## 

Account Creation using Landing Zone Accelerator

Table of Contents

[Description 3](#_Toc163553782)

[Architecture 4](#_Toc163553783)

[Step Function Workflow for Account Creation 4](#_Toc163553784)

[Folder Structure 4](#_Toc163553785)

[Pre-requisite Steps 7](#_Toc163553786)

[DISCLAIMER 9](#_Toc163553787)

[Deployment Steps 9](#_Toc163553788)

[How to Run AWS Step Function 10](#_Toc163553789)

[Arguments 10](#_Toc163553790)

[Invoking the AWS Step Function using the provided python script 11](#_Toc163553791)

[Invoking the AWS Step Function using API Gateway 12](#_Toc163553792)

[Check AWS Account Name Availability 12](#_Toc163553793)

[Check Account Creation Status 13](#_Toc163553794)

[Troubleshooting 13](#_Toc163553795)

[Issue #1 13](#_Toc163553796)

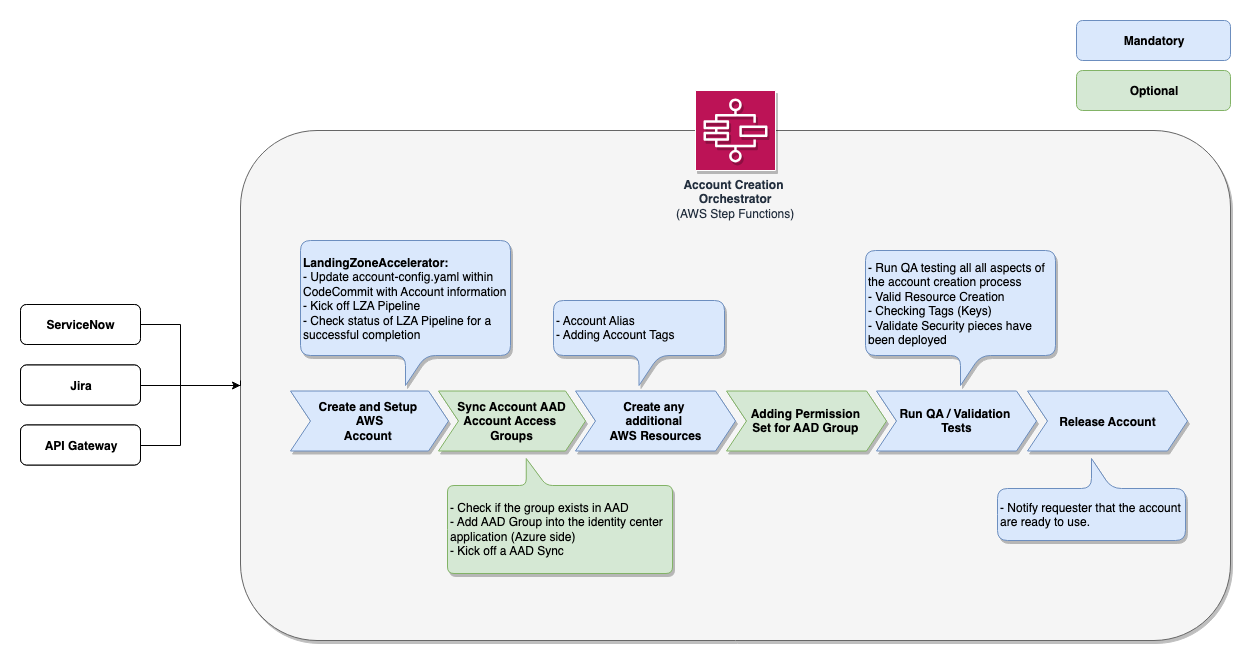
[Solution 14](#_Toc163553797)

## Description

This solution will enable the ability to provide customers with a Self-Service mechanism to deploy a new AWS Account using the Landing Zone Accelerator (LZA) solution. It uses an AWS Step Function to orchestrate a number of AWS Lambda Functions to add the account information to a Git repository, trigger the LZA CodePipeline, validate that all AWS Resources are in place, and then send out a completion email to the person requesting the account.

