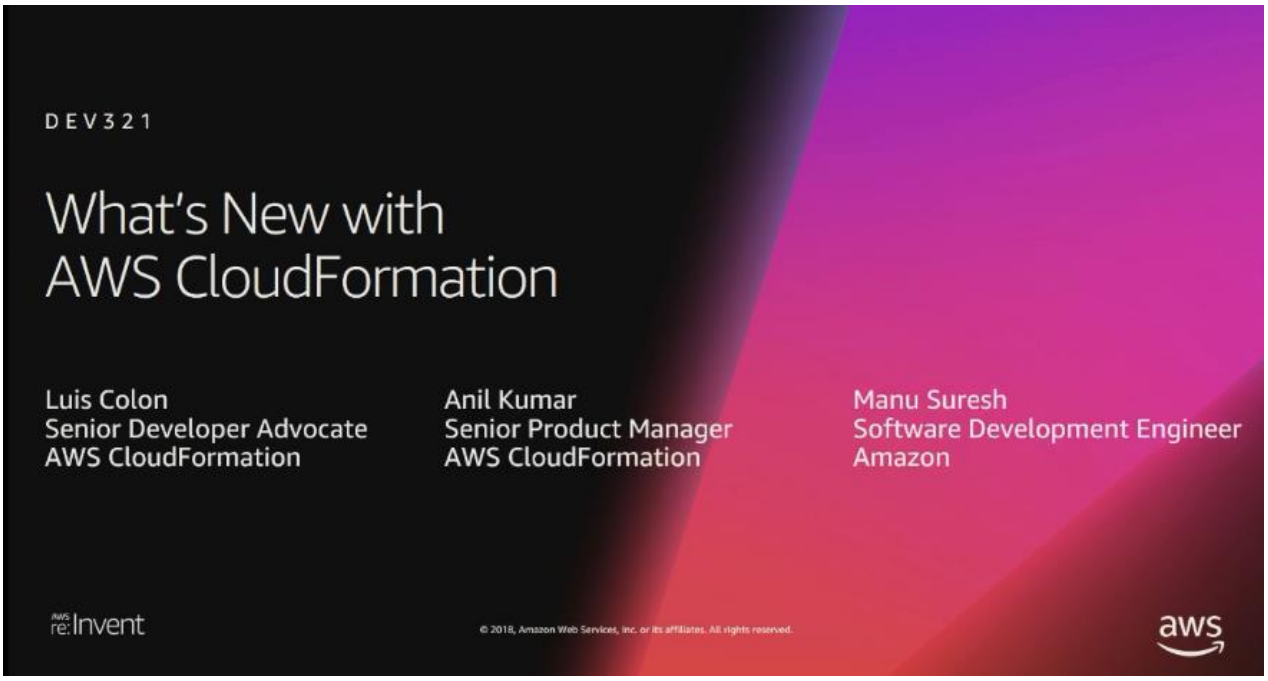
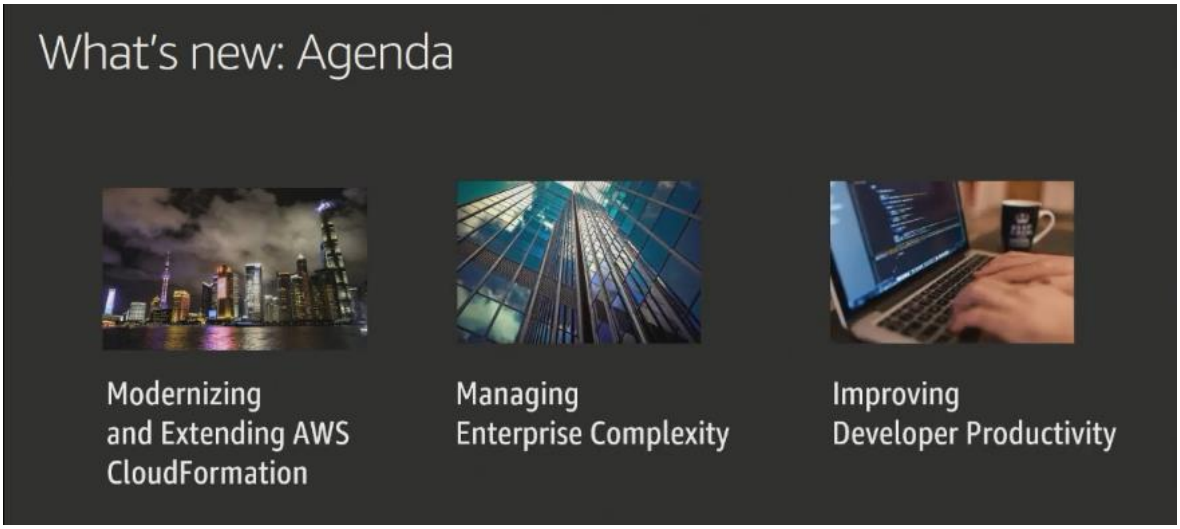


AWS CloudFormation is one of the most widely used tools in the AWS ecosystem, enabling infrastructure as code, deployment automation, repeatability, compliance and standardization. In this session, we cover the latest improvements and best practices for AWS CloudFormation customers in particular, and for seasoned infrastructure engineers in general. We cover new features and improvements that span many use cases, including programmability options, cross region and cross account automation, operational safety, and additional integration with many other AWS services.



