Why did we build AWS CloudFormation?

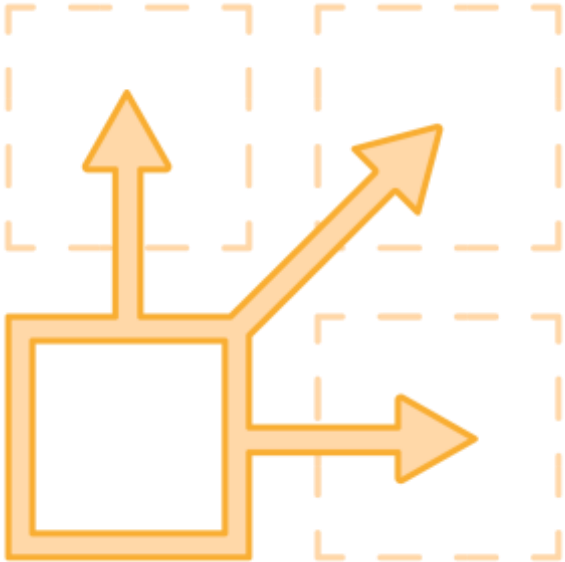
Some customer challenges

- Creating and managing AWS resources
- Provisioning and updating infrastructure resources in an orderly manner
- Version controlling infrastructure like software

What is CloudFormation?

amazon web services

Templated resource provisioning

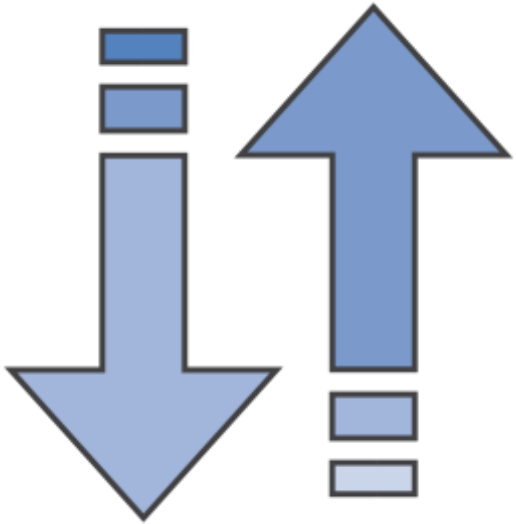


- Create templates to describe the AWS resources used to run your application
- Provision identical copies of a stack

Infrastructure as code

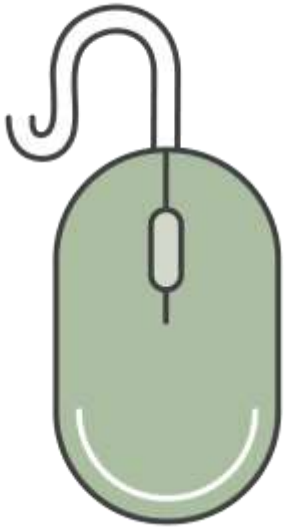
- Templates can be stored in a source control system
- Track all changes made to your infrastructure stack
- Modify and update resources in a controlled and predictable way

Declarative and flexible



- Just choose the resources and configurations you need
- Customize your template through parameters

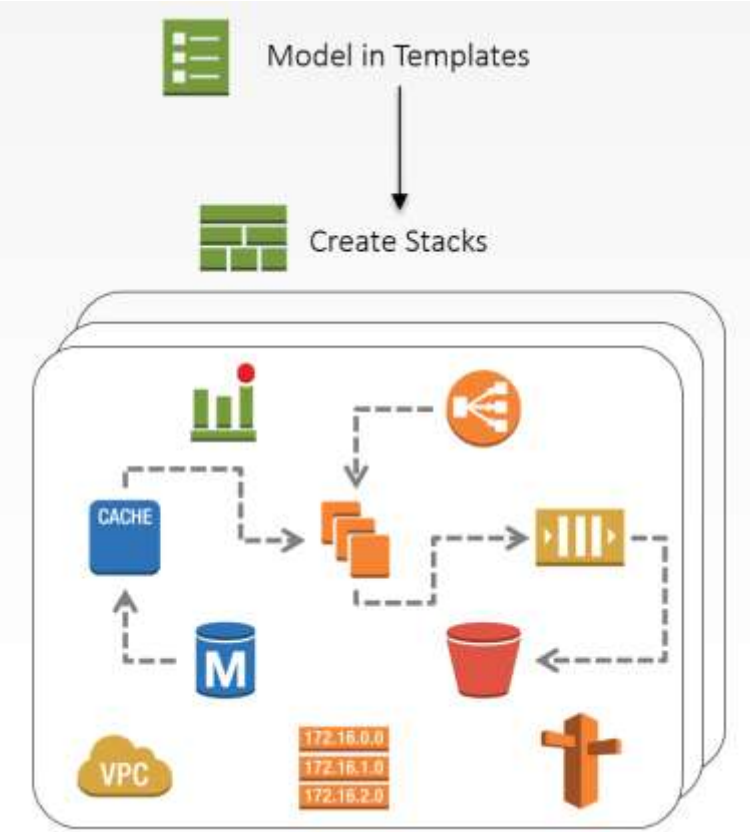
Easy to use



- Access through console, CLI, or SDKs
- Start with one of the many sample templates
- Integrate with your development and management tools