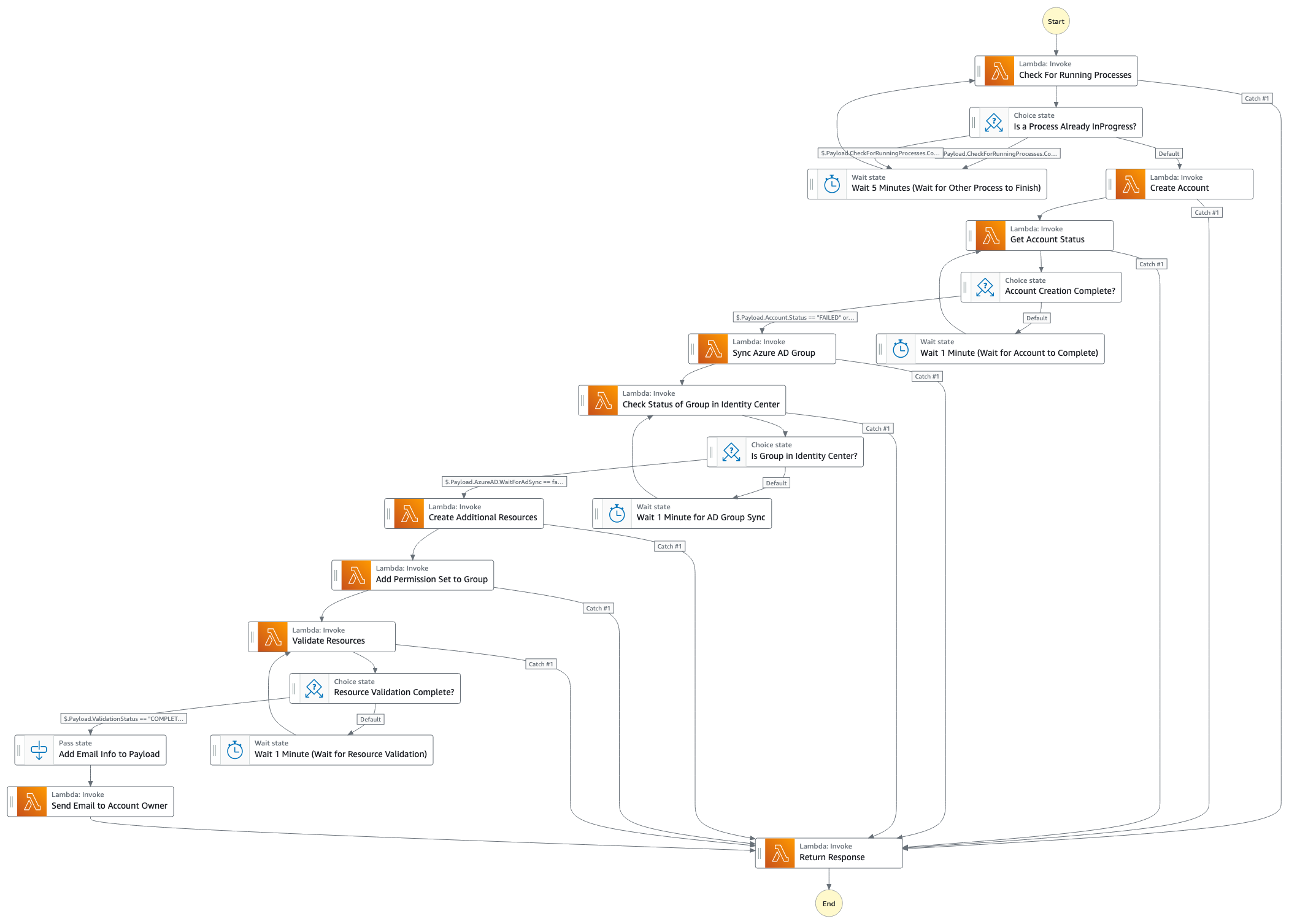
Optionally, there is a feature that will integrate Microsoft Entra ID (Azure Active Directory) groups with permission sets during the account request. This feature can be enabled in the deploy-config.yaml file.

