AWS’ native Infrastructure as Code (IaC) service, CloudFormation has a new feature named “Disable Rollback”. Disable Rollback works exactly as it sounds and as easily as it sounds. If, during the creation or update of a CloudFormation Stack a failure occurs, rather than rolling back the creation or update of the stack, the current state is maintained. This is of benefit to those developing CloudFormation templates as it allows the current state of the stack’s resources to be interactively reviewed, the root cause for the failure to be identified, remediated and the stack creation or update to be retried from the point of failure, reducing the time spent iterating through issues, fixes and tests.

The feature is deployed as a runtime option, and is provided during the creation or update of the stack (in other words, it is not part of the template or stack itself), so can be selectively enabled or disabled (disabled being the default state) during each creation or update execution. As it’s a runtime option, it is accessible via the AWS CLI or the Console.

While this feature can work standalone (as demonstrated later in this post), it becomes increasingly beneficial when combined with a CloudFormation template development pipeline. By introducing this feature into the pipeline, and utilizing notifications, a developer can follow a typical template development approach of committing template changes to a version control system of their choice, have a bug created in their ticketing system (from the CloudFormation notification) when the create/update fails and see, real time, why the create/update failed. Corrections can then be made via another commit, at which point the create/update continues from the failure point.

Outside of the runtime option, and the behavior it evokes, all other behaviors remain the same, including retention/cleanup, stack set and change set functionality and custom resource behavior. This is a key item to note as no other changes need to be done to templates, custom resources, or existing processes for managing templates and stacks in order to make use of this feature. The same also holds true for failure states – individual resource update or creation failures (and their nuances for why their creation/update failed) will trigger a failure of the stack creation/update and based on the enabled/disable state of the Disable Rollback feature, the stack will be either rolled back or not.

Now that I’ve talked about what it is and why I’m going to be using it for my projects, I’ll briefly demonstrate, using the AWS Console, how it works.

Let’s start with a simple template that creates an S3 bucket (note that the deletion policy is set to retain, which will make demonstration of this feature very easy):

Text

Description automatically generated

I’ll create a stack using this template without (or with) the Disable Rollback feature enabled (since the bucket is new, it will not have any effect):

Graphical user interface, text, application

Description automatically generated

Once the stack has been created, I’ll create another stack, using the same template, this time with the feature enabled:

Graphical user interface, text, application, email, Teams

Description automatically generated

Once the stack creation starts, I will very quickly see it fail, but instead of rolling back, the Console will show me the following message:

Graphical user interface, text, application

Description automatically generated

From here I can either choose to retry, update or roll back (which was the only behavior prior to the introduction of this feature). Retry works exactly as it sounds – it will retry the stack creation/update without any changes. Update, on the other hand, allows you to provide updated parameters (including the template) before proceeding.

What’s great about having both retry and update options is that I can address issues that are likely environmental, and not template driven (such a resource that already exists, as demonstrated with in this example) by manually intervening (deleting the resource, in this case) or I can correct the template itself (providing a new S3 bucket name, in this case), update it and continue the process of creating/updating the stack, without having to create a new stack.

For my example, I’ll delete the bucket, and then retry:

Graphical user interface, text, application, email

Description automatically generated

And, as expected, it was a success:

Graphical user interface, application

Description automatically generated

As you can see, the feature is very easy to use and very easy to demonstrate its capabilities. The feature can work standalone, or can be integrated into a pipeline, addressing the practices and preferences of a variety of individuals and organizations. Features such this one will only continue to drive adoption of CloudFormation and Infrastructure as Code, which in and of itself is exciting!

Ultimately this capability dramatically reduces the development time required to create and deploy applications, which allows us to help our customers to spend more time on their products and services and allows us spend more time on architecture, compliance, security and managed services.