Automating the Cloud with Infrastructure as Code

This ain’t your Dad’s IaC

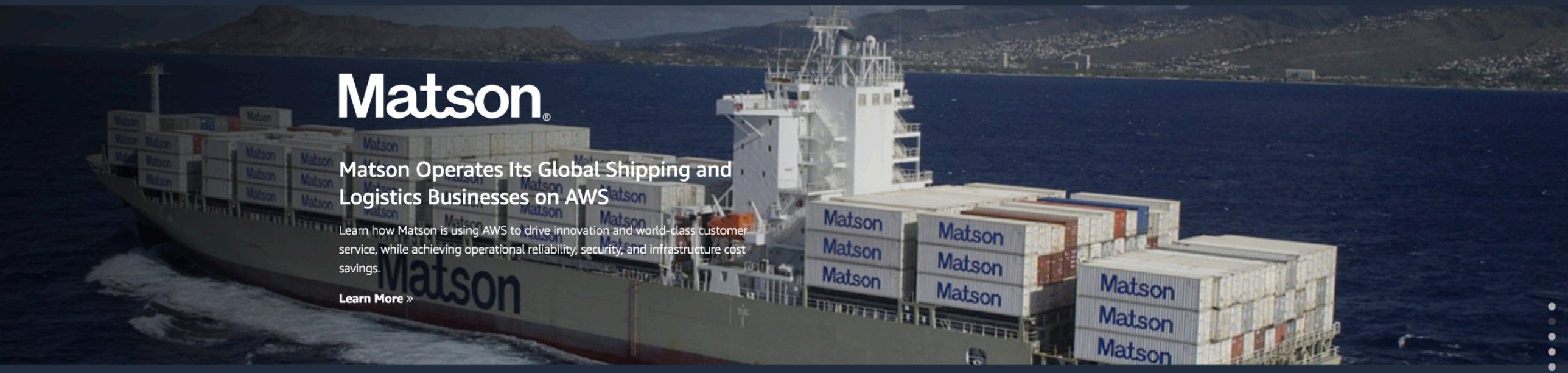
@davetownsend

Dave Townsend

Principal Software Engineer

Innovation & Architecture

Matson®



Real-Time Container Tracking

Matson built a flagship mobile application for global container tracking that allows customers to perform real-time tracking of their freight shipments. Other valuable features in the application include interactive vessel schedule searching, location-based port map lookups, and live gate-camera feeds.

Mobile Device Access

All mobile devices access AWS via **Amazon API Gateway**. This provides highly available edge located endpoints for access into resources within Matson's existing virtual private clouds.

Serverless Computing

The **AWS Lambda** functions are designed using the microservices pattern and are modeled around specific ocean-based business contexts, such as shipment tracking and vessel schedules.

Database Configuration and Storage

Amazon DynamoDB manages configuration as well as user-feedback configuration and user-feedback notifications sent from mobile devices. DynamoDB Streams provides real-time notifications to Matson's customer service team.

Data Monitoring and Alerts

Matson's customers rely on accurate, up-to-the-minute container tracking and vessel status information. Monitoring and alerting of system events is achieved by using **Amazon CloudWatch**, **Amazon SNS**, **Amazon SES**, **AWS Lambda**, and CloudWatch Logs.

End-to-End Serverless Application

Matson can now offer customers an end-to-end serverless application to help track their shipments, and has no infrastructure to maintain.

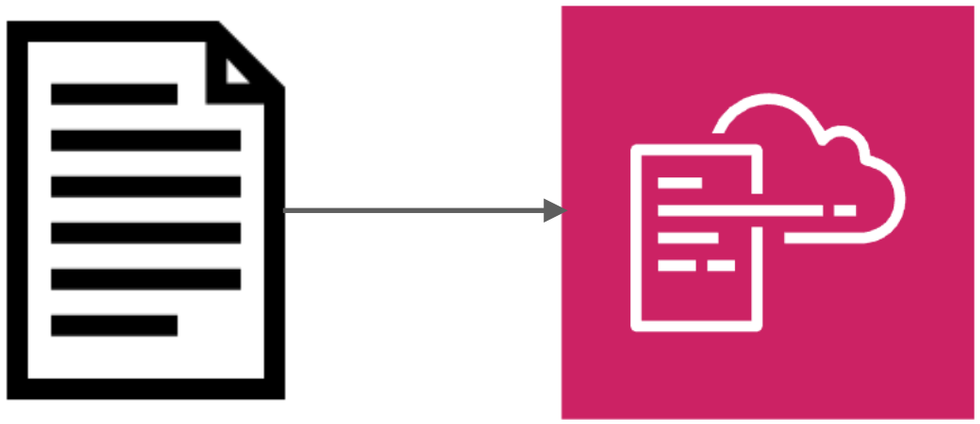
agenda

what is IaC?
why should we use IaC?
how to use IaC.