AWS CloudFormation 101

Code in YAML or JSON directly or use sample templates

Upload local files or from an S3 bucket

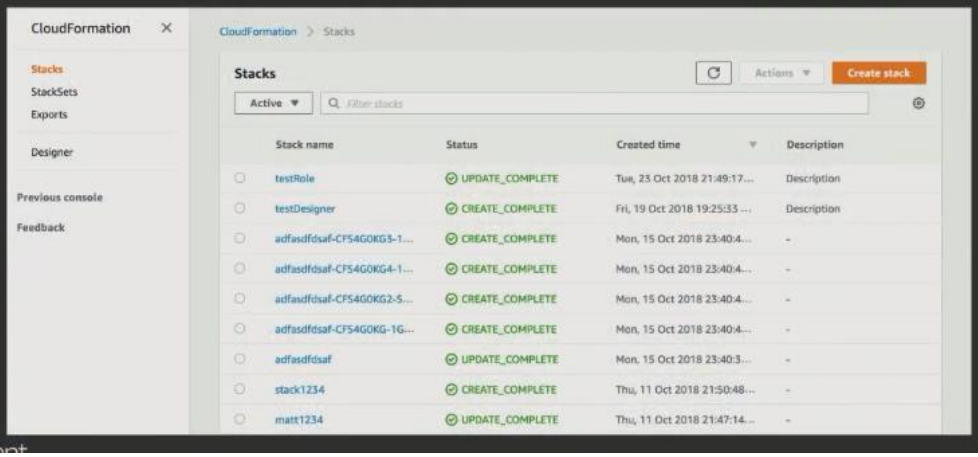
Create stack using console, API or CLI

Stacks and resources are provisioned

Modernizing and Extending AWS CloudFormation

Redesigned user experience

Redesigned AWS CloudFormation console

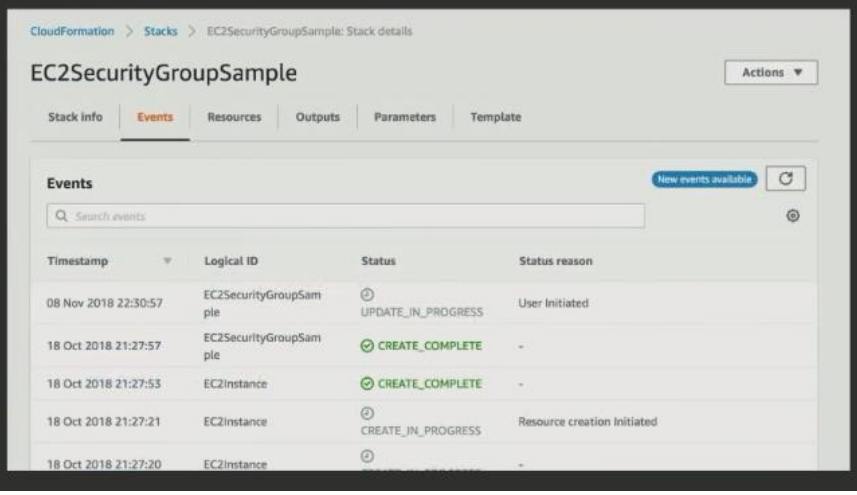


Redesigned user experience

WCAG 2.0 AA accessibility standards

Responsive across various screen sizes

New flexible base for new features

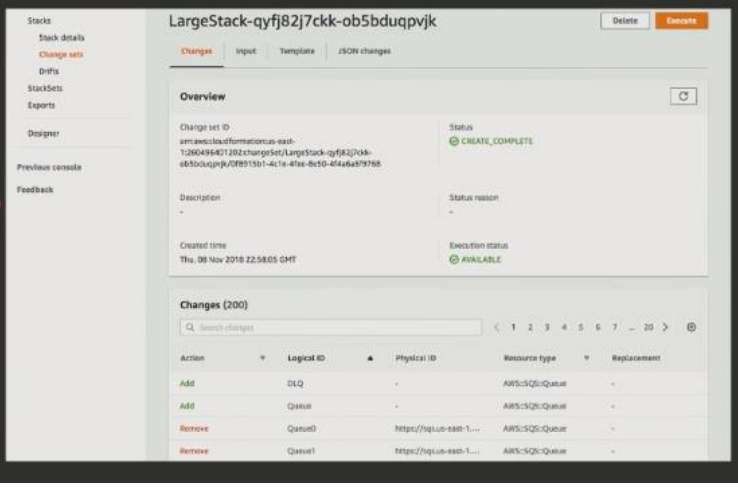


Redesigned user experience

Great feedback from thousands of early users

Switch back and forth

Looking ahead: Help us drive improvements by continuing to provide comments



Resource types support

Over 300 resource types supported

Added support for over 65 new resource types year-to-date

Amazon API Gateway
Amazon CloudWatch Events
Amazon Elastic Cloud Compute (Amazon EC2)
Amazon GuardDuty
Amazon MQ
Amazon Neptune

Amazon SageMaker
Amazon Simple Email Service (Amazon SES)
Amazon AppStream
AWS AppSync
AWS Budgets
AWS CodePipeline

AWS Config
AWS IoT 1-Click
AWS Secrets Manager
AWS Service Catalog
AWS Systems Manager
AWS Application Auto Scaling

Extending AWS CloudFormation to support more…

Custom Resources

Use the AWS::CloudFormation::CustomResource or Custom::String resource type to define custom resources in your templates

Include resources that aren't available as AWS CloudFormation resource types

What about native support for non-AWS resource types?

Alexa::ASK::Skill - this resource creates an Alexa skill that enables customers to access new abilities

Send us your feedback (@anilsdomain) on what other types you would like us to support

Extending AWS CloudFormation to support more…

Use Infrastructure as code techniques to build and operate Amazon Alexa skills

Use AWS CloudFormation features to create and update skills on the Amazon Alexa side

Access other related AWS resources to support Amazon Alexa skills

Easily iterate to enhance Amazon Alexa skills by modifying the templates and updating AWS CloudFormation stacks

Extending AWS CloudFormation to support more…

Example: "Hello World" Amazon Alexa skill using Alexa::ASK::Skill resource type

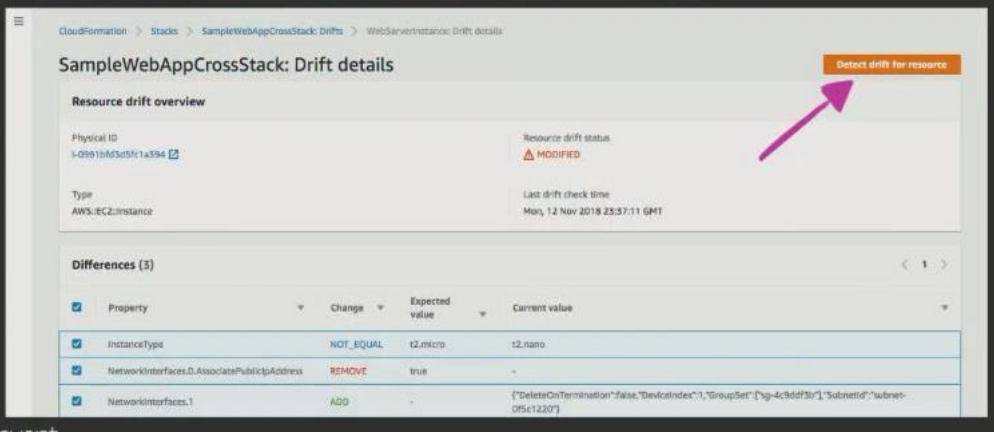
```
---
Resources:
  LambdaExecutionRole:
    Type: "AWS::IAM::Role"
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Principal:
              Service: lambda.amazonaws.com
            Action: "sts:AssumeRole"
  SkillFunction:
    Type: "AWS::Lambda::Function"
    Properties:
      Handler: "com.amazon.ask.helloworld.HelloWorldStreamHandler"
      Role: !GetAtt LambdaExecutionRole.Arn
      Code:
        S3Bucket: "ask-java-sample-stacks"
        S3Key: "helloworld-1.0-jar-with-dependencies.jar"
      Runtime: "java8"
      MemorySize: 512
      Timeout: 60
  AlexaSkillFuncAlexaSkillEventPermission:
    Type: "AWS::Lambda::Permission"
    Properties:
      Action: "lambda:invokeFunction"
      FunctionName: !Ref SkillFunction
      Principal: "alexa-appkit.amazon.com"
  HelloWorldSkill:
    Type: "Alexa::ASK::Skill"
    Properties:
      SkillPackage:
        S3Bucket: "ask-java-sample-stacks"
        S3Key: "helloworld.zip"
        Overrides:
          Manifest:
            apis:
              custom:
                endpoint:
                  uri: !GetAtt SkillFunction.Arn
      AuthenticationConfiguration:
        ClientId: ""
        ClientSecret: ""
        RefreshToken: ""
      VendorId: ""
```

There are 4 resource types in this template here, 3 native AWS types and a custom Alexa type.

Managing enterprise complexity

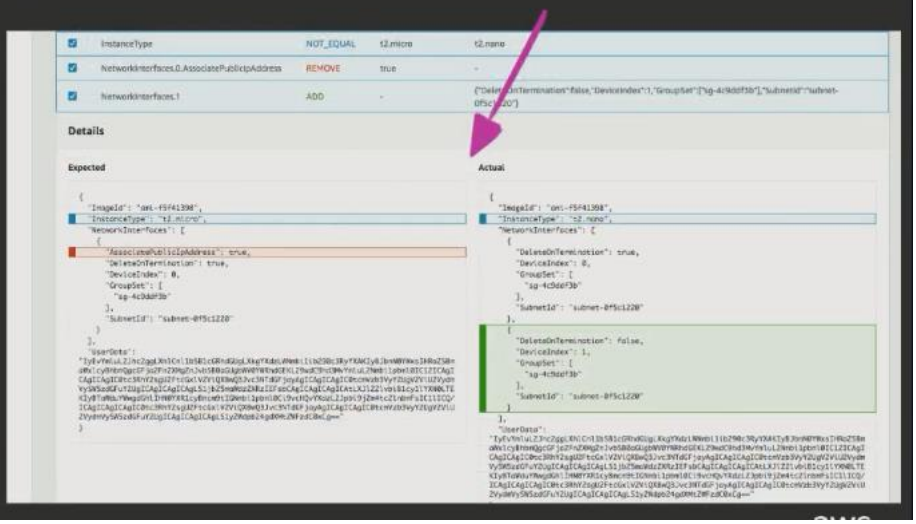
Drift detection

Allows you to detect if configuration changes were made to your stack resources outside of AWS CloudFormation via the AWS Management Console, CLI, and SDKs