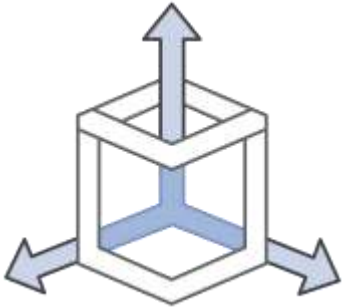
CloudFormation

- Create templates of the infrastructure and applications you want to run on AWS
- Have CloudFormation automatically provision the required AWS resources and their relationships from the templates
- Easily version, replicate, or update the infrastructure and applications using the templates
- Integrates with other development, CI/CD, and management tools

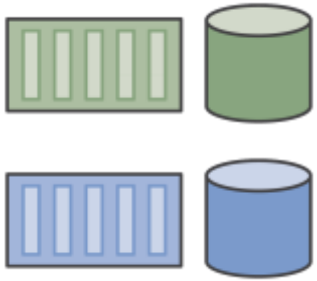


Common use cases

Stack replication



Infrastructure scale out



Blue/green deployments

Infrastructure as code

Pricing

- There is no additional charge for CloudFormation
- Customers pay only for the AWS resources (e.g., EC2 instances, EBS volumes) created using CloudFormation

Use a wide range of AWS services

- ✓ Auto Scaling
- ✓ Amazon CloudFront
- ✓ AWS CloudTrail
- ✓ Amazon CloudWatch
- ✓ Amazon DynamoDB
- ✓ Amazon EC2
- ✓ AWS Elastic Beanstalk
- ✓ Amazon ElastiCache
- ✓ Elastic Load Balancing
- ✓ AWS Identity and Access Management (IAM)
- ✓ Amazon Kinesis
- ✓ AWS OpsWorks
- ✓ Amazon RDS
- ✓ Amazon Redshift
- ✓ Amazon Route 53
- ✓ Amazon S3
- ✓ Amazon SNS
- ✓ Amazon SQS
- ✓ Amazon VPC

and more …

Frequent improvements–Posted June 9, 2016

- New resource support
  - Amazon VPC flow logs
  - Amazon Kinesis Firehose
- Updated support for existing resources
  - AWS Lambda source code
  - SNS topic updates
  - Amazon Kinesis stream names
  - New Auto Scaling update policy

For new announcements, see: https://aws.amazon.com/new/#management-tools

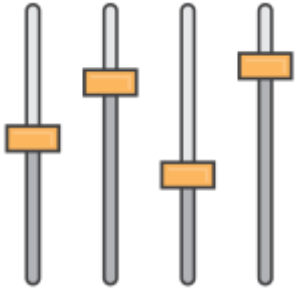
How do I get started with CloudFormation?