10k foot view

template

template CloudFormation

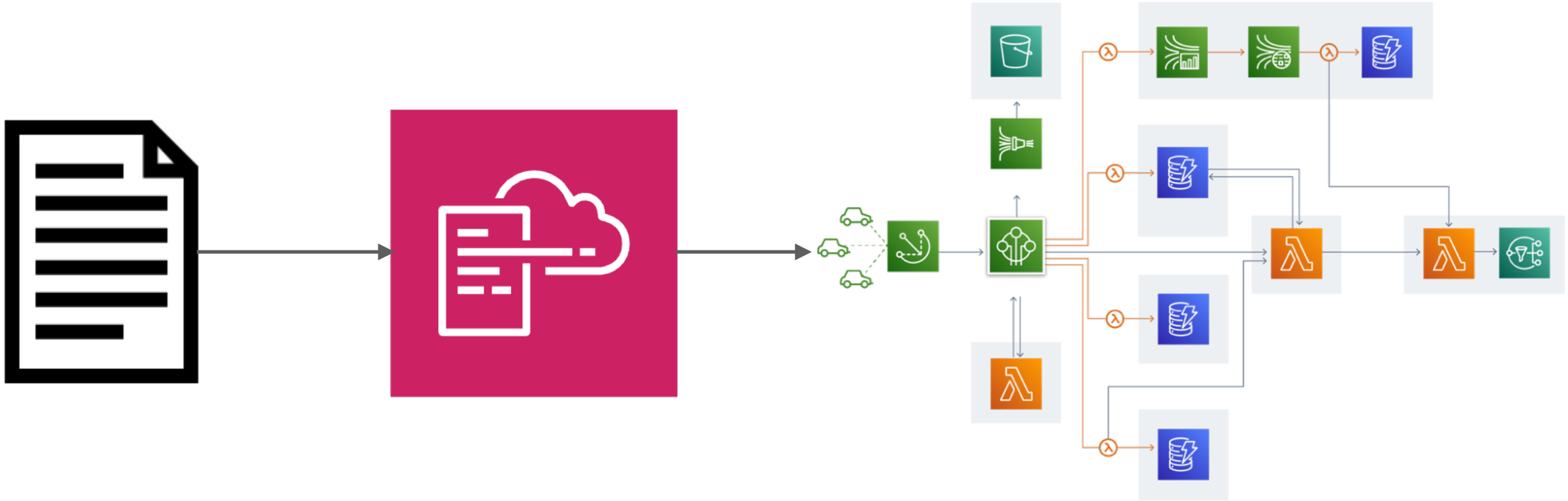


template CloudFormation stack

template

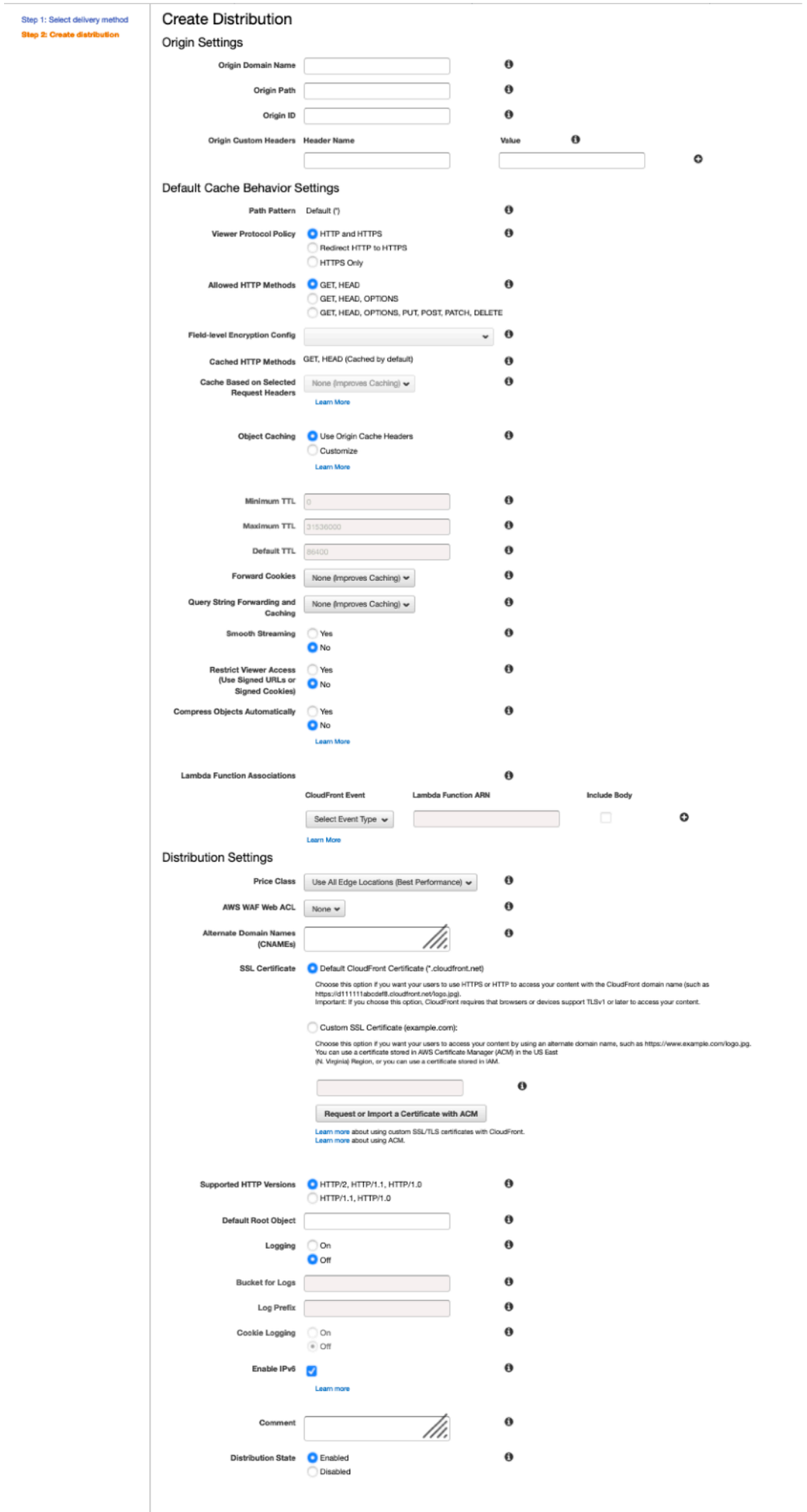
CloudFormation

stack



why?

this...


Step 1: Select delivery method
Step 2: Create distribution
Create Distribution
Origin Settings
Origin Domain Name
Origin Path
Origin ID
Origin Custom Headers
Header Name
Value
Default Cache Behavior Settings
Path Pattern
Default (*)
Viewer Protocol Policy
HTTP and HTTPS
Redirect HTTP to HTTPS
HTTPS Only
Allowed HTTP Methods
GET, HEAD
GET, HEAD, OPTIONS
GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE
Field-level Encryption Config
Cached HTTP Methods
GET, HEAD (Cached by default)
Cache Based on Selected Request Headers
None (Improves Caching)
Learn More
Object Caching
Use Origin Cache Headers
Customize
Learn More
Minimum TTL
0
Maximum TTL
31536000
Default TTL
86400
Forward Cookies
None (Improves Caching)
Query String Forwarding and Caching
None (Improves Caching)
Smooth Streaming
Yes
No
Restrict Viewer Access (Use Signed URLs or Signed Cookies)
Yes
No
Compress Objects Automatically
Yes
No
Learn More
Lambda Function Associations
CloudFront Event
Lambda Function ARN
Include Body
Select Event Type
Learn More
Distribution Settings
Price Class
Use All Edge Locations (Best Performance)
AWS WAF Web ACL
None
Alternate Domain Names (CNAMEs)
SSL Certificate
Default CloudFront Certificate (*.cloudfront.net)
Choose this option if you want your users to use HTTPS or HTTP to access your content with the CloudFront domain name (such as https://d111111abcdef8.cloudfront.net/logo.jpg).
Important: If you choose this option, CloudFront requires that browsers or devices support TLSv1 or later to access your content.
Custom SSL Certificate (example.com):
Choose this option if you want your users to access your content by using an alternate domain name, such as https://www.example.com/logo.jpg.
You can use a certificate stored in AWS Certificate Manager (ACM) in the US East (N. Virginia) Region, or you can use a certificate stored in IAM.
Request or Import a Certificate with ACM
Learn more about using custom SSL/TLS certificates with CloudFront.
Learn more about using ACM.
Supported HTTP Versions
HTTP/2, HTTP/1.1, HTTP/1.0
HTTP/1.1, HTTP/1.0
Default Root Object
Logging
On
Off
Bucket for Logs
Log Prefix
Cookie Logging
On
Off
Enable IPv6
Learn more
Comment
Distribution State
Enabled
Disabled

VS.