Drift Detection

Use the diff viewer in the console to pinpoint the changes



Drift Detection

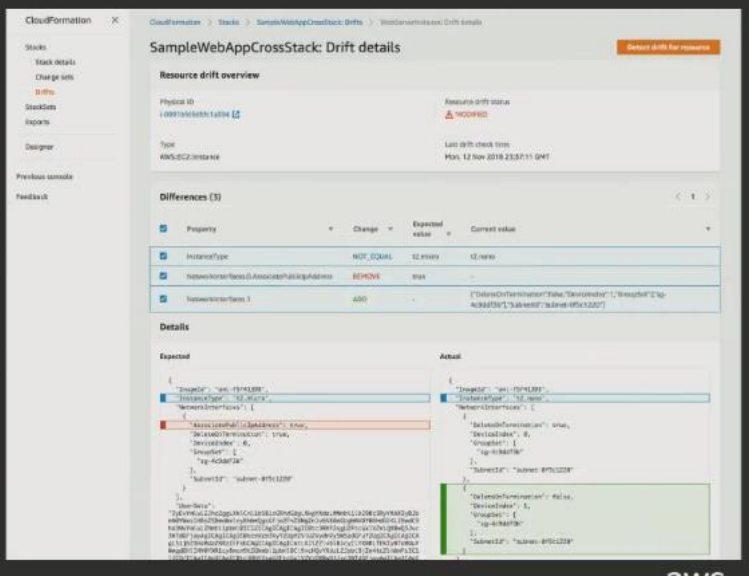
Also available via CLI and API

Supports the most commonly used resources

Automatic drift alerts via AWS Config rule

Remediate by updating live configuration values to match the template values

Looking ahead: supporting more resources, preventing false positives, handling edge cases - help us by providing feedback!



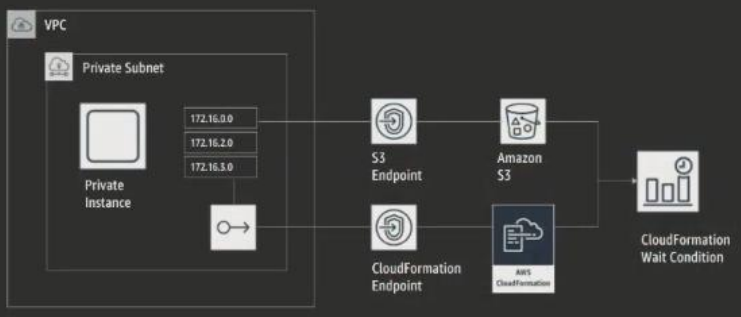
AWS PrivateLink support

AWS PrivateLink is a purpose-built technology designed to access AWS services, while keeping all the network traffic within the AWS network

Use AWS CloudFormation APIs inside of your VPC and route data between your VPC and AWS CloudFormation entirely within the AWS network.

No proxies, NATs, or Internet Gateways required

Improve security posture. E.g. sending a signal back to AWS CloudFormation stack from within a private VPC without going across the public internet



New Improvements for StackSets

StackSets extends the functionality of stacks by enabling you to create, update, or delete stacks across multiple accounts and regions with a single operation.

Custom execution roles and administrator roles

More fine-grained control of stack instance updates

Extended limits

Example: Temporary Accounts

Fast cross-account changes with less code

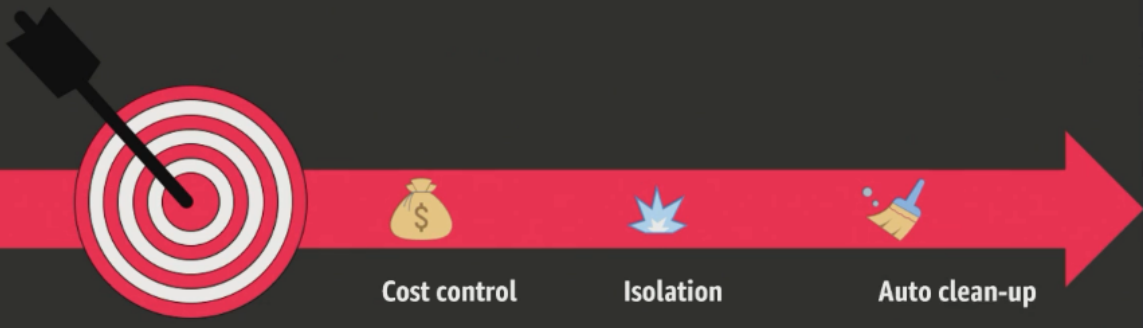
100
Average number of short-lived accounts created daily

375
Highest number of short-lived accounts created in a day

$1,000
Maximum budget allowed in a short-lived AWS account

2
Number of active short-lived accounts that a developer can have at a time

Sandbox environments

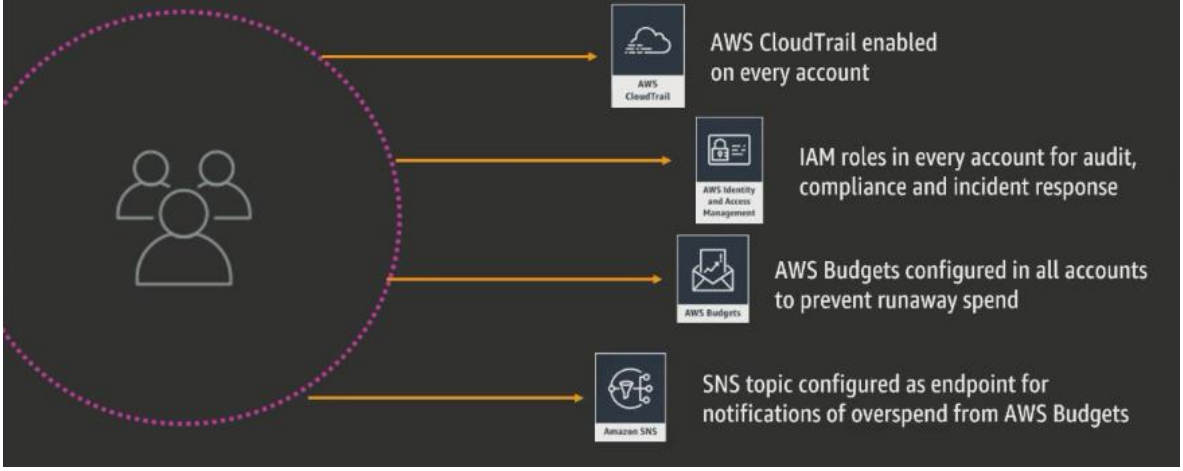


Cost control
Enforce budget limits on cost of resources for experiments

Isolation
Experiments cannot run in the same environment used for production

Auto clean-up
Resources must be cleaned up after experiments are complete

Short-lived account policy requirements



AWS CloudTrail enabled on every account

IAM roles in every account for audit, compliance and incident response

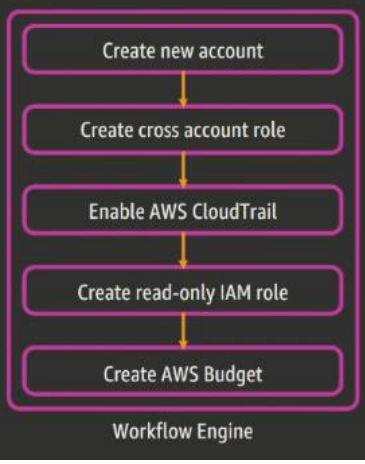
AWS Budgets configured in all accounts to prevent runaway spend

