Infrastructure as Code

AWS Solutions Best Practices

Richard Boyd, Sr. Developer Advocate, AWS Code Services
Luis Colon, Sr. Developer Advocate, AWS CloudFormation

April 20, 2020

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
Agenda

Infrastructure as Code (IaC): The Basics

IaC with AWS CloudFormation

Best Practices

AWS Cloud Development Kit (CDK)

Other IaC Tools
Infrastructure as Code: The Basics
Managing cloud applications involves managing the lifecycle of its resources:
Managing cloud applications involves managing the lifecycle of its resources:
Managing cloud applications involves managing the lifecycle of its resources:
Resources

The building blocks, or components of cloud applications
Resources

The building blocks, or components of cloud applications

Several options to manage your resource’s lifecycle:
Resources

The building blocks, or components of cloud applications

Several options to manage your resource’s lifecycle:

AWS Management Console
Resources

The building blocks, or components of cloud applications

Several options to manage your resource’s lifecycle:

AWS Management Console

AWS Command Line Interface
Resources

The building blocks, or components of cloud applications

Several options to manage your resource’s lifecycle:

AWS Management Console
AWS Command Line Interface
AWS Tools & Software Development Kits
Resources

The building blocks, or components of cloud applications

Several options to manage your resource's lifecycle:

AWS Management Console
AWS Command Line Interface
AWS Tools & Software Development Kits
AWS CloudFormation
Infrastructure as Code
With AWS CloudFormation
Infrastructure as Code

AWSTemplateFormatVersion: "2010-09-09"
Description: A CodeCommit Repo and Cloud9 Environment

Resources:
MyRepo:
  Type: "AWS::CodeCommit::Repository"
  Properties:
    RepositoryName: MyRepo
    RepositoryDescription: Sample Repository for Demo

MyC9Environment:
  Type: "AWS::Cloud9::EnvironmentEC2"
  Properties:
    Repositories:
      - PathComponent: /cfn
        RepositoryUrl: !GetAtt MyRepo.CloneUrlHttp
    InstanceType: t2.micro
AWS CloudFormation

The code template describes the intended state of your resources

CloudFormation translates the intention to API calls
AWS CloudFormation

The code template describes the intended state of your resources

CloudFormation translates the intention to API calls
AWS CloudFormation

The code template describes the intended state of your resources

CloudFormation translates the intention to API calls

1. Code your template
2. Upload, test, review changes
AWS CloudFormation

The code template describes the intended state of your resources

CloudFormation translates the intention to API calls

1. Code your template
2. Upload, test, review changes
3. A stack is created by executing the changes

© 2020, Amazon Web Services, Inc. or its Affiliates. All rights reserved.
AWS CloudFormation

The code template describes the intended state of your resources

CloudFormation translates the intention to API calls

1. Code your template
2. Upload, test, review changes
3. A stack is created by executing the changes
4. Manage many stacks and stack sets over time
AWS CloudFormation
Best Practices
Best Practices: Core Artifacts
Best Practices: Core Artifacts

Template

Describes resources, attributes, dependencies and their intended state
Best Practices: Core Artifacts

**Template**
Describes resources, attributes, dependencies and their intended state

**Change Set**
Describes an execution plan to implement the intended state of a stack’s resources
Best Practices: Core Artifacts

**Template**
Describes resources, attributes, dependencies and their intended state

**Change Set**
Describes an execution plan to implement the intended state of a stack’s resources

**Stack**
A group of resources and their intended states
**Best Practices: Core Artifacts**

- **Template**: Describes resources, attributes, dependencies and their intended state.
- **Change Set**: Describes an execution plan to implement the intended state of a stack's resources.
- **Stack**: A group of resources and their intended states.
- **StackSet**: A group of stack instances across accounts and regions.
AWSTemplateFormatVersion: "2010-09-09"
Description: A CodeCommit Repo and Cloud9 Environment
Resources:
  MyRepo:
    Type: "AWS::CodeCommit::Repository"
    Properties:
      RepositoryName: MyRepo
      RepositoryDescription: Sample Repository for Demo