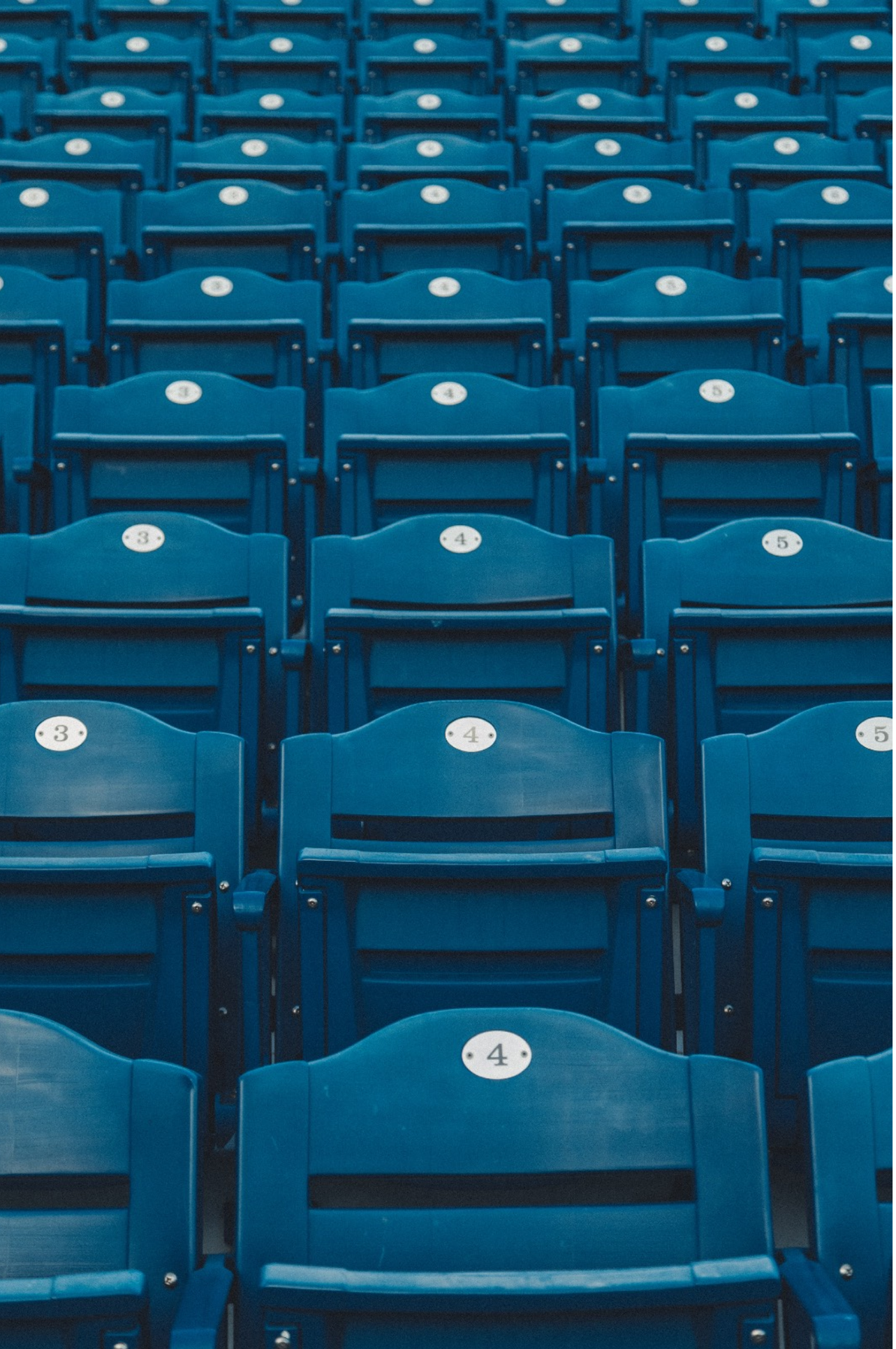
this.

```yaml
MyCloudFront:
  Type: AWS::CloudFront::Distribution
  Properties:
    DistributionConfig:
      Aliases:
        - !Ref "DomainName"
      CacheBehaviors: []
      DefaultCacheBehavior:
        AllowedMethods:
          - GET
          - HEAD
        CachedMethods:
          - HEAD
          - GET
        Compress: true
        TargetOriginId: S3Bucket
        ForwardedValues:
          QueryString: false
          Cookies:
            Forward: none
          Headers: []
      SmoothStreaming: false
      ViewerProtocolPolicy: redirect-to-https
    Enabled: true
    HttpVersion: http2
    Origins:
      - DomainName: !Sub "${DomainName}.s3-website-us-west-2.amazonaws.com"
        Id: S3Bucket
        CustomOriginConfig:
          HTTPPort: 80
          OriginProtocolPolicy: http-only
    PriceClass: PriceClass_100
    ViewerCertificate:
      SslSupportMethod: sni-only
      AcmCertificateArn: !Ref "SSLCertArn"
```

initial time investment.

buuut…

automation



deterministic

environment parity

disaster

recovery

deeper understanding
of the architecture

more control of the entire stack


We all face the same questions every day:
What do I want? And how can I get it?
How can I live more happily
and work more effectively?
The Decision Book
Fifty models for strategic thinking

tooling landscape

AWS CloudFormation

AWS CloudFormation

5 point plan for adopting IaC

1. start learning CloudFormation, now
2. stop requesting resources
3. don’t use the console to create resources (experiments ok)
4. build *everything* you need with IaC (start in a sandbox)
5. **submit templates, not tickets!** 👊😎

we can take this further…

use CI/CD!