SNS topic configured as endpoint for notifications of overspend from AWS Budgets

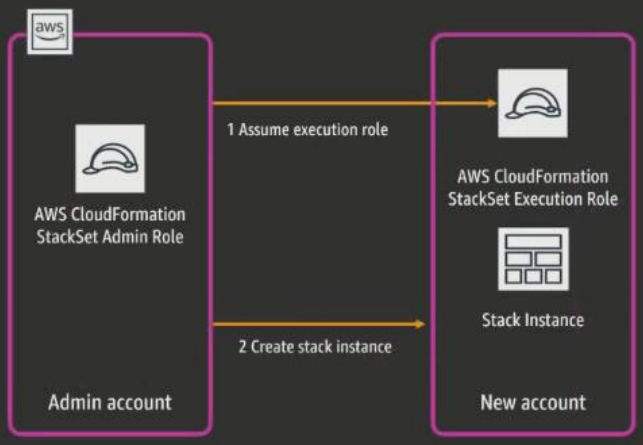
Before: Account creation without Stacksets

Provisioning each resource required to meet policy requirements involves:

- Create a step in the account creation workflow
- Assume cross-account role on newly created account
- Call AWS SDK APIs to provision the resource



After: Account creation with Stacksets



Each stack instance configures all the following resources in new accounts:

- AWS CloudTrail
- Cross-account IAM Roles
- AWS Budgets

Before: Code (1/4)

```
public class AwsBudgetsOperations {
private static final Log LOG = LogFactory.getLog(AwsBudgetsOperations.class);
private static final String BUDGET_NAME = "BurnerBudget";
private static final BigDecimal BUDGET_AMOUNT = new BigDecimal(1000.0D);
private static final String BUDGET_UNIT = "USD";
private static final String TIME_PERIOD = "MONTHLY";
private static final String BURNER_SNS_REGION = "us-east-1";
private static final String BURNER_SNS_TOPIC = "DO-NOT-UNSUBSCRIBE-BurnerBudgetAlarm";
private static final String GREATER_THAN_COMPARATOR = "GREATER_THAN";
private static final String EQUAL_TO_COMPARATOR = "EQUAL_TO";
private static final String NOTIFICATION_TYPE = "ACTUAL";
private static final String THRESHOLD_TYPE = "PERCENTAGE";
private static final String SUBSCRIPTION_TYPE_SNS = "SNS";
private AwsBudgetsClientFactory budgetsClientFactory;

public AwsBudgetsOperations() {
}
public CreateBudgetResult createBudget(String awsAccountId) {
    String burnerBudgetAlarmSnsTopic = String.format("arn:aws:sns:%s:%s:%s", "us-east-1", awsAccountId, "DO-NOT-UNSUBSCRIBE-BurnerBudgetAlarm");
    CreateBudgetRequest createBudgetRequest = this.createBudgetRequest(awsAccountId, "GREATER_THAN", 100.0D, "SNS", burnerBudgetAlarmSnsTopic);
    CreateBudgetResult result = this.getBudgetsClientFactory(awsAccountId).createBudget(createBudgetRequest);
    return result;
}
private AWSBudgets getBudgetsClientFactory(String awsAccountId) {
    return this.budgetsClientFactory.getAwsBudgetsClient(awsAccountId);
}
```

Before: Code (2/4)

```
@VisibleForTesting
CreateBudgetRequest createBudgetRequest(String awsAccountId, String comparatorOperator, Double threshold, String snsSubscriptionType, String snsSubscriptionAddress) {
    Budget budget = new Budget();
    budget.setBudgetName("BurnerBudget");
    Spend spend = new Spend();
    spend.setAmount(BUDGET_AMOUNT);
    spend.setUnit("USD");
    budget.setBudgetLimit(spend);
    budget.setBudgetType(BudgetType.COST);
    CostTypes costTypes = new CostTypes();
    costTypes.setIncludeTax(true);
    costTypes.setIncludeSubscription(true);
    costTypes.setUseBlended(false);
    budget.setCostTypes(costTypes);
    TimePeriod timePeriod = new TimePeriod();
    timePeriod.setStart(new Date(Instant.now().toEpochMilli()));
    timePeriod.setEnd(new Date(Instant.now().plus(30L, ChronoUnit.DAYS).toEpochMilli()));
    budget.setTimePeriod(timePeriod);
    budget.setTimeUnit("MONTHLY");
    Notification notification = this.createNotification(awsAccountId, comparatorOperator, threshold);
    Subscriber snsSubscriber = this.createSubscriber(snsSubscriptionType, snsSubscriptionAddress);
    NotificationWithSubscribers notificationWithSubscribers = (new NotificationWithSubscribers()).withNotification(notification).withSubscribers(new Subscriber[]{snsSubscriber});
    CreateBudgetRequest request = new CreateBudgetRequest();
    request.setAccountId(awsAccountId);
    request.setBudget(budget);
    request.setNotificationsWithSubscribers(Lists.newArrayList(new NotificationWithSubscribers[]{notificationWithSubscribers}));
    return request;
}
```

Before: Code (3/4)

```
@VisibleForTesting
Notification createNotification(String awsAccountId, String comparatorOperator, Double threshold) {
    Notification notification = new Notification();
    notification.setComparisonOperator(comparatorOperator);
    notification.setNotificationType("ACTUAL");
    notification.setThreshold(threshold);
    notification.setThresholdType("PERCENTAGE");
    return notification;
}
@VisibleForTesting
Subscriber createSubscriber(String subscriptionType, String subscriptionAddress) {
    Subscriber subscriber = new Subscriber();
    subscriber.setAddress(subscriptionAddress);
    subscriber.setSubscriptionType(subscriptionType);
    return subscriber;
}
public AwsBudgetsClientFactory getBudgetsClientFactory() {
    return this.budgetsClientFactory;
}
public void setBudgetsClientFactory(AwsBudgetsClientFactory budgetsClientFactory) {
    this.budgetsClientFactory = budgetsClientFactory;
}
public class AwsBudgetsClientFactory {
private static final Log LOG = LogFactory.getLog(AwsBudgetsClientFactory.class);
private static final Integer SESSION_DURATION_SECONDS = 900;
private static final String EXTERNAL_ID_PREFIX = "BurnerAccessExtId-";
private static final Map<String, String> ROLE_ARN_PREFIX = AppConfig.findMap("RoleArnPrefix");
private static final String ROLE_ARN_SUFFIX = ":role/BurnerConsoleAccessClientRole-DO-NOT-DELETE";
private static final String CLASSIC_PARTITION = "classic";
private static final String AWS_BUDGETS_MATERIAL_SET = AppConfig.findString("OdinMaterialSetNameAwsBudgets");
private static final String SESSION_NAME = "AwsBudgetsOperation";
```

Before: Code (4/4)

```
public AwsBudgetsClientFactory() {
}
public AWSBudgets getAwsBudgetsClient(String awsAccountId) {
    String externalId = this.getExternalID(awsAccountId);
    String roleArn = this.getRoleARN(awsAccountId, "classic");
    AssumeRoleRequest assumeRoleRequest = (new
AssumeRoleRequest()).withDurationSeconds(SESSION_DURATION_SECONDS).withExternalId(externalId).withRoleArn(roleArn).withRoleSessionName("AwsBudgetsOperation");
    AWSSecurityTokenServiceClient stsClient = this.configureSTSClient(credentialsProvider);
    AssumeRoleResult assumeRoleResult = stsClient.assumeRole(assumeRoleRequest);
    BasicSessionCredentials credentials = new BasicSessionCredentials(assumeRoleResult.getCredentials().getAccessKeyId(),
assumeRoleResult.getCredentials().getSecretAccessKey(), assumeRoleResult.getCredentials().getSessionToken());
    AWSBudgetsClient budgetsClient = new AWSBudgetsClient(credentials);
    return budgetsClient;
}
private String getRoleARN(String awsAccountID, String partition) {
    return (String)ROLE_ARN_PREFIX.get(partition.toLowerCase()) + awsAccountID + ":role/BurnerConsoleAccessClientRole-DO-NOT-DELETE";
}
private String getExternalID(String awsAccountID) {
    return "BurnerAccessExtId-" + awsAccountID;
}
private AWSSecurityTokenServiceClient configureSTSClient(AWSCredentialsProvider credentialsProvider) {
    AWSSecurityTokenServiceClient stsClient = new AWSSecurityTokenServiceClient(credentialsProvider);
    return stsClient;
}
```

