include::attributes.txt[]

// Attributes

[.topic] [#bootstrapping-troubleshoot] = Troubleshoot {aws} CDK bootstrapping issues :info\_titleabbrev: Troubleshoot bootstrapping :keywords: {aws} CDK, Bootstrapping, Troubleshoot

== [abstract]

## Troubleshoot common issues when bootstrapping your environment with the {aws} Cloud Development Kit ({aws} CDK).

// Content start

Troubleshoot common issues when bootstrapping your environment with the {aws} Cloud Development Kit ({aws} CDK).

* For an introduction to bootstrapping, see xref:bootstrapping[{aws} CDK bootstrapping].
* For instructions on bootstrapping, see xref:bootstrapping-env[Bootstrap your environment for use with the {aws} CDK].

[#bootstrapping-troubleshoot-s3-bucket-name] == When bootstrapping using the default template, you get a ‘CREATE\_FAILED’ error for the Amazon S3 bucket

When bootstrapping using the {aws} CDK Command Line Interface (CDK CLI) cdk bootstrap command with the default bootstrap template, you receive the following error:

== [source,none,subs=“verbatim,attributes”]

CREATE\_FAILED | {aws}::S3::Bucket | ++++++already exists —-++++++

Before troubleshooting, ensure that you are using the latest version of the CDK CLI.

* To check your version, run cdk --version.
* To install the latest version, run npm install -g aws-cdk.

After installing the latest version, try bootstrapping your environment again. If you still receive the same error, continue with troubleshooting.

[#bootstrapping-troubleshoot-s3-bucket-name-causes] === Common causes

When you bootstrap your environment, the {aws} CDK generates physical IDs for your bootstrap resources. For more information, see xref:bootstrapping-env-default-id[Resource IDs created during bootstrapping].

Unlike the other bootstrap resources, Amazon S3 bucket names are global. This means that each bucket name must be unique across all {aws} accounts in all {aws} Regions within a partition. For more information, see https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingBucket.html[Buckets overview] in the *Amazon S3 User Guide*. Therefore, the most common cause of this error is that the physical ID generated as your bucket name already exists somewhere within the partition. This could be within your account or another account.

The following is an example bucket name: cdk-hnb659fds-assets-012345678910-us-west-1. While unlikely, due to the qualifier and account ID being a part of the name, it is possible that this name for an Amazon S3 bucket is used by another {aws} account. Since bucket names are globally scoped, it can`’t be used by you if its used by a different account in the same partition. Most likely, a bucket with the same name exists somewhere in your account. This could be in the Region you are attempting to bootstrap, or another Region.

Generally, the resolution is to locate this bucket in your account and determine what to do with it, or customize bootstrapping to create bootstrap resources of a different name.

[#bootstrapping-troubleshoot-s3-bucket-name-resolution] === Resolution

First, determine if a bucket with the same name exists within your {aws} account. Using an {aws} identity with valid permissions to lookup Amazon S3 buckets in your account, you can do this in the following ways:

* Use the {aws} Command Line Interface ({aws} CLI) aws s3 ls command to view a list of all your buckets.
* Look up bucket names for each Region in your account using the https://console.aws.amazon.com/s3[Amazon S3 console].

If a bucket with the same name exists, determine if it’s being used. If it’s not being used, consider deleting the bucket and attempting to bootstrap your environment again.

If a bucket with the same name exists and you don’t want to delete it, determine if it’s already associated with a bootstrap stack in your account. You may have to check multiple Regions. The Region in the Amazon S3 bucket name doesn’t necessarily mean that the bucket is in that Region. To check if it’s associated with the CDKToolkit bootstrap stack, you can do either of the following for each Region:

* Use the {aws} CLI aws cloudformation describe-stack-resources --stack-name <CDKToolkit> --region <Region> command to view the resources in your bootstrap stack and check if the bucket is listed.
* In the https://console.aws.amazon.com/cloudformation[{aws} CloudFormation console], locate the CDKToolkit stack. Then, on the *Resources* tab, check if the bucket exists.

If the bucket is associated with a bootstrap stack, determine if the bootstrap stack is in the same Region that you are attempting to bootstrap. If it is, your environment is already bootstrapped and you should be able to start using the CDK to deploy applications into your environment. If the Amazon S3 bucket is associated with a bootstrap stack in a different Region, you`’ll need to determine what to do. Possible resolutions include renaming the existing Amazon S3 bucket, deleting the current Amazon S3 bucket if its not being used, or using a new name for the Amazon S3 bucket you are attempting to create.

If you are unable to locate an Amazon S3 bucket with the same name in your account, it may exist in a different account. To resolve this, you`’ll need to customize bootstrapping to create new names for all of your bootstrap resources or for just your Amazon S3 bucket. To create new names for all bootstrap resources, you can modify the qualifier. To create a new name for only your Amazon S3 bucket, you can provide a new bucket name.

To customize bootstrapping, you can use options with the CDK CLI cdk bootstrap command or by modifying the bootstrap template. For instructions, see xref:bootstrapping-customizing[Customize {aws} CDK bootstrapping].

If you customize bootstrapping, you will need to apply the same changes to synthesis before you can properly deploy an application. For instructions, see xref:bootstrapping-custom-synth[Customize CDK stack synthesis].