## Once a repo is created, tie the Cloud9 EC2 Instance to
## the repository automatically
MyC9Environment:
  Type: "AWS::Cloud9::EnvironmentEC2"
  Properties:
    Repositories:
      -PathComponent: /cfn
        RepositoryUrl: !GetAtt MyRepo.CloneUrlHttp
  InstanceType: t2.micro
AWSTemplateFormatVersion: "2010-09-09"
Description: A CodeCommit Repo and Cloud9 Environment
Resources:
  MyRepo:
    Type: "AWS::CodeCommit::Repository"
    Properties:
      RepositoryName: MyRepo
      RepositoryDescription: Sample Repository for Demo

## Once a repo is created, tie the Cloud9 EC2 Instance to
## the repository automatically

MyC9Environment:
  Type: "AWS::Cloud9::EnvironmentEC2"
  Properties:
    Repositories:
    - PathComponent: /cfn
      RepositoryUrl: !GetAtt MyRepo.CloneUrlHttp
      InstanceType: t2.micro
Templates

```yaml
AWSTemplateFormatVersion: "2010-09-09"
Description: A sample template

- Errors:
  - Catch: Missing

- Parameters:
  - myParam:
    Type: String
    Default: String
    Description: String

- Resources:
  - MyEC2Instance1:
    Type: "AWS::EC2::Instance1"
    Properties:
      ImageId: "ami-2f726546"
      InstanceType: t1.micro
      KeyName: t
      Properties:
        Ebs:
          -VolumeType: gp2
          -Size: 10
          -DeleteOnTermination: True
          -Iops: 100
        BlockDeviceMappings:
          - DeviceName: /dev/sda1
            Ebs:
              VolumeSize: 10
              VolumeType: gp2
              DeleteOnTermination: True
              KmsKeyId: None
```
Templates

```
AWSTemplateFormatVersion: "$2010-09-09"
Description: A sample template

- Errors:
  - Catch: Missing

- Parameters:
  - myParam:
    - Type: String
    - Default: String
    - Description: String

- Resources:
  - MyEC2Instance1:
    - Type: "$AWS::EC2::Instance1"

  - MyEC2Instance:
    - Type: "$AWS::EC2::Instance"
```

<table>
<thead>
<tr>
<th>Severity</th>
<th>Provider</th>
<th>Description</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Warning</td>
<td>Cfn-Lint</td>
<td>Top level item Errors isn't valid</td>
<td>3:1</td>
</tr>
<tr>
<td>Warning</td>
<td>Cfn-Lint</td>
<td>Parameter myParam not used</td>
<td>6:1</td>
</tr>
<tr>
<td>Warning</td>
<td>Cfn-Lint</td>
<td>Invalid Type AWS::EC2::Instance1 for resource MyEC2Instance1</td>
<td>12:1</td>
</tr>
<tr>
<td>Warning</td>
<td>Cfn-Lint</td>
<td>Properties not defined for resource MyEC2Instance1</td>
<td>12:1</td>
</tr>
<tr>
<td>Warning</td>
<td>Cfn-Lint</td>
<td>Invalid Property FakeKey for resource MyEC2Instance</td>
<td>18:1</td>
</tr>
<tr>
<td>Warning</td>
<td>Cfn-Lint</td>
<td>Invalid Property BadSubX2Key for resource MyEC2Instance</td>
<td>26:1</td>
</tr>
</tbody>
</table>
Change Sets

1. Create change set
2. View/accept change set
3. (optional) Create additional change sets
4. Execute change set

Original Stack/Template → Change Set → Change Set → AWS CloudFormation updates your stack
Layer your application
Extract configuration