After: AWS CloudFormation Code

```
"BurnerBudget": {
    "Type": "AWS::Budgets::Budget",
    "Properties": {
        "Budget": {
            "BudgetLimit": {
                "Amount": "100",
                "Unit": "USD"
            },
            "TimeUnit": "MONTHLY",
            "TimePeriod": {
                "Start": { "Ref" : "BudgetStartDate" },
                "End": { "Ref" :  "BudgetEndDate" }
            },
            "BudgetType": "COST"
        },
        "NotificationsWithSubscribers": [
            {
                "Notification": {
                    "NotificationType": "ACTUAL",
                    "ComparisonOperator": "GREATER_THAN",
                    "Threshold": 99
                },
                "Subscribers": [
                    {
                        "SubscriptionType": "SNS",
                        "Address": { "Fn::Join" : [ ":", ["arn:aws:sns:us-east-1", { "Ref" : "AccountId" }, "BurnerBudgetAlarm-1" ] ] }
                    }
                ]
            }
        ]
    },
    "DependsOn": "BudgetSNSTopic"
}
```

Developer productivity gains

3x faster to deploy a new AWS resource when policy requirements changes

8x less code when provisioning through StackSet deployments

Improved handling of secrets

Improve your infrastructure code by extracting global configuration details

Last year: Support for Parameter Store String and String List Parameters

This year: Parameter Store Secure String and Secrets Manager support

Dynamic references for quick retrieval, used on these parameter types

Improved handling of secrets

```
Parameters:
  InstanceType:
    Type: 'AWS::SSM::Parameter::Value<String>'
    Default: ssbEC2iDev
  KeyName:
    Type: 'AWS::EC2::KeyPair::KeyName'
    Default: brinks
  LatestAmiId:
    Type: 'AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>'
    Default: '/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2'
Resources:
  PSBInstance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: !Ref LatestAmiId
      KeyName: !Ref KeyName
      InstanceType: !Ref InstanceType
```

This is a typical CF template, we are declaring 2 parameters here

```
Parameters:
  InstanceType:
    Type: 'AWS::SSM::Parameter::Value<String>'
    Default: ssbEC2iDev
  KeyName:
    Type: 'AWS::EC2::KeyPair::KeyName'
    Default: brinks
  LatestAmiId:
    Type: 'AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>'
    Default: '/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2'
Resources:
  PSBInstance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: !Ref LatestAmiId
      KeyName: !Ref KeyName
      InstanceType: !Ref InstanceType
```

We now remove the need for the parts highlighted

New dynamic references

```
Parameters:
  KeyName:
    Type: 'AWS::EC2::KeyPair::KeyName'
    Default: brinks
  LatestAmiId:
    Type: 'AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>'
    Default: '/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2'
Resources:
  PSBInstance:
    Type: 'AWS::EC2::Instance'
    Properties:
      ImageId: !Ref LatestAmiId
      KeyName: !Ref KeyName
      InstanceType: '{{resolve:ssm:ssbEC2iDev:1}}'
```

This is what we now have

New dynamic references & secure strings

```
Resources:
  MyRDSDB:
    Type: "AWS::RDS::DBInstance"
    Properties:
      DBInstanceClass: db.t2.medium
      AllocatedStorage: '20'
      Engine: mariadb
      EngineVersion: '10.2'
      MasterUsername: appadmin
      MasterUserPassword: ch4ng1ng-s3cr3t

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}}'
```

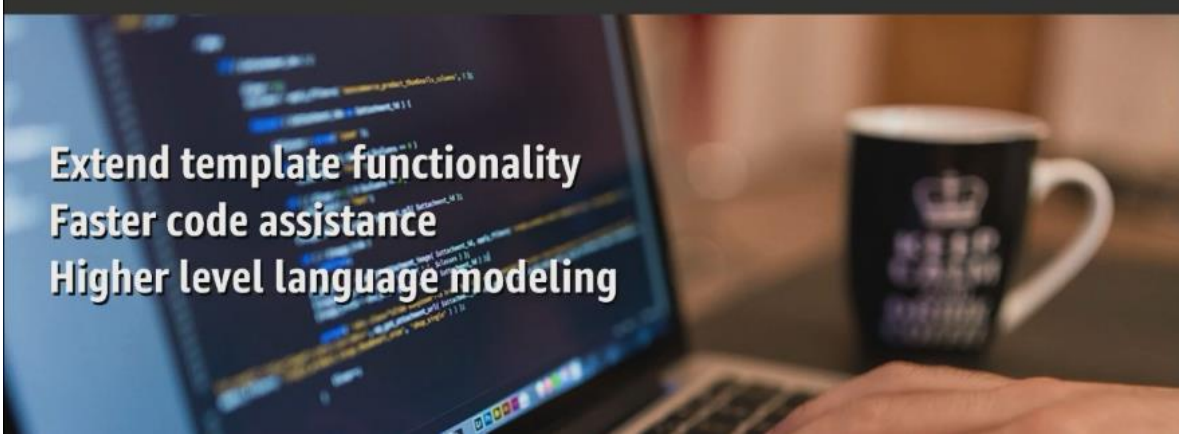
We can replace the password with the secure strings in the lower part

New dynamic references & secrets manager

```
Resources:
  MyRDSDB:
    Type: "AWS::RDS::DBInstance"
    Properties:
      DBInstanceClass: db.t2.medium
      AllocatedStorage: '20'
      Engine: mariadb
      EngineVersion: '10.2'
      MasterUsername: appadmin
      MasterUserPassword: ch4ng1ng-s3cr3t

Resources:
  MyRDSDB:
    Type: "AWS::RDS::DBInstance"
    Properties:
      DBInstanceClass: db.t2.medium
      AllocatedStorage: '20'
      Engine: mariadb
      EngineVersion: '10.2'
      MasterUsername: '{{resolve:secretsmanager:MyRDSSecret:SecretString:username}}'
      MasterUserPassword: '{{resolve:secretsmanager:MyRDSSecret:SecretString:password}}'
```

We can also use the secrets manager as above

Improving developer productivity



New AWS CloudFormation Macros

Enables template coders to write short-hand, abbreviated instructions that expand automatically when deployed

- Add utility functions, iteration/looping, strings, and others
- Ensure resources are defined to comply to your standards
- Easy to share and reuse across stacks

Iterator Macro

```
Transform:
  - Count
Resources:
  Bucket:
    Type: AWS::S3::Bucket
    Count: 3
```

```
Resources:
  Bucket1:
    Type: AWS::S3::Bucket
  Bucket2:
    Type: AWS::S3::Bucket
  Bucket3:
    Type: AWS::S3::Bucket
```

```
Transform:
  - Count
  Sqs:
    Type: AWS:::SQS::Queue
    Count: 2
```

```
Resources:
  Sqs1:
    Type: AWS:::SQS::Queue
  Sqs2:
    Type: AWS:::SQS::Queue
```

Making your Macro

```
import copy

def process_template(template):
    new_template = copy.deepcopy(template)
    status = 'success'

    for name, resource in template['Resources'].items():
        if 'Count' in resource:
            count = new_template['Resources'][name].pop('Count')
            multiplied = multiply(name, new_template['Resources'][name], count)
            if not set(multiplied.keys()) & set(new_template['Resources'].keys()):
                new_template['Resources'].update(multiplied)
            else:
                status = 'failed'
                return status, template
    return status, new_template

def multiply(resource_name, resource_structure, count):
    resources = {}
    for iteration in range(1, count):
        resources[resource_name+str(iteration)] = resource_structure
    return resources

def handler(event, context):
    result = process_template(event['fragment'])
    return {
        'requestId': event['requestId'],
        'status': result[0],
        'fragment': result[1],
    }
```