## Architecture



### 

### Step Function Workflow for Account Creation



### 

### Folder Structure

| Folder/File | Description |
| --- | --- |
| app | CDK Application Infrastructure Code (e.g. Lambdas / StepFunctions / SES / S3). |
| app/cdk\_helpers | Directory to holds all definitions used to help deploy and manage the Account Creation solution. |
| app/lambda\_layer | Directory to holds all AWS Lambda Layer’s source code. This is code that is referenced in more than one Lambda Function. |
| app/lambda\_layer/account\_creation\_helper | AWS Lambda Layer that has all common code that is used across the solution. This includes modules for assuming roles / checking Service Catalog progress. |
| app/lambda\_layer/boto3 | AWS Lambda Layer for the boto3, since AWS Lambda doesn’t always get packaged with the latest version of Boto3. |
| app/lambda\_layer/identity\_center\_helper | AWS Lambda Layer that support common AWS IAM Identity Center calls. |
| app/lambda\_src | Directory to hold AWS Lambda Functions source code. |
| app/lambda\_src/api | Directory to hold AWS Lambda Functions that are used behind AWS API Gatway. |
| app/lambda\_src/api/CreateStatus | AWS Lambda Function that will get the status of a running or completed AWS Step Function execution of the Account Creation solution. |
| app/lambda\_src/api/NameAvailability | AWS Lambda Function that will check to see AWS Account Name exists in AWS Organizations. |
| app/lambda\_src/api/RunStepFunction | AWS Lambda Function that will run the Account Creation AWS Step Function. This is also the Lambda Function that the Terraform code will run. |
| app/lambda\_src/event | Directory to hold AWS Lambda Functions that are triggered by an Event. |
| app/lambda\_src/event/AccountTagToSsmParameter | AWS Lambda Function creates an SSM Parameter in the target account based on Tags attached to the account within AWS Organizations. The SSM Parameter will be prefixed with “/account/tags/”. |
| app/lambda\_src/stepfunction | Directory to hold AWS Lambda Functions that are used within the AWS Step Function. |
| app/lambda\_src/stepfunction/AttachPermissionSet | AWS Lambda Function that will add a permissions set to an SSO Group. |
| app/lambda\_src/stepfunction/AzureADGroupSync | AWS Lambda Function that will sync the desired Microsoft Entra ID Group to AWS AWS IAM Identity Center. |
| app/lambda\_src/stepfunction/CheckForRunningProcesses | AWS Lambda Function that will check to see if the Decommissioning CodeBuild project and LZA Pipeline is currently running. If one of those resources are running it will delay the AWS Step Function. |
| app/lambda\_src/stepfunction/CreateAccount | AWS Lambda Function that will use LZA to create an AWS Account. |
| app/lambda\_src/stepfunction/CreateAddtionalResources | AWS Lambda Function that will create AWS resources that couldn’t be managed by LZA or CloudFormation (e.g. Account Alias / Service Catalog Tags). |
| app/lambda\_src/stepfunction/GetAccountStatus | AWS Lambda Function that will scan the AWS Service Catalog Provisioned Product to see if the account creation has completed. |
| app/lambda\_src/stepfunction/ReturnResponse | AWS Lambda Function that will return either an Account Number (if account creation successful) or an error message (if there is a failure in the creation process). |
| app/lambda\_src/stepfunction/SendEmailWithSES | AWS Lambda Function that will send out emails to account requester or team mates waiting for the account creation to finish. |
| app/lambda\_src/stepfunction/ValidateADGroupSyncToSSO | AWS Lambda Function validate that the desired Microsoft Entra ID Group to AWS IAM Identity Center. |
| app/lambda\_src/stepfunction/ValidateResources | AWS Lambda Function that will ensure all Control Tower Customizations have run successfully. |
| app/stepfunction | Directory that holds the AWS Step Function definitions. |
| configs | Configuration files used for the solution. |
| configs/deploy-config.yaml | Configuration file used for deployment and application infrastructure. |
| images | Images used in README document. |
| pipeline | CDK Deployment Infrastructure Code (e.g. CodePipeline / CodeCommit / CodeBuild). |
| scripts | Supporting scripts to ensure the Solution uses best practices |
| tests | All testing code should reside. |
| requirements.txt | Pip requirements file for deployment environment. |