> *Every cloud workflow in the org should share the same command to start a deployment:*
>
> `git push`

Richard Boyd
Cloud Data Engineer at @IRobot

yes, DB migrations are still hard

…but, a huge part (if not all) of it can be automated with IaC

learning CloudFormation

templates, stacks and change sets

templates

```
AWSTemplateFormatVersion: '2010-09-09'
 Resources:
  MyBucket:
    Type: AWS::S3::Bucket
```

CloudFormation template (yaml based)

```yaml
AWSTemplateFormatVersion: '2010-09-09'
Description: CloudFormation template for creating S3 bucket
Parameters:
  AppName:
    Description: Enter application name
    Type: String
  Stage:
    Description: Enter deployment stage
    Type: String
Resources:
  BuildBucket:
    Type: AWS::S3::Bucket
    Properties:
      BucketName: !Sub ${AppName}-services-${Stage}-build-artifact
      PublicAccessBlockConfiguration:
        BlockPublicAcls: true
        BlockPublicPolicy: true
        IgnorePublicAcls: true
        RestrictPublicBuckets: true
```

CloudFormation template (json based)

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "CloudFormation template for creating S3 bucket",
    "Parameters": {
        "AppName": {
            "Description": "Enter application name",
            "Type": "String"
        },
        "Stage": {
            "Description": "Enter deployment stage",
            "Type": "String"
        }
    },
    "Resources": {
        "BuildBucket": {
            "Type": "AWS::S3::Bucket",
            "Properties": {
                "BucketName": {
                    "Fn::Sub": "${AppName}-services-${Stage}-build-artifact"
                },
                "PublicAccessBlockConfiguration": {
                    "BlockPublicAcls": true,
                    "BlockPublicPolicy": true,
                    "IgnorePublicAcls": true,
                    "RestrictPublicBuckets": true
                }
            }
        }
    }
}
```

stacks

CloudFormation > Stacks

Stacks (15)

Delete | Update | Stack actions | Create stack

inspek | Active | View nested | 1

Stack name	Status	Created time	Updated time	Description
inspektor-web-deploy-role	CREATE_COMPLETE	2019-09-13 15:15:59 UTC-0700	-	Deployer role for inspektor website.
inspektor-web-pipeline	UPDATE_COMPLETE	2019-09-12 13:09:51 UTC-0700	2019-09-13 18:10:32 UTC-0700	Deploy pipeline for inspektor-web site
inspektor-website	CREATE_COMPLETE	2019-08-14 17:42:31 UTC-0700	-	Full Inspektor WebSite Stack (S3, CloudFront /w OAI, Route53, WAF WebACL)
inspektor-inspection-service-sandbox	UPDATE_COMPLETE	2019-07-31 16:14:07 UTC-0700	2019-09-06 13:08:28 UTC-0700	The AWS CloudFormation template for this Serverless application
inspektor-refdata-service-sandbox	UPDATE_COMPLETE	2019-06-06 11:49:00 UTC-0700	2019-09-06 13:07:12 UTC-0700	The AWS CloudFormation template for this Serverless application
inspektor-photo-service-sandbox	UPDATE_COMPLETE	2019-04-22 16:21:07 UTC-0700	2019-09-06 13:08:02 UTC-0700	The AWS CloudFormation template for this Serverless application