Deploy your Macro

```yaml
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31

Resources:
  MyMacro:
    Type: AWS::CloudFormation::Macro
    Properties:
      Name: Count
      FunctionName: !GetAtt CountMacroFunction.Arn
  CountMacroFunction:
    Type: AWS::Serverless::Function
    Properties:
      CodeUri: src
      Handler: index.handler
      Runtime: python3.6
      Timeout: 5
```

Macro: add string functions

```yaml
Parameters:
  InputString:
    Default: "This is a test input string"
    Type: String
Resources:
  S3Bucket:
    Type: "AWS::S3::Bucket"
    Properties:
      Tags:
        - Key: Upper
          Value:
            'Fn::Transform':
              - Name: 'StringMacro'
                Parameters:
                  InputString: !Ref InputString
                  Operation: Upper
```

```yaml
Parameters:
  InputString:
    Default: "This is a test input string"
    Type: String
Resources:
  S3Bucket:
    Type: "AWS::S3::Bucket"
    Properties:
      Tags:
        - Key: Upper
          Value: "THIS IS A TEST INPUT STRING"
```

Macro: generate additional resources

```yaml
Transform: Defaults

Resources:
  Bucket1:
    Type: AWS::S3::Bucket
```

```yaml
Resources:
  Bucket1:
    Type: AWS::S3::Bucket
    Properties:
      AccessControl: Private

  Bucket1Policy:
    Type: AWS::S3::BucketPolicy
    Properties:
      Bucket:
        Ref: Bucket1
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Deny
            Principal: "*"
            Action: "s3:Delete*"
            Resource:
              Fn::Sub: "arn:aws:s3:::${Bucket1}/*"
            Condition:
              Bool:
                aws:MultiFactorAuthPresent: "false"
```

Whenever a bucket is defined…
Add access control property
Add bucket policy
Generate additional resources, intrinsic function calls, conditions, more
Macro can allow user to override defaults

Faster Code Assistance with new AWS CloudFormation Linter

- Integrated with IDE's via plugins
- Provides the quickest feedback on code errors, warnings
- Robust validation powered by our resource specification
- Open Source - extend by building your own validation rules
- Use in headless mode for automated testing in pipelines

Fast feedback, quickly evolving

```
AWSTemplateFormatVersion: "2010-09-09"
Description: A sample template
Errors:
  Catch: Missing
Parameters:
  myParam:
    Type: String
    Default: String
    Description: String
Resources:
  ## Missing Properties
  MyEC2Instance1:
    Type: "AWS::EC2::Instance1"
  ## Fake Properties Key on main level
  ## Bad sub properties in BlockDeviceMappings/Ebs and NetworkInterfaces
  MyEC2Instance:
    Type: "AWS::EC2::Instance"
    Properties:
      ImageId: "ami-2f726546"
      InstanceType: t1.micro
      KeyName: 1
      FakeKey: MadeYouLook
      BlockDeviceMappings:
        -
```

Severity	Provider	Description	Line
Warning	Cfn-Lint	Top level item Errors isn't valid	3:1
Warning	Cfn-Lint	Parameter myParam not used	6:1
Warning	Cfn-Lint	Invalid Type AWS::EC2::Instance1 for resource MyEC2Instance1	12:1
Warning	Cfn-Lint	Properties not defined for resource MyEC2Instance1	12:1
Warning	Cfn-Lint	Invalid Property FakeKey for resource MyEC2Instance	18:1
Warning	Cfn-Lint	Invalid Property BadSubX2Key for resource MyEC2Instance	26:1

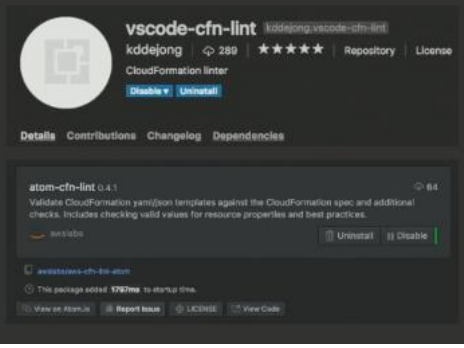
- Process multiple files
- Better handling of Conditions/Fn::If
- SAM Local integration for SAM templates
- CloudFormation limit checks
- Service rules for Route53 and CodePipeline

This is a screenshot of the linter in VSCode that shows you issues where they exist in your code

Setup, plugins, stats

https://github.com/awslabs/cfn-python-lint

```
pip install cfn-lint
cfn-lint -t simple-vpc.yaml
```



GitHub

306 Stars
290 PRs merged
153 Issues closed
61 Forks

Current release: v.0.9.1

Pypi

6,905 installs this week
23,232 installs this month
55,871 installs since release

Higher level language modeling

When doing Infrastructure as code with AWS CloudFormation, you use declarative syntax

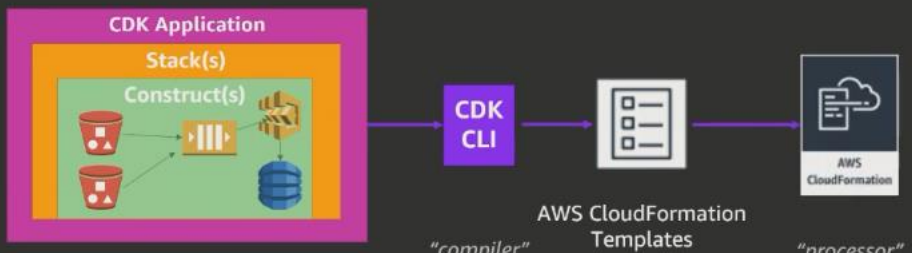
- Provides a level of abstraction above API calls or CLI commands
- More approachable for coders with less high-level language experience…
- …at the expense of not providing full language constructs, like object oriented functions (macros close the gap in some cases)

However, domain-specific, higher level language options also exist

- More opportunities for coders of JS/TS, Python, Ruby, and others
- More opportunities to componentize and abstract, which can make reuse easier

Infrastructure as Code coding options are expanding

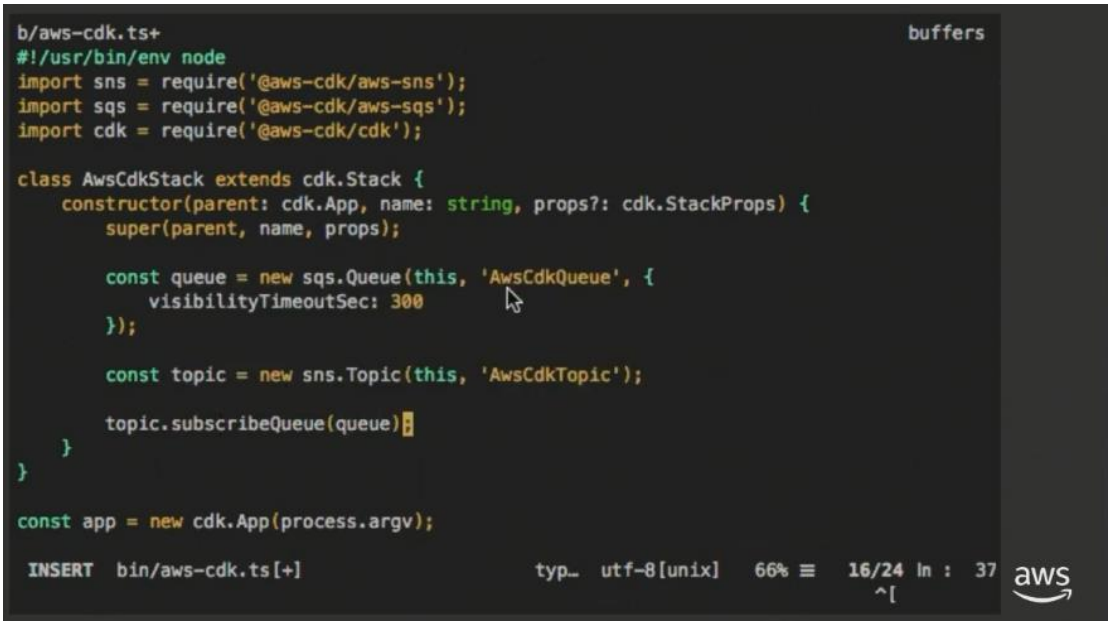
AWS Cloud Development Kit (CDK) in Dev Preview

