S3 buckets if you decide to delete the AWS CloudFormation stack to prevent against accidental data loss. After uninstalling the solution, you can manually delete the S3 buckets if you do not need to retain the data. Follow these steps to delete the Amazon S3 buckets.

1. Sign in to the Amazon S3 console.
2. Choose **Buckets** from the left navigation pane.
3. Locate the *<stack-name>* S3 buckets.
4. Select one of the S3 buckets and choose **Delete**.

Repeat the steps until you have deleted all the *<stack-name>* S3 buckets.

To delete the S3 buckets using AWS CLI, run the following command:

```
$ aws s3 rb s3://<bucket-name> --force
```

Alternatively, you can configure the AWS CloudFormation template to delete the Amazon S3 buckets automatically. Prior to deleting the stack, change the deletion behavior in the AWS CloudFormation DeletionPolicy attribute.

Deleting the CloudWatch Logs

This solution retains the CloudWatch Logs if you decide to delete the AWS CloudFormation stack to prevent against accidental data loss. After uninstalling the solution, you can manually delete the logs if you do not need to retain the data. Follow these steps to delete the CloudWatch Logs.

1. Sign in to the Amazon CloudWatch console.
2. Choose **Log Groups** from the left navigation pane.
3. Locate the log groups created by the solution.
4. Select one of the log groups.
5. Choose **Actions** and then choose **Delete**.

Repeat the steps until you have deleted all the solution log groups.

Appendix D: Collection of operational metrics

This solution includes an option to send anonymous operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. When enabled, the following information is collected and sent to AWS:

- **Solution ID:** The AWS solution identifier
- **Version:** The version of the deployed solution
- **Timestamp:** The UTC formatted timestamp of when the event occured
- **Unique ID (UUID):** Randomly generated, unique identifier for each solution deployment
- **Event Name:** A value of `STATE_UPDATED` is used to indicate the state of the application has changed
- **Event Value:** A value representing the new state (`active`, `fenced`, `failover`) for the application

Note that AWS will own the data gathered via this survey. Data collection will be subject to the AWS Privacy Policy. To opt out of this feature, modify the AWS CloudFormation template mapping section as follows:

```
  Solution:
      Metrics:
        SendAnonymousData: Yes
```

to

```
Solution:
    Metrics:
      SendAnonymousData: No
```

Source code

You can visit our GitHub repository to download the templates and scripts for this solution, and to share your customizations with others.

Contributors

The following individuals contributed to this document:

- Eric Quinones

Document revisions

Date	Change
June 2020	Initial release
January 2021	Release version 1.1.0: Added capability for write access in the secondary Region for failovers; separated the front-end component to be a separate and optional stack; updated the back-end layers; for more information about the changes, refer to the CHANGELOG.md file in the GitHub repository

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents AWS current product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. AWS responsibilities and liabilities to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Multi-Region Application Architecture is licensed under the terms of the of the Apache License Version 2.0 available at The Apache Software Foundation.