inspektor-services-codebuild-status-monitor	CREATE_COMPLETE	2019-04-18 15:07:42 UTC-0700	-	CodeBuild status notifications for inspektor-services
inspektor-status-service-sandbox	UPDATE_COMPLETE	2019-04-18 13:48:24 UTC-0700	2019-09-06 13:06:52 UTC-0700	The AWS CloudFormation template for this Serverless application
inspektor-notification-service-sandbox	UPDATE_COMPLETE	2019-04-17 15:30:28 UTC-0700	2019-09-06 13:07:41 UTC-0700	The AWS CloudFormation template for this Serverless application
inspektor-services-kms-key	UPDATE_COMPLETE	2019-04-15 16:10:14 UTC-0700	2019-08-06 17:17:01 UTC-0700	Creates KMS key for Inspektor-services.

stacks

inspektor-website

Delete | Update | Stack actions | Create stack

Stack info | Events | Resources | Outputs | Parameters | Template | Change sets

Overview

Stack ID
arn:aws:cloudformation[redacted]stack/inspektor-website/90b49710-bef5-11e9-88fa-0a33685a019e

Description
Full Inspektor WebSite Stack (S3, CloudFront /w OAI, Route53, WAF WebACL)

Status
CREATE_COMPLETE

Status reason
-

Root stack
-

Parent stack
-

Created time
2019-08-14 17:42:31 UTC-0700

Deleted time
-

Updated time
-

Drift status
NOT_CHECKED

Last drift check time
-

Termination protection
Disabled

IAM role
-

stacks

inspektor-website

Delete | Update | Stack actions | Create stack

Stack info | Events | Resources | Outputs | Parameters | Template | Change sets

Events

Search events

Timestamp	Logical ID	Status	Status reason
2019-08-14 18:29:26 UTC-0700	inspektor-website	CREATE_COMPLETE	-
2019-08-14 18:29:24 UTC-0700	InspektorWWWWebAddress	CREATE_COMPLETE	-
2019-08-14 18:29:24 UTC-0700	InspektorNonWWWWebAddress	CREATE_COMPLETE	-
2019-08-14 18:28:53 UTC-0700	InspektorWWWWebAddress	CREATE_IN_PROGRESS	Resource creation Initiated
2019-08-14 18:28:52 UTC-0700	InspektorNonWWWWebAddress	CREATE_IN_PROGRESS	Resource creation Initiated
2019-08-14 18:28:52 UTC-0700	InspektorWWWWebAddress	CREATE_IN_PROGRESS	-
2019-08-14 18:28:51 UTC-0700	InspektorNonWWWWebAddress	CREATE_IN_PROGRESS	-
2019-08-14 18:28:47 UTC-0700	InspektorCloudFrontWWW	CREATE_COMPLETE	-
2019-08-14 18:28:47 UTC-0700	InspektorCloudFrontNonWWW	CREATE_COMPLETE	-
2019-08-14 18:03:15 UTC-0700	InspektorCloudFrontNonWWW	CREATE_IN_PROGRESS	Resource creation Initiated
2019-08-14 18:03:11 UTC-0700	InspektorCloudFrontNonWWW	CREATE_IN_PROGRESS	-
2019-08-14 18:03:11 UTC-0700	InspektorCloudFrontWWW	CREATE_IN_PROGRESS	Resource creation Initiated
2019-08-14 18:03:06 UTC-0700	InspektorCloudFrontWWW	CREATE_IN_PROGRESS	-

stacks

inspektor-website

Delete | Update | Stack actions | Create stack

Stack info | Events | Resources | Outputs | Parameters | Template | Change sets

Resources (13)

Search resources

Logical ID	Physical ID	Type	Status	Status reason
InspektorCFOriginAccessIdentity	E18NNWF6BW995X	AWS::CloudFront::CloudFrontOriginAccessIdentity	CREATE_COMPLETE	-
InspektorCloudFrontNonWWW	E1UVYSSY6V1ARL	AWS::CloudFront::Distribution	CREATE_COMPLETE	-