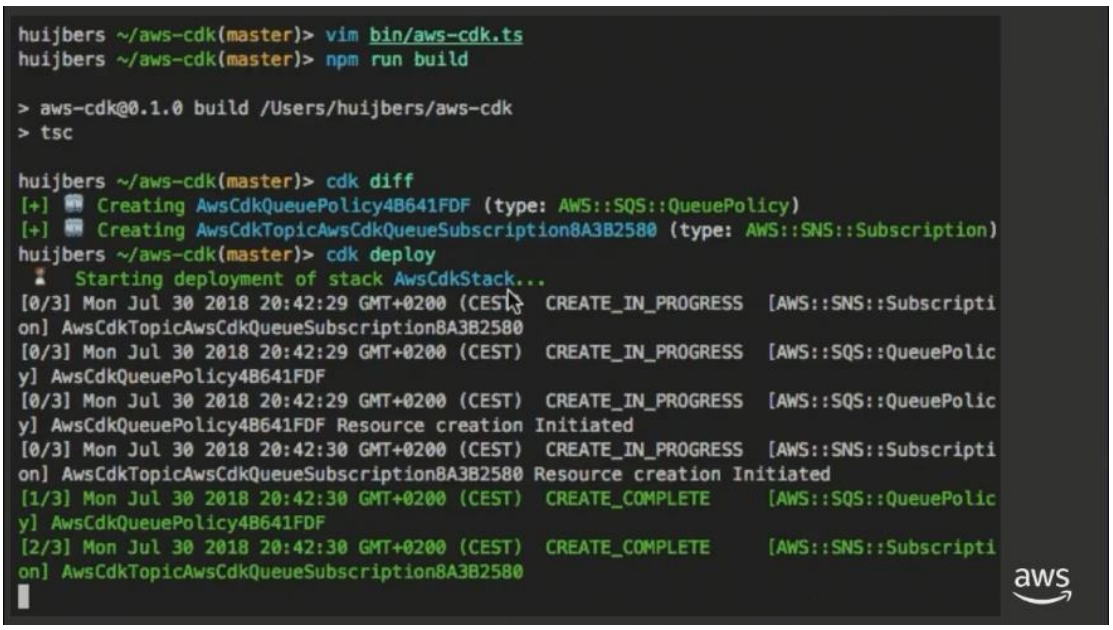


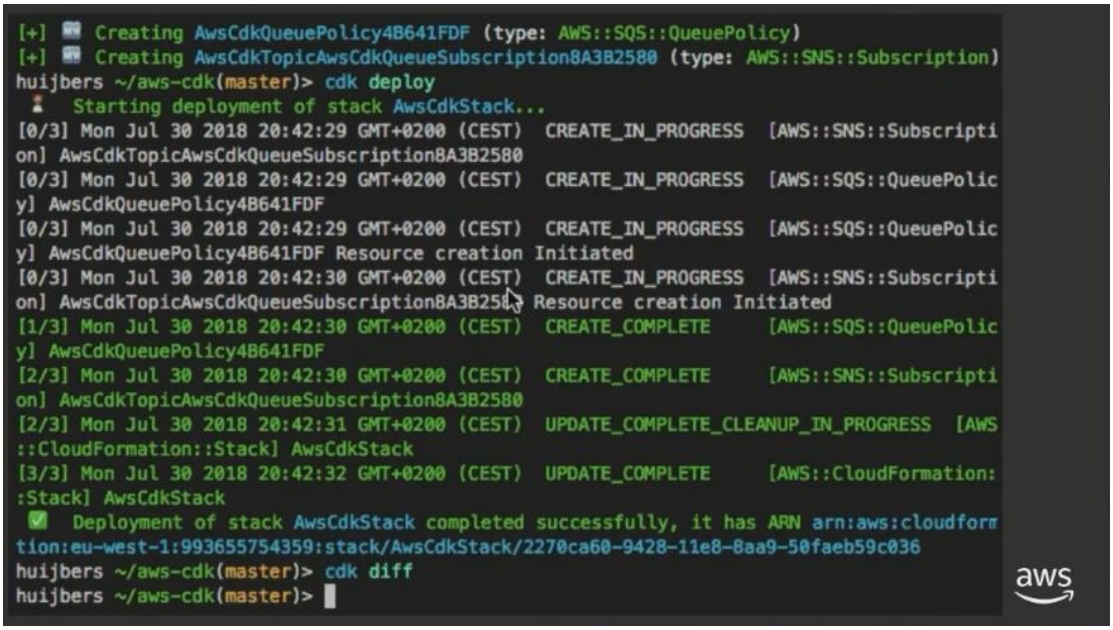
```
huijbers ~/aws-cdk> npm install -g aws-cdk
/usr/local/bin/cdk -> /usr/local/lib/node_modules/aws-cdk/bin/cdk
+ aws-cdk@0.7.4-beta
updated 1 package in 16.203s
huijbers ~/aws-cdk> cdk init app --language typescript
Initializing a new git repository...
Applying project template app for typescript
Executing npm install...
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN aws-cdk@0.1.0 No repository field.
npm WARN aws-cdk@0.1.0 No license field.

# Useful commands
 * `npm run build`   compile typescript to js
 * `npm run watch`   watch for changes and compile
 * `cdk deploy`      deploy this stack to your default AWS account/region
 * `cdk diff`        compare deployed stack with current state
 * `cdk synth`       emits the synthesized CloudFormation template

huijbers ~/aws-cdk(master)>
```

```
# Useful commands
 * `npm run build`   compile typescript to js
 * `npm run watch`   watch for changes and compile
 * `cdk deploy`      deploy this stack to your default AWS account/region
 * `cdk diff`        compare deployed stack with current state
 * `cdk synth`       emits the synthesized CloudFormation template

huijbers ~/aws-cdk(master)> npm run build

> aws-cdk@0.1.0 build /Users/huijbers/aws-cdk
> tsc

huijbers ~/aws-cdk(master)> cdk deploy
  Starting deployment of stack AwsCdkStack...
[0/2] Mon Jul 30 2018 20:41:15 GMT+0200 (CEST)  CREATE_IN_PROGRESS  [AWS::CloudFormation:
:WaitConditionHandle] WaitCondition
[0/2] Mon Jul 30 2018 20:41:15 GMT+0200 (CEST)  CREATE_IN_PROGRESS  [AWS::CloudFormation:
:WaitConditionHandle] WaitCondition Resource creation Initiated
[1/2] Mon Jul 30 2018 20:41:16 GMT+0200 (CEST)  CREATE_COMPLETE     [AWS::CloudFormation:
:WaitConditionHandle] WaitCondition
[2/2] Mon Jul 30 2018 20:41:17 GMT+0200 (CEST)  CREATE_COMPLETE     [AWS::CloudFormation:
:Stack] AwsCdkStack
```

```
[0/5] Mon Jul 30 2018 20:41:27 GMT+0200 (CEST)  CREATE_IN_PROGRESS  [AWS::SNS::Topic] Aws
CdkTopicF164620F Resource creation Initiated
[0/5] Mon Jul 30 2018 20:41:27 GMT+0200 (CEST)  CREATE_IN_PROGRESS  [AWS::SQS::Queue] Aws
CdkQueue7B79C8BE Resource creation Initiated
[1/5] Mon Jul 30 2018 20:41:28 GMT+0200 (CEST)  CREATE_COMPLETE     [AWS::SQS::Queue] Aws
CdkQueue7B79C8BE
[1/5] Mon Jul 30 2018 20:41:29 GMT+0200 (CEST)  CREATE_IN_PROGRESS  [AWS::CDK::Metadata]
CDKMetadata Resource creation Initiated
[2/5] Mon Jul 30 2018 20:41:30 GMT+0200 (CEST)  CREATE_COMPLETE     [AWS::CDK::Metadata]
CDKMetadata
[3/5] Mon Jul 30 2018 20:41:38 GMT+0200 (CEST)  CREATE_COMPLETE     [AWS::SNS::Topic] Aws
CdkTopicF164620F
[3/5] Mon Jul 30 2018 20:41:40 GMT+0200 (CEST)  UPDATE_COMPLETE_CLEANUP_IN_PROGRESS  [AWS
::CloudFormation::Stack] AwsCdkStack
[3/5] Mon Jul 30 2018 20:41:42 GMT+0200 (CEST)  DELETE_IN_PROGRESS  [AWS::CloudFormation:
:WaitConditionHandle] WaitCondition
[4/5] Mon Jul 30 2018 20:41:42 GMT+0200 (CEST)  DELETE_COMPLETE     [AWS::CloudFormation:
:WaitConditionHandle] WaitCondition
[5/5] Mon Jul 30 2018 20:41:43 GMT+0200 (CEST)  UPDATE_COMPLETE     [AWS::CloudFormation:
:Stack] AwsCdkStack
  Deployment of stack AwsCdkStack completed successfully, it has ARN arn:aws:cloudform
tion:eu-west-1:993655754359:stack/AwsCdkStack/2270ca60-9428-11e8-8aa9-50faeb59c036
huijbers ~/aws-cdk(master)> vim bin/aws-cdk.ts
```


