Cloudformation Template Documentation

Overview

This document will provide a brief explanation of the deployed environment and a brief explanation of the CloudFormation templates.

Additional material

There is a high-level diagram which provides an overview of the solution in the same location as this document, the name of the file is *overview.pdf*

Deployment Strategy

I used CloudFormation templates to deploy the solution as it provides potential for further automation.

I tried to logically separate each part of the solution as it allowed for faster deployments and easier troubleshooting with the intention being to combine all the yml files into one larger all-encompassing CloudFormation template once everything was working.

The repo consists of the following templates:

vpc-m.yml

This is an aws template that I used to deploy a custom VPC, found here: https://docs.aws.amazon.com/codebuild/latest/userguide/cloudformation-vpc-template.html

Only changes I made were to the CIDR blocks via the parameters, this can easily be stripped out so that no user interaction would be required.

s3.yml

For the creation of an s3 bucket for use by the elastic beanstalk deployment. I enabled versioning and encryption, not publicly viewable by default.

roles.yml

This was used for troubleshooting while running up the elastic beanstalk stack. I ran into quite a few issues and so I created a new role and allowed it to assume

arn:aws:iam::aws:policy/AWSElasticBeanstalkFullAccess while it spun up resources for the beanstalk deployment. I eventually discovered that the issue was the method I was using to select instance types for the autoscaling group. I added some additional *OptionName* parameters in eb-m.yml which allowed it proceed further into the deployment.

rds-m.yml

For the deployment of the RDS Postgres solution into a private subnet of the custom VPC via providing a DBSubnetGroup, which requires subnets in at least two unique Availability Zones.

Deployed postgres 11.6 and allocated 20GB of storage.

eb-m.yml

For the elastic beanstalk deployment, unfortunately beanstalk likes to deploy into the default VPC, therefore I needed to add the following *OptionNames*:

- VPCID
- Subnets
- ELBSubnets -
- DBSubnets
- ELBScheme

The elastic beanstalk deployment also pulls the ruby app from s3.

Testing

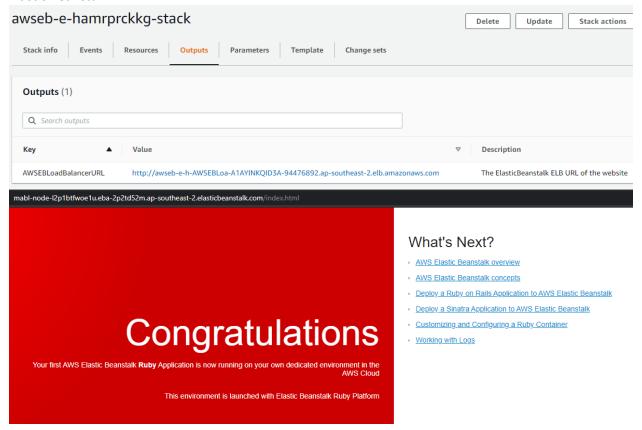
Was performed manually, will briefly outline some of the tests below:

Database

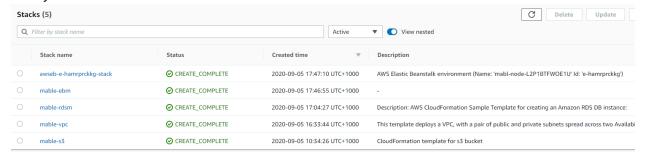
Spun up an ec2 instance which I used to connect to the postgres db

```
[ec2-user@ip-10-10-10-52~]$ psql -U sysalykes -h mablerdsm.cbdlh26f7kda.ap-southeast-
2. rds. amazonaws.com -d mableDBm
Password for user sysalykes:
psql (9.2.24, server 11.6)
WARNING: psql version 9.2, server version 11.0.
        Some psql features might not work.
SSL connection (cipher: ECDHE-RSA-AES256-GCM-SHA384, bits: 256)
Type "help" for help.
mableDBm=> \1
                                  List of databases
  Name
              Owner
                     | Encoding | Collate
                                                    Ctype
                                                                  Access privileges
mableDBm | sysalykes | UTF8
                                en_US. UTF-8 | en_US. UTF-8 |
                              en_US. UTF-8 | en_US. UTF-8 |
postgres | sysalykes | UTF8
```

Elastic Beanstalk



Cloudformation Stacks



Further CloudFormation Optimisations

- Once the VPC is created, it is then required to manually replace any hardcoded values for VPCIDs and Subnets in the cloudformation templates.
- Create a single cloudformation template for the entire stack.