## 

## Pre-requisite Steps

* [Install Landing Zone Accelerator into the Management Account](https://aws.amazon.com/solutions/implementations/landing-zone-accelerator-on-aws/).
* [Install the Cloud Development Kit (CDK)](https://docs.aws.amazon.com/cdk/v2/guide/getting_started.html).
* [Install Docker](https://docs.docker.com/engine/install/) and start the Docker Engine.
* Ensure you have AWS CLI and Console access to the AWS Management Account.
  + NOTE: Please read disclaimer.
* (Optional) If you choose to use the Microsoft Entra ID integration the following steps will need to be performed to create a AWS Secret for that integration.
  + [How to get the required data from Microsoft Entra ID](docs/GET_MS_DATA.md)
* (Optional) If you would like to integrate the AWS Permissions Sets to an Microsoft Entra ID Group you will need to create an AWS Secret for GraphAPI.
  + Use the values collected from the previous step to set the variables for the AWS CLI Command.
* Example.
* # Variables   
  TENANT\_ID='00000000-1111-2222-3333-444444444444'  
  CLIENT\_ID='55555555-6666-7777-8888-999999999999'  
  OBJECT\_ID='aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee'  
  SECRET\_ID='ffffffff-gggg-hhhh-iiii-jjjjjjjjjjjj'  
  SECRET\_VALUE='\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'  
  APP\_ROLE\_ID='kkkkkkkk-llll-mmmm-nnnn-oooooooooooo'  
  ENTERPRISE\_APP\_NAME='AAAA'  
    
  # Creating the Secret  
  aws secretsmanager create-secret --name GraphApiSecret --secret-string "{\"client\_id\": \"${CLIENT\_ID}\", \"tenant\_id\": \"${TENANT\_ID}\", \"object\_id\": \"${OBJECT\_ID}\", \"app\_role\_id\": \"${APP\_ROLE\_ID}\", \"secret\_value\": \"${SECRET\_VALUE}\", \"secret\_id\": \"${SECRET\_ID}\"}"  
    
  # Updating the Secret  
  aws secretsmanager update-secret --secret-id GraphApiSecret --secret-string "{\"client\_id\": \"${CLIENT\_ID}\", \"tenant\_id\": \"${TENANT\_ID}\", \"object\_id\": \"${OBJECT\_ID}\", \"app\_role\_id\": \"${APP\_ROLE\_ID}\", \"secret\_value\": \"${SECRET\_VALUE}\", \"secret\_id\": \"${SECRET\_ID}\"}"
  + Use LZA to install the 3 dependant AWS CloudForamtion templates located in the dependency’s directory.
    - account-creation-validation-role.yaml : This template will be deployed to all AWS Accounts and will provide the Validation Lambda Function with read-only access to the new account. This will ensure that the account was provisioned properly.
    - account-tagging-to-ssm-parameter-role.yaml : This template will be deployed to all AWS Accounts and will create an AWS SSM Parameter Store Parameter based on tags stored on the account within AWS Organizations. Each SSM Parameter will be prefixed with “/account/tags/”.
    - config-log-validation-role.yaml : This template will be deployed into the LogArchive AWS Account and allow the Validation Lambda Function access to the S3 Bucket and AWS Config rules to ensure that they are setup properly.
    - Update the customizations-config.yaml file to deploy the 3 CloudFormation templates. Below is an example of the new entries for the config file.
  + # customizations-config.yaml  
    customizations:  
     cloudFormationStacks:  
     # --------------------------------------  
     # Account Creation / Validation Roles  
     # --------------------------------------  
     - deploymentTargets:  
     organizationalUnits:  
     - Root  
     description: IAM Role to allow Account Validation  
     name: lza-account-creation-validation  
     regions:  
     - us-east-1  
     template: cloudformation/account-creation-validation-role.yaml  
     runOrder: 1   
     terminationProtection: true  
     parameters:  
     - name: pManagementAccountId  
     value: "{{ account Management }}"  
     - deploymentTargets:  
     accounts:  
     - LogArchive  
     description: IAM Role to validate Config and Logs  
     name: lza-config-log-validation-role  
     regions:  
     - us-east-1  
     template: cloudformation/config-log-validation-role.yaml  
     runOrder: 1   
     terminationProtection: true   
     parameters:  
     - name: pManagementAccountId  
     value: "{{ account Management }}"   
     # ---------------------------------  
     # Account Tagging to Account SSM  
     # ---------------------------------  
     - deploymentTargets:  
     organizationalUnits:  
     - Root  
     excludedAccounts:  
     - Management  
     description: IAM Role to create SSM Parameters based on Account Tagging  
     name: lza-account-tagging-to-ssm-parameter  
     regions:  
     - us-east-1  
     template: cloudformation/account-tagging-to-ssm-parameter-role.yaml  
     runOrder: 1   
     terminationProtection: true   
     parameters:  
     - name: pManagementAccountId  
     value: "{{ account Management }}"