InspektorCloudFrontWWW	E3QHLRC0K6GAAW	AWS::CloudFront::Distribution	CREATE_COMPLETE	-
InspektorLogBucket	logs-inspektor.matsonlabs.com	AWS::S3::Bucket	CREATE_COMPLETE	-
InspektorNonWWWWebAddress	inspektor-website-InspektorNonWWWWebAddress-15QXQQ7UNPCC0	AWS::Route53::RecordSetGroup	CREATE_COMPLETE	-
InspektorRedirectBucket	inspektor.matsonlabs.com	AWS::S3::Bucket	CREATE_COMPLETE	-
InspektorRedirectBucketPolicy	inspektor-website-InspektorRedirectBucketPolicy-1L5B0IR31PCY7	AWS::S3::BucketPolicy	CREATE_COMPLETE	-
InspektorSiteIPWhiteListSet	140ef26b-b9be-4d3c-9e23-1c9229707507	AWS::WAF::IPSet	CREATE_COMPLETE	-
InspektorSiteWAFRuleMatsonAccess	ab9ab92e-cb0b-4057-9d67-955e3f33aa21	AWS::WAF::Rule	CREATE_COMPLETE	-

stacks

create

update

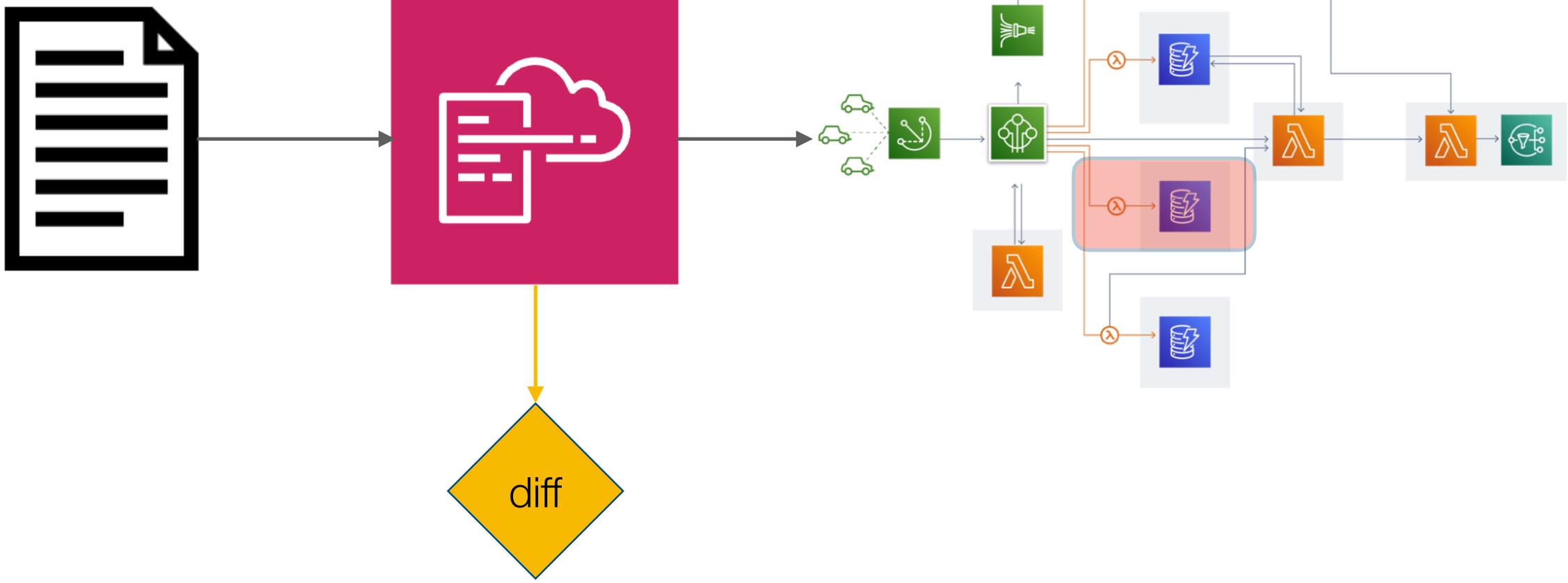
delete

update process

updated template

CloudFormation

stack



change sets

inspektor--deploy-role-change-set

Delete | Execute

Changes | Input | Template | JSON changes

5.

Overview

1.

Change set ID
arn:aws:cloudformation:us-west-2:275416279984:changeSet/inspektor--deploy-role-change-set/89abcc81-785d-4971-92d5-21febede2978

Status
CREATE_COMPLETE

Description
-

Status reason
-

2.

Created time
2019-09-16 10:17:24 UTC-0700

Execution status
AVAILABLE

Changes (1)

Search changes

1

3.

4.

Action	Logical ID	Physical ID	Resource type	Replacement
Modify	MvpSiteDeployerRole	inspektor-web-deploy-role	AWS::IAM::Role	False

update rules example

AWS::ApiGateway::Resource

The `AWS::ApiGateway::Resource` resource creates a resource in an API.

Syntax

To declare this entity in your AWS CloudFormation template, use the following syntax:

`PathPart`

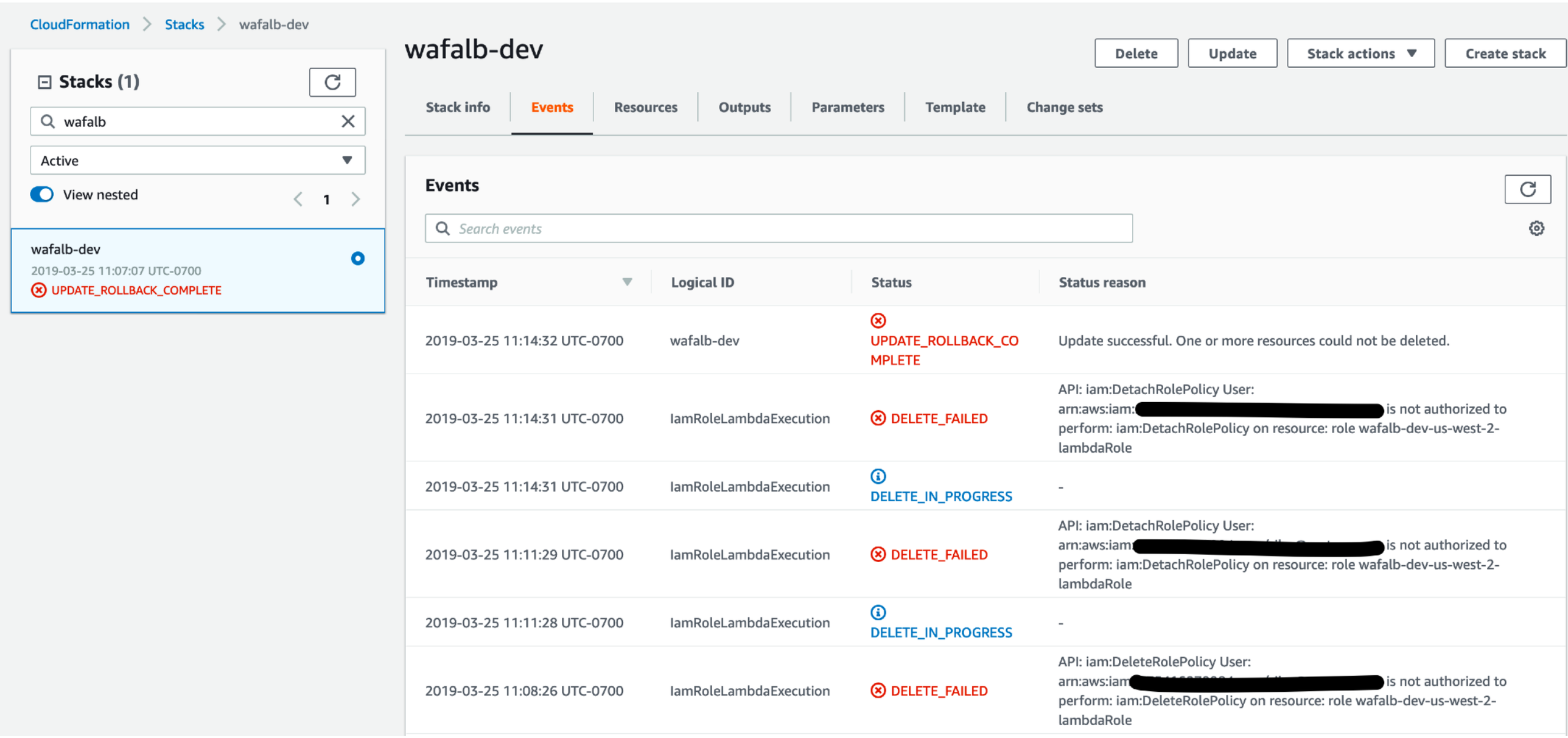
A path name for the resource.

Required: Yes

Type: String

Update requires: Replacement

stack errors



service roles

Configure stack options

Tags

You can specify tags (key-value pairs) to apply to resources in your stack. You can add up to 50 unique tags for each stack. **Learn more**

Key | *Value* | **Remove**

Add tag

Permissions

Choose an IAM role to explicitly define how CloudFormation can create, modify, or delete resources in the stack. If you don't choose a role, CloudFormation uses permissions based on your user credentials. **Learn more**

IAM role - optional

Choose the IAM role for CloudFormation to use for all operations performed on the stack.

IAM role name ▼ | cf-creator ▼ | **Remove**

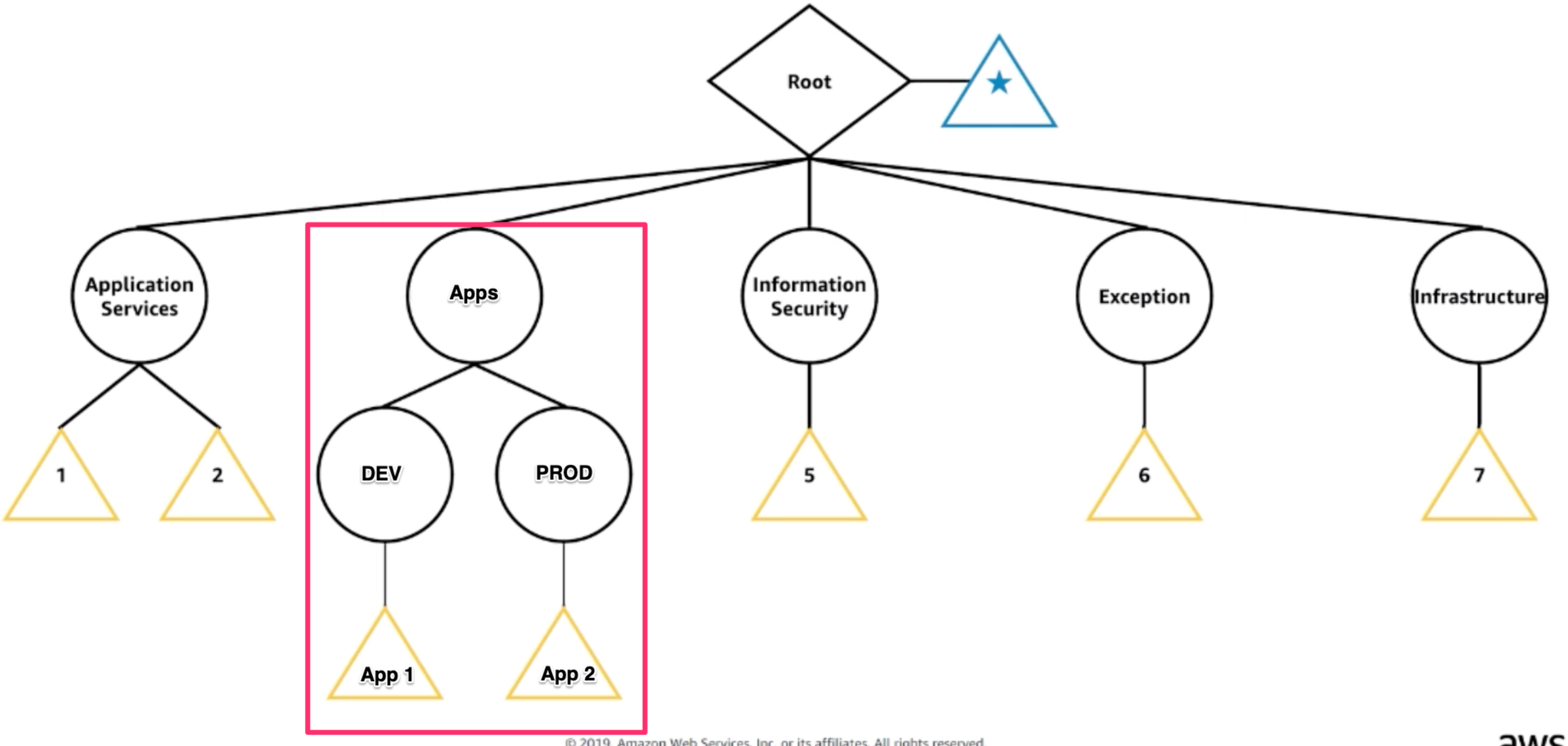
⚠ AWS CloudFormation will use this role for all stack operations. Other users that have permissions to operate on this stack will be able to use this role, even if they don't have permission to pass it. Ensure that this role grants least privilege.

be wary of micro-templates

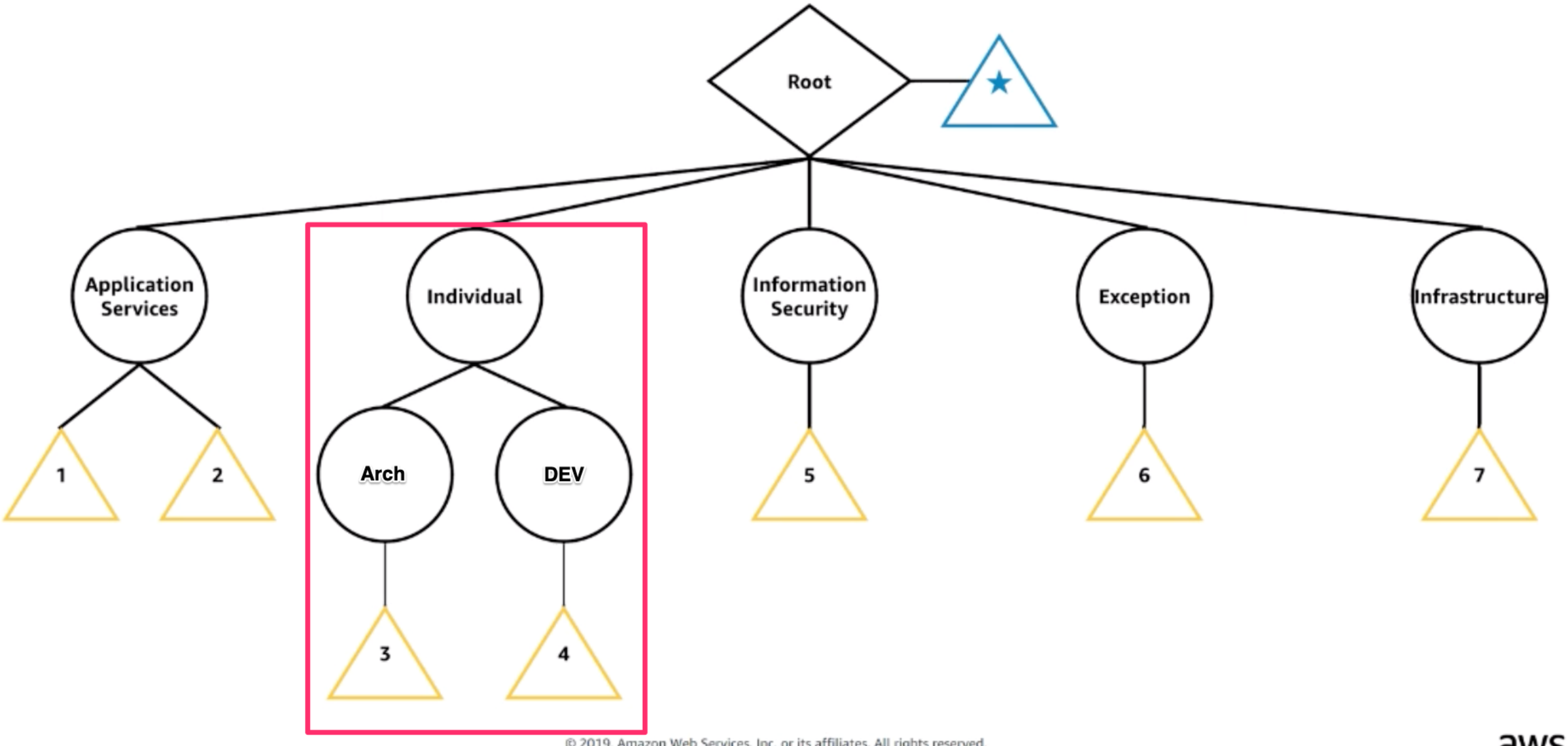
200 stacks per-account (hard limit)

AWS account management strategy

Organize your accounts



Organize your accounts



CloudFormation

-= core concepts =-

parameters
pseudo parameters
intrinsic functions
mappings

parameters