b/aws-cdk.ts+ buffers
#!/usr/bin/env node
import sns = require('@aws-cdk/aws-sns');
import sqs = require('@aws-cdk/aws-sqs');
import cdk = require('@aws-cdk/cdk');
class AwsCdkStack extends cdk.Stack {
constructor(parent: cdk.App, name: string, props?: cdk.StackProps) {
super(parent, name, props);
const queue = new sqs.Queue(this, 'AwsCdkQueue', {
visibilityTimeoutSec: 300
});
const topic = new sns.Topic(this, 'AwsCdkTopic');
topic.subscribeQueue(queue);
}
}
const app = new cdk.App(process.argv);
INSERT bin/aws-cdk.ts[+] typ… utf-8[unix] 66% ≡ 16/24 ln : 37
^[
aws


huijbers ~/aws-cdk(master)> vim bin/aws-cdk.ts
huijbers ~/aws-cdk(master)> npm run build
> aws-cdk@0.1.0 build /Users/huijbers/aws-cdk
> tsc
huijbers ~/aws-cdk(master)> cdk diff
[+] Creating AwsCdkQueuePolicy4B641FDF (type: AWS::SQS::QueuePolicy)
[+] Creating AwsCdkTopicAwsCdkQueueSubscription8A3B2580 (type: AWS::SNS::Subscription)
huijbers ~/aws-cdk(master)> cdk deploy
Starting deployment of stack AwsCdkStack...
[0/3] Mon Jul 30 2018 20:42:29 GMT+0200 (CEST) CREATE_IN_PROGRESS [AWS::SNS::Subscription] AwsCdkTopicAwsCdkQueueSubscription8A3B2580
[0/3] Mon Jul 30 2018 20:42:29 GMT+0200 (CEST) CREATE_IN_PROGRESS [AWS::SQS::QueuePolicy] AwsCdkQueuePolicy4B641FDF
[0/3] Mon Jul 30 2018 20:42:29 GMT+0200 (CEST) CREATE_IN_PROGRESS [AWS::SQS::QueuePolicy] AwsCdkQueuePolicy4B641FDF Resource creation Initiated
[0/3] Mon Jul 30 2018 20:42:30 GMT+0200 (CEST) CREATE_IN_PROGRESS [AWS::SNS::Subscription] AwsCdkTopicAwsCdkQueueSubscription8A3B2580 Resource creation Initiated
[1/3] Mon Jul 30 2018 20:42:30 GMT+0200 (CEST) CREATE_COMPLETE [AWS::SQS::QueuePolicy] AwsCdkQueuePolicy4B641FDF
[2/3] Mon Jul 30 2018 20:42:30 GMT+0200 (CEST) CREATE_COMPLETE [AWS::SNS::Subscription] AwsCdkTopicAwsCdkQueueSubscription8A3B2580
aws


[+] Creating AwsCdkQueuePolicy4B641FDF (type: AWS::SQS::QueuePolicy)
[+] Creating AwsCdkTopicAwsCdkQueueSubscription8A3B2580 (type: AWS::SNS::Subscription)
huijbers ~/aws-cdk(master)> cdk deploy
Starting deployment of stack AwsCdkStack...
[0/3] Mon Jul 30 2018 20:42:29 GMT+0200 (CEST) CREATE_IN_PROGRESS [AWS::SNS::Subscription] AwsCdkTopicAwsCdkQueueSubscription8A3B2580
[0/3] Mon Jul 30 2018 20:42:29 GMT+0200 (CEST) CREATE_IN_PROGRESS [AWS::SQS::QueuePolicy] AwsCdkQueuePolicy4B641FDF
[0/3] Mon Jul 30 2018 20:42:29 GMT+0200 (CEST) CREATE_IN_PROGRESS [AWS::SQS::QueuePolicy] AwsCdkQueuePolicy4B641FDF Resource creation Initiated
[0/3] Mon Jul 30 2018 20:42:30 GMT+0200 (CEST) CREATE_IN_PROGRESS [AWS::SNS::Subscription] AwsCdkTopicAwsCdkQueueSubscription8A3B2580 Resource creation Initiated
[1/3] Mon Jul 30 2018 20:42:30 GMT+0200 (CEST) CREATE_COMPLETE [AWS::SQS::QueuePolicy] AwsCdkQueuePolicy4B641FDF
[2/3] Mon Jul 30 2018 20:42:30 GMT+0200 (CEST) CREATE_COMPLETE [AWS::SNS::Subscription] AwsCdkTopicAwsCdkQueueSubscription8A3B2580
[2/3] Mon Jul 30 2018 20:42:31 GMT+0200 (CEST) UPDATE_COMPLETE_CLEANUP_IN_PROGRESS [AWS::CloudFormation::Stack] AwsCdkStack
[3/3] Mon Jul 30 2018 20:42:32 GMT+0200 (CEST) UPDATE_COMPLETE [AWS::CloudFormation::Stack] AwsCdkStack
Deployment of stack AwsCdkStack completed successfully, it has ARN arn:aws:cloudformation:eu-west-1:993655754359:stack/AwsCdkStack/2270ca60-9428-11e8-8aa9-50faeb59c036
huijbers ~/aws-cdk(master)> cdk diff
huijbers ~/aws-cdk(master)>
aws

Driving growth with our community

Get AWS Cloud Development Kit (CDK) Dev Preview at https://github.com/awslabs/aws-cdk

Get AWS CloudFormation linter at https://github.com/awslabs/cfn-python-lint

Expect more open source projects to improve developer experience

#CloudFormation Slack Channel over 500 collaborators in < 5 months

DM via Twitter to get added: @luiscolon1 and @anilsdomain

Expect more contributions, including macros, custom resources, utilities, samples, and more!

Summary

Modernizing and Extending CFN
- Redesigned console
- New resources

Managing Enterprise Complexity
- Drift Detection
- VPC Private Link
- Stacksets enhancements
- Secure Strings and AWS Secrets Manager

Improving Developer Productivity
- Macros
- Linter
- CDK
- Community

Thank you!

Luis Colon
Senior Developer Advocate
AWS CloudFormation

Anil Kumar
Senior Product Manager
AWS CloudFormation

Manu Suresh
Software Developer Engineer
Amazon