### DISCLAIMER

This solution will be deployed into your AWS Management Account. The AWS Management Account is a highly sensitive account that should be protected as much as possible using the least privileged permission model. We recommend that customers use a federated role for access *NOT* and an IAM user. The required permissions are listed below.

For this example S3 Bucket Access Logging is not enabled but is recommended that you do so when added to your enterprise.

## Deployment Steps

* Ensure you have access to the AWS Management Account.
* Change directory into the repository directory.
* cd lza-account-creation-workflow
* Update the config/deploy-config.yaml file with the appropriate values. Typical values that will need updating; accountCreationFailure, accountCompletionFromEmail, ssoLoginUrl, rootEmailPrefix, rootEmailDomain, useGraphApiSync, and enableAzureADIntegration.
  + To use the optional Microsoft Entra ID integration, you will need to set enableAzureADIntegration to true and make sure that the graphApiSecretName value matches the AWS Secret created in the prerequisite step.
* Ensure that the Docker Engine is running, then run cdk the following commands to deploy the solution’s deployment infrastructure. This will allow the solution to be enhanced via a CI/CD Pipeline. This will setup the deployment pipeline and dependent resources (e.g. CodeCommit / CodeBuild / CodePipeline). After the deployment the Git Repository (CodeCommit) will be populated automatically and trigger the CI/CD Pipeline (CodePipeline). Once the pipeline is complete the solution will be completely deployed.
* cdk bootstrap  
  cdk synth  
  cdk deploy
* *If not deploying into the AWS Region us-east-1. This is needed since AWS Organizations only pushed Cloudtrail events to us-east-1, this solution will allow AWS Organizations to push events to other regions using the targets event bridge.*
  + Get the Custom LZA Account Creation Workflow EventBus Arn (oLzaAccountCreationWorkflowEventBusArn) from *lza-account-creation-workflow-application* CloudFormation Output.
  + Switch over to AWS Region, us-east-1 and run the following command.
* aws cloudformation deploy --stack-name AccountTagToSsmParameterCrossRegionEventRule --template-file dependencies/global-resource-event-rule.yaml --parameter-override pMainEventRuleArn=<<Custom\_EventBus\_Arn>> --capabilities CAPABILITY\_NAMED\_IAM

## How to Run AWS Step Function

### Arguments

* **account-name** **(-a)** (*string*) – [REQUIRED]
* The name for the newly managed AWS Account that will be created by AWS Service Catalog / Control Tower.
* **support-dl** **(-s)** (*string*) – [REQUIRED]
* Support Distribution Email Address for the new AWS Account.
* **managed-org-unit** **(-m)** (*string*) – [REQUIRED]
* Managed organizational unit. The managed Account will be placed under this Organizational Unit.
* **ad-integration** **(-ad)** (*string dictionary*) –
* Microsoft Entra ID Group integration to SSO Permission Sets.
* Example.
* --ad-integration "{\"PermissionSetName\": \"AzureActiveDirectoryGroupName\"}"
* **account-email** **(-e)** (*string*) –
* The email address used for the root userof the newly managed AWS Account. If this argument isn’t used an email address will be generated using the values *rootEmailPrefix* and *rootEmailDomain* from the deploy-config.yaml file.
* Example of the generated email address rootEmailPrefix+awsAccountName@rootEmailDomain.
* **region** **(-r)** (*string*) –
* AWS Region in which the AWS Step Function exists. Default: us-east-1
* **force-update** **(-f)** (*string boolean*) –
* This argument will force a Service Catalog update Provisioned Product.
* **bypass-creation** **(-b)** (*string boolean*) –
* Skip adding the Account to the accounts-config.yaml and skip running of the of the Landing Zone Accelorator CodePipeline. This argument is typically used for testing the Account Creation Workflow process.
* **tags** **(-t)** (*string*) –
* Additional tag to add to the AWS Account. Default Tags: account-name, vendor (aws), product-version and support-dl.
* Example.
* --tags TEST1=VALUE1 TEST2=VALUE2