Resources:
MyRDSDB:
Type: "AWS::RDS::DBInstance"
Properties:
  DBInstanceClass: db.t2.medium
  AllocatedStorage: '20'
  Engine: mariadb
  EngineVersion: '10.2'
  MasterUsername: appadmin
  MasterUserPassword:
    '{{resolve:ssm-secure:ssbRDSmEcntl:1}}'
Infrastructure as Code
With AWS Cloud Development Kit (CDK)
CDK

Model infrastructure as reusable components

class UrlShortener extends Stack {
  constructor(scope: App, id: string, props?: UrlShortenerProps) {
    super(scope, id, props);

    const vpc = new ec2.Vpc(this, 'vpc', { maxAzs: 2 });
    const cluster = new ecs.Cluster(this, 'cluster', { vpc: vpc });
    const service = new patterns.NetworkLoadBalancedFargateService(this, 'sample-app', {
      cluster,
      taskImageOptions: {
        image: ecs.ContainerImage.fromAsset('ping'),
      },
    });

    });
    domainName
    domainZone

    // Setup AutoScaling policy
    const scaling = service.service.autoScaleTaskScaling.
      scaleOnCpuUtilization('CpuScaling',
        targetUtilizationPercent: 50,
        scaleInCooldown: Duration.seconds(60),
        scaleOutCooldown: Duration.seconds(60)
      );
  }
}
CDK

Multi-language Framework
CDK

Constructs
Source Code

CDK CLI

Templates + Assets
Cloud Assembly

AWS CloudFormation

execute
synthesize
deploy
provision
CDK Main Components
CDK Main Components

Core Framework
CDK Main Components

Core Framework

AWS Construct Library
CDK Main Components

Core Framework

AWS Construct Library

AWS CDK CLI
export class MyCoolServiceStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, props);
    const api = new apigw.RestApi(this, 'Api');
    // ...
    new Walters.HealthMonitor(this, 'Monitor', {
      app: 'MyCoolService',
      region: 'us-east-1',
      endpoint: api.url
    });
  }
}
export class MyCoolServiceStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, props);

    const api = new apigw.RestApi(this, 'Api');
    // ...

    new Walters.HealthMonitor(this, 'Monitor', {
      app: 'MyCoolService',
      region: 'us-east-1',
      endpoint: api.url
    });
  }
}
CDK Constructs

L1 constructs – all resources in CloudFormation specification
CDK Constructs

L1 constructs – all resources in CloudFormation specification

L2 constructs – higher-level abstractions with sensible defaults
CDK Constructs

L1 constructs – all resources in CloudFormation specification
L2 constructs – higher-level abstractions with sensible defaults
L3 constructs – opinionated reference architectures and design patterns using multiple AWS services
new patterns.ApplicationLoadBalancedFargateService(stack, 'MyFargateService',{  
taskImageOptions: {  
    image: ecs.ContainerImage.fromRegistry("amazon/amazon-ecs-sample")  
  }])
Infrastructure as Code: Additional Tooling Options
AWS Native Options

AWS CloudFormation

AWS Cloud Development Kit (CDK)
Additional AWS Options

AWS OpsWorks

AWS Service Catalog
Third Party Options

Terraform
Chef
Puppet
Ansible
SaltStack
Pulumi
...many more
Summary

Infrastructure as Code (IaC) makes managing cloud applications and their resources more repeatable, safer

AWS provides multiple options for IaC, including CloudFormation and CDK

IaC best practices apply across AWS and non-AWS tools
More Information

AWS Site and Documentation
https://aws.amazon.com/cloudformation/
https://docs.aws.amazon.com/cloudformation/index.html
https://aws.amazon.com/cdk/

AWS Open Source Resources
https://github.com/aws
https://github.com/aws/aws-cdk
https://github.com/awslabs/aws-cloudformation-templates
https://github.com/aws-cloudformation/cfn-python-lint
https://github.com/aws-cloudformation/cloudformation-cli
Q&A
Thank You!

Richard Boyd, Sr. Developer Advocate, AWS Code Services
Luis Colon, Sr. Developer Advocate, AWS CloudFormation

April 20, 2020