```yaml
AWSTemplateFormatVersion: "2010-09-09"
Parameters:
  SiteBucket:
    Description: Enter the site hosting S3 bucket name
    Type: String
  …

MyBucket:
  Type: AWS::S3::Bucket
  Properties:
    BucketName: !Ref SiteBucket

…

Statement:
  - Effect: Allow
    Action:
      - s3:PutObject
    Resource: !Sub "arn:aws:s3:::${SiteBucket}/*"
```

parameter lists

```yaml
Parameters:
  LambdaMemorySize:
    Type: String
    Default: 128
    AllowedValues:
      - 256
      - 512
      - 1024
    Description: Select memory size for Lambda
```

AWS parameter types

```
Parameters:
  myKeyPair:
    Description: Amazon EC2 Key Pair
    Type: "AWS::EC2::KeyPair::KeyName"
  mySubnetIDs:
    Description: Subnet IDs
    Type: "List<AWS::EC2::Subnet::Id>"
```

parameter constraints

```
AWSTemplateFormatVersion: "2010-09-09"
Parameters:
  SiteBucket:
    Description: Enter the site hosting S3 bucket name
    Type: String
    MinLength: "5"
    MaxLength: "20"
    AllowedPattern: "[a-zA-Ż][a-zA-Ż0-9]"
```

parameter constraints

```
AWSTemplateFormatVersion: "2010-09-09"
Parameters:
  DataBasePassword:
    Description: Enter the database password
    Type: String
    MinLength: "1"
    MaxLength: "20"
    NoEcho: true
```

pseudo parameters

```
AWS::AccountId
AWS::NotificationARNs
AWS::NoValue
AWS::Partition
AWS::Region
AWS::StackId
AWS::StackName
AWS::URLSuffix
```

pseudo parameters

```yaml
Resources:
  DeployerRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Sub ${AWS::StackName}-role
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Principal:
              AWS: !Sub arn:aws:iam:${AWS::Region}:${AWS::AccountId}:role/name
              Service: s3.amazonaws.com
            Action: sts:AssumeRole
      Path: /
```

intrinsic functions

```
Fn::Base64
Fn::Cidr
Fn::FindInMap
Fn::GetAtt
Fn::GetAZs
Fn::ImportValue
Fn::Join
Fn::Select
Fn::Split
Fn::Sub
Fn::Transform
Ref
Conditional Functions
```

Sub
Ref
Join
FindInMap
GetAtt
Conditional Functions

Sub

```
!Sub ${String}
```

Sub

```yaml
Resources:
  DeployerRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Sub ${AWS::StackName}-role
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Principal:
              AWS: !Sub arn:aws:iam::${AWS::AccountId}:role/${XaccntRoleName}
              Service: s3.amazonaws.com
            Action: sts:AssumeRole
      Path: /
```

Ref