For example, you can provide a new qualifier with cdk bootstrap:

== [source,none,subs=“verbatim,attributes”]

$ cdk bootstrap –qualifier ++++++—-++++++

The following is an example Amazon S3 bucket name that will be created with this modification: cdk-abcde0123-assets-012345678910-us-west-1. All bootstrap resources created during bootstrapping will use this qualifier.

When developing your CDK app, you must specify your custom qualifier in your synthesizer. This helps the CDK with identifying your bootstrap resources during synthesis and deployment. The following is an example of customizing the default synthesizer for your stack instance:

==== [role=“tablist”] TypeScript:: + [source,javascript,subs=“verbatim,attributes”] — new MyStack(this, ‘MyStack’, { synthesizer: new DefaultStackSynthesizer({ qualifier: ‘abcde0123’, }), }); —

JavaScript:: + [source,javascript,subs=“verbatim,attributes”] — new MyStack(this, ‘MyStack’, { synthesizer: new DefaultStackSynthesizer({ qualifier: ‘abcde0123’, }), }) —

Python:: + [source,python,subs=“verbatim,attributes”] — MyStack(self, “MyStack”, synthesizer=DefaultStackSynthesizer( qualifier=“abcde0123” )) —

Java:: + [source,java,subs=“verbatim,attributes”] — new MyStack(app, “MyStack”, StackProps.builder() .synthesizer(DefaultStackSynthesizer.Builder.create() .qualifier(“abcde0123”) .build()) .build(); ) —

C#:: + [source,csharp,subs=“verbatim,attributes”] — new MyStack(app, “MyStack”, new StackProps { Synthesizer = new DefaultStackSynthesizer(new DefaultStackSynthesizerProps { Qualifier = “abcde0123” }) }); —

Go:: + [source,go,subs=“verbatim,attributes”] — func NewMyStack(scope constructs.Construct, id string, props \*MyStackProps) awscdk.Stack { var sprops awscdk.StackProps if props != nil { sprops = props.StackProps } stack := awscdk.NewStack(scope, &id, &sprops)

…. synth := awscdk.NewDefaultStackSynthesizer(&awscdk.DefaultStackSynthesizerProps{ Qualifier: jsii.String(“abcde0123”), })

stack.SetSynthesizer(synth)

return stack } —- ….

You can also specify the new qualifier in the cdk.json file of your CDK project:

== [source,json,subs=“verbatim,attributes”]

{ “app”: “…”, “context”: { “@aws-cdk/core:bootstrapQualifier”: “abcde0123” } } —

To modify only the Amazon S3 bucket name, you can use the xref:ref-cli-cmd-bootstrap-options-bootstrap-bucket-name[--bootstrap-bucket-name] option. The following is an example:

== [source,none,subs=“verbatim,attributes”]

$ cdk bootstrap –bootstrap-bucket-name ‘++++++’ —- ====++++++

When developing your CDK app, you must specify your new bucket name in your synthesizer. The following is an example of customizing the default synthesizer for your stack instance:

==== [role=“tablist”] TypeScript:: + [source,javascript,subs=“verbatim,attributes”] — new MyStack(this, ‘MyStack’, { synthesizer: new DefaultStackSynthesizer({ fileAssetsBucketName: ‘my-new-bucket-name’, }), }); —

JavaScript:: + [source,javascript,subs=“verbatim,attributes”] — new MyStack(this, ‘MyStack’, { synthesizer: new DefaultStackSynthesizer({ fileAssetsBucketName: ‘my-new-bucket-name’, }), }) —

Python:: + [source,python,subs=“verbatim,attributes”] — MyStack(self, “MyStack”, synthesizer=DefaultStackSynthesizer( file\_assets\_bucket\_name=‘my-new-bucket-name’ )) —

Java:: + [source,java,subs=“verbatim,attributes”] — new MyStack(app, “MyStack”, StackProps.builder() .synthesizer(DefaultStackSynthesizer.Builder.create() .fileAssetsBucketName(“my-new-bucket-name”) .build()) .build(); ) —

C#:: + [source,csharp,subs=“verbatim,attributes”] — new MyStack(app, “MyStack”, new StackProps { Synthesizer = new DefaultStackSynthesizer(new DefaultStackSynthesizerProps { FileAssetsBucketName = “my-new-bucket-name” }) }); —

Go:: + [source,go,subs=“verbatim,attributes”] — func NewMyStack(scope constructs.Construct, id string, props \*MyStackProps) awscdk.Stack { var sprops awscdk.StackProps if props != nil { sprops = props.StackProps } stack := awscdk.NewStack(scope, &id, &sprops)

…. synth := awscdk.NewDefaultStackSynthesizer(&awscdk.DefaultStackSynthesizerProps{ FileAssetsBucketName: jsii.String(“my-new-bucket-name”), })

stack.SetSynthesizer(synth)

return stack } —- ==== ….

[#bootstrapping-troubleshoot-s3-bucket-name-prevention] === Prevention

We recommend that you proactively bootstrap each {aws} environment that you plan to use. For more information, see xref:bootstrapping-env-when[When to bootstrap your environment]. Specifically for the Amazon S3 bucket naming issue, this will create Amazon S3 buckets in each {aws} environment and prevent others from using your Amazon S3 bucket name.