### Invoking the AWS Step Function using the provided python script

* Ensure you have access to the AWS Management Account.
* Change directory into the repository directory.
* cd lza-account-creation-workflow
* Ensure requirement-run.txt has been installed on the machine you would like to invoke AWS Step Function.
  + To install run requirements use the following command.
  + pip install -r requirements-run.txt
* Run the following commands to invoke the AWS Step Function.
* python ./run-stepfunction.py \  
   --account-name "lza-test-01" \  
   --support-dl "johnsmith@example.com" \  
   --managed-org-unit "Workloads/Workload-1" \  
   --force-update true \  
   --ad-integration "{\"CustomerAccountAdmin\": \"platform-admin\", \"CustomerAccountDev\": \"workload1-app1\"}" \  
   --bypass-creation true \  
   --tags APPLICATION=TestingMicroService

### Invoking the AWS Step Function using API Gateway

* Get the API Gateway Endpoint from the *lza-account-creation-workflow-application* CloudFormation Output.
* Use the API Gateway Endpoint value in the following command to invoke the AWS Step Function.
* curl --header "Content-Type: application/json" \  
   --request POST \  
   --aws-sigv4 "aws:amz:us-east-1:execute-api" \  
   --user "${AWS\_ACCESS\_KEY\_ID}:${AWS\_SECRET\_ACCESS\_KEY}" \  
   --data @api-example.json \  
   https://<<API\_Gateway\_Endpoint>>/execute

### Check AWS Account Name Availability

The AWS Account Name is available.

curl --request GET \  
 --aws-sigv4 "aws:amz:us-east-1:execute-api" \  
 --user "${AWS\_ACCESS\_KEY\_ID}:${AWS\_SECRET\_ACCESS\_KEY}" \  
 https://<<API\_Gateway\_Endpoint>>/check\_name?account\_name=lza2-test-113

Result

Null

The AWS Account Name is not available

curl --request GET \  
 --aws-sigv4 "aws:amz:us-east-1:execute-api" \  
 --user "${AWS\_ACCESS\_KEY\_ID}:${AWS\_SECRET\_ACCESS\_KEY}" \  
 https://<<API\_Gateway\_Endpoint>>/check\_name?account\_name=lza2-test-11

Result

"000000000000"

### Check Account Creation Status

Pass

curl --request GET \  
 --aws-sigv4 "aws:amz:us-east-1:execute-api" \  
 --user "${AWS\_ACCESS\_KEY\_ID}:${AWS\_SECRET\_ACCESS\_KEY}" \  
 https://<<API\_Gateway\_Endpoint>>/create\_status?execution=lza2-test-12-16

Pass Result

{"Status": "SUCCEEDED"}

Failure

curl --request GET \  
 --aws-sigv4 "aws:amz:us-east-1:execute-api" \  
 --user "${AWS\_ACCESS\_KEY\_ID}:${AWS\_SECRET\_ACCESS\_KEY}" \  
 https://<<API\_Gateway\_Endpoint>>/create\_status?execution=lza2-test-12-09

Failure Result

{"Status": "FAILED", "Cause": "{\"errorMessage\":\"('A task in the account creation step function failed: ', 'The accounts-config.yaml already contains an account with the name of %s and the force update flag is set to False lza2-test-12')\",\"errorType\":\"TypeError\",\"requestId\":\"00000000-1111-2222-3333-444444444444\",\"stackTrace\":[\" File \\\"/var/task/main.py\\\", line 53, in lambda\_handler\\n raise TypeError(str(e)) from e\\n\"]}"}

## Troubleshooting

### Issue #1

'A task in the account creation step function failed: ', 'An error occurred (AccessDenied) when calling the AssumeRole operation: User: arn:aws:sts::000000000000:assumed-role/lza-account-creation-work-rLambdaFunctionValidateRe-aaaaaaaaaaaa/ValidateResources is not authorized to perform: sts:AssumeRole on resource: arn:aws:iam::111111111111:role/account-creation-validation'

### Solution

Ensure that the role located in the deploy-config.yaml *lzaAccountValidationRole* matches the role that has been deployed within the LZA solution.