```
!Ref logicalName
```

Ref

```
RedirectBucket:
    Type: AWS::S3::Bucket
    DeletionPolicy: Delete
    Properties:
      BucketName: !Ref DomainName
      AccessControl: Private

 BucketPolicy:
    Type: AWS::S3::BucketPolicy
    Properties:
      Bucket: !Ref RedirectBucket
…
```

Ref

AWS::DynamoDB::Table

Filter View: All

The `AWS::DynamoDB::Table` resource creates a DynamoDB table. For more information, see CreateTable in the *Amazon DynamoDB API Reference*.

Return Values

Ref

When you pass the logical ID of this resource to the intrinsic `Ref` function, `Ref` returns the resource name. For example:

```
{ "Ref": "MyResource" }
```

For the resource with the logical ID `myDynamoDBTable`, `Ref` will return the DynamoDB table name.

For more information about using the `Ref` function, see Ref.

Mappings

```
Mappings:
  Mapping01:
    Key01:
      Name:Value01
    Key02:
      Name:Value02
    Key03:
      Name:Value03
```

Mappings

```
Mappings:
  RegionMap:
    us-east-1:
      HVM64: ami-0ff8a91507f77f867
      HVMG2: ami-0a584ac55a7631c0c
    us-west-2:
      HVM64: ami-0bdb828fd58c52235
      HVMG2: ami-066ee5fd4a9ef77f1
    eu-west-1:
      HVM64: ami-047bb4163c506cd98
      HVMG2: ami-0a7c483d527806435
```

FindInMap

```
!FindInMap [ MapName, TopLevelKey, SecondLevelKey ]
```

FindInMap

```
AWSTemplateFormatVersion: "2010-09-09"
Mappings:
  RegionMap:
    us-east-1:
      HVM64: ami-0ff8a91507f77f867
      HVMG2: ami-0a584ac55a7631c0c
    us-west-2:
      HVM64: ami-0bdb828fd58c52235
      HVMG2: ami-066ee5fd4a9ef77f1
    eu-west-1:
      HVM64: ami-047bb4163c506cd98
      HVMG2: ami-0a7c483d527806435
Resources:
  myEC2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId: !FindInMap [RegionMap, !Ref "AWS::Region", HVMG2]
      InstanceType: m1.small
```

GetAtt

```
!GetAtt logicalNameOfResource.attributeName
```

GetAtt

```yaml
myELB:
  Type: AWS::ElasticLoadBalancing::LoadBalancer
  Properties:
    AvailabilityZones:
      - eu-west-1a
    Listeners:
      - LoadBalancerPort: '80'
        InstancePort: '80'
        Protocol: HTTP

myELBIngressGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: ELB ingress group
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: '80'
        ToPort: '80'
        SourceSecurityGroupOwnerId: !GetAtt myELB.SourceSecurityGroup.OwnerAlias
        SourceSecurityGroupName: !GetAtt myELB.SourceSecurityGroup.GroupName
```

GetAtt

AWS::ElasticLoadBalancing::LoadBalancer

Filter View: All

Fn::GetAtt

The `Fn::GetAtt` intrinsic function returns a value for a specified attribute of this type. The following are the available attributes and sample return values.

For more information about using the `Fn::GetAtt` intrinsic function, see Fn::GetAtt.

`CanonicalHostedZoneName`

The name of the Route 53 hosted zone that is associated with the load balancer. Internal-facing load balancers don't use this value, use `DNSName` instead.

`CanonicalHostedZoneNameID`

The ID of the Route 53 hosted zone name that is associated with the load balancer.

`DNSName`

The DNS name for the load balancer.

`SourceSecurityGroup.GroupName`

The name of the security group that you can use as part of your inbound rules for your load balancer's back-end instances.

`SourceSecurityGroup.OwnerAlias`

The owner of the source security group.

Join

```
!Join: [ delimiter, [ comma-delimited list of values ] ]
```

Join

```
EC2:
  Type: "AWS::EC2::Instance"
  Properties:
    ImageId: ami-xxxxxxx
    InstanceType: t2-micro

Outputs:
  wpadmin:
    Description: WP Admin Login URL
    Value:
      !Join ["", ["http://", !GetAtt EC2.PublicIp, "/wordpress/wp-login.php"]]
```

conditional functions

```
!Equals [value_1, value_2]

!Not [condition]

!And [condition,…]

!Or [condition,…]

!If [condition_name, value_if_true, value_if_false]
```

Equals, Not

```
Conditions:
  isProd: !Equals [!Ref AccountType, "prod"]
  isNotProd: !Not [!Equals [!Ref AccountType, "prod"]]

EC2:
  Type: AWS::EC2::Instance
  Condition: isProd

Lambda:
  Type: AWS::Lambda::Function
  Condition: isNotProd
```

And, Or

```
MyAndCondition: !And
  - !Equals ["sg-mysggroup", !Ref ASecurityGroup]
  - !Condition SomeOtherCondition

MyOrCondition:
  !Or [
    !Equals [sg-mysggroup, !Ref ASecurityGroup],
    Condition: SomeOtherCondition,
  ]
```

If

```
SecurityGroups:
  - !If [CreateNewSecurityGroup, !Ref NewSecurityGroup, !Ref ExistingSecurityGroup]
```

If cont'd

```
Parameters:
  SnapshotId:
    Type: String
    Default: ""
    Description: Enter snapshop Id to restore

Conditions:
  isRestore: !Not [!Equals [!Ref SnapshotId, ""]]

DB:
  Type: "AWS::RDS::DBInstance"
  DeletionPolicy: Snapshot
  Properties:
    AllocatedStorage: 5
    StorageType: gp2
    DBInstanceClass: !FindInMap [InstanceSize, !Ref EnvironmentSize, DB]
    DBName: !If [isRestore, !Ref "AWS::NoValue", !Ref DatabaseName]
    Engine: MySQL
    MasterUsername: !If [isRestore, !Ref "AWS::NoValue", !Ref DatabaseUser]
    MasterUserPassword:
      !If [isRestore, !Ref "AWS::NoValue", !Ref DatabasePassword]
    DBSnapshotIdentifier: !Ref SnapToRestore
```

configuration management

user data
cfn-init
cfn-signal
cfn-hup

quick demo

resources

CloudFormation Docs
https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/Welcome.html

Introduction to CloudFormation (A Cloud Guru, 2.5 hrs)
https://acloud.guru/learn/aws-cloudformation

Advanced CloudFormation (A Cloud Guru, 12 hrs)
https://acloud.guru/learn/aws-advanced-cloudformation

Managing Multi-Account AWS Environments Using AWS Organizations
AWS re:Inforce 2019 (FND314)
https://youtu.be/fxo67UeeN1A

Presentation Material
https://github.com/davetownsend/presentations/tree/master/2019/IaC

@davetownsend