amazon web services

CloudFormation terminology

Stacks

Templates



Parameters

Policies

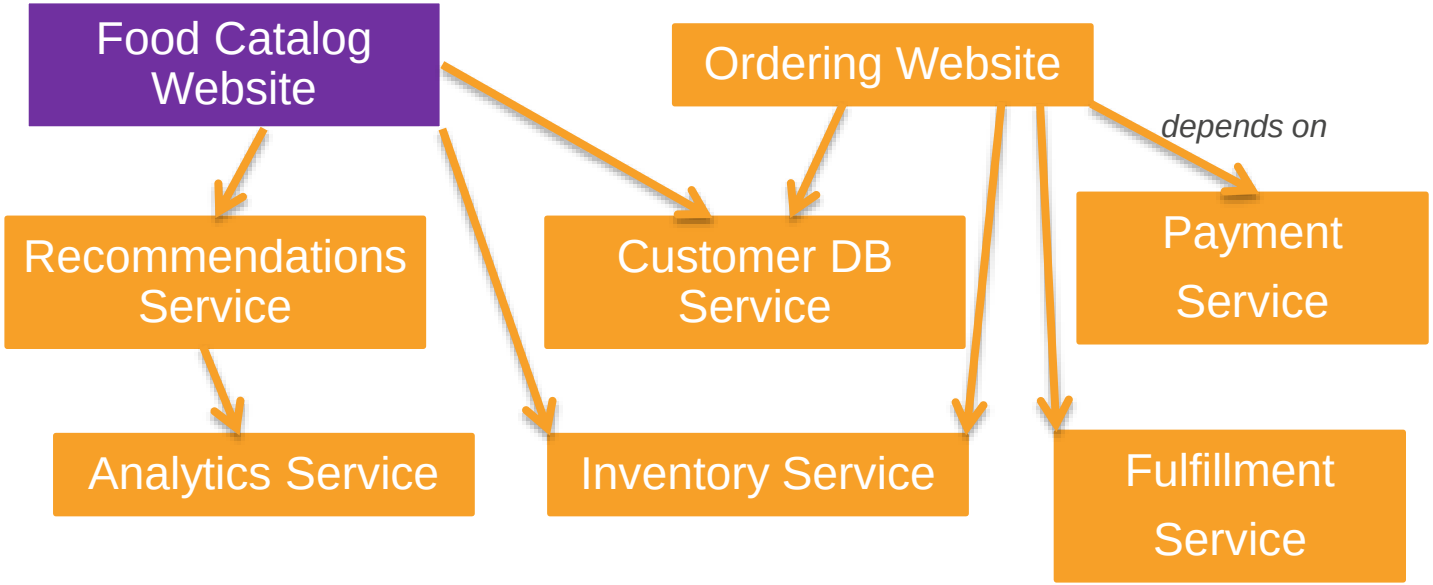
Basic steps

- Create or use an existing template
- Upload template to CloudFormation
- Specify parameter values
- Set up policies, tags, or notification options
- Review and create

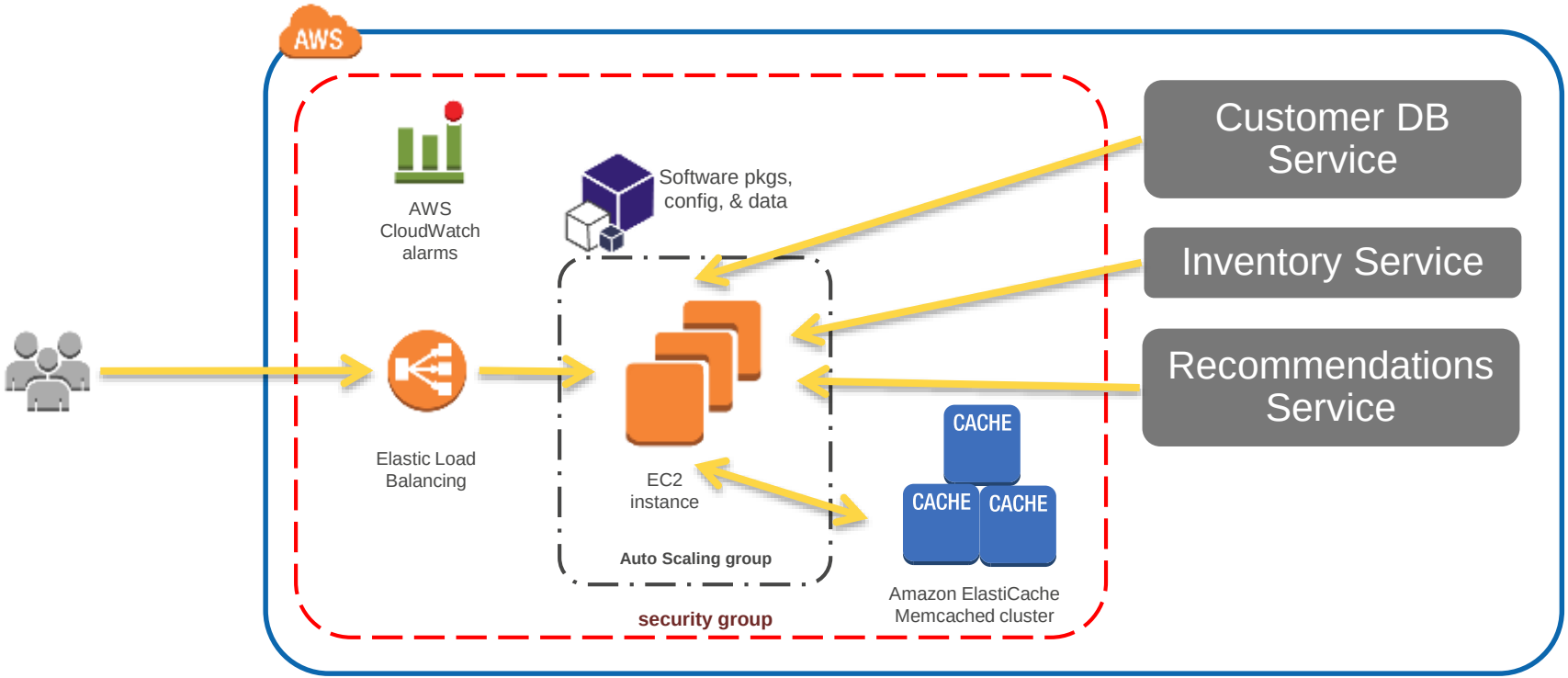
Sample Resources section of a template

```
"Resources" : {
    "EC2Instance" : {
      "Type" : "AWS::EC2::Instance",
      "Properties" : {
        "InstanceType" : { "Ref" : "InstanceType" },
        "SecurityGroups" : [ { "Ref" : "InstanceSecurityGroup" } ],
        "KeyName" : { "Ref" : "KeyName" },
        "ImageId" : { "Fn::FindInMap" : [ "AWSRegionArch2AMI", { "Ref" : "AWS::Region" },
                            { "Fn::FindInMap" : [ "AWSInstanceType2Arch", { "Ref" :
"InstanceType" }, "Arch" ] } ] }
      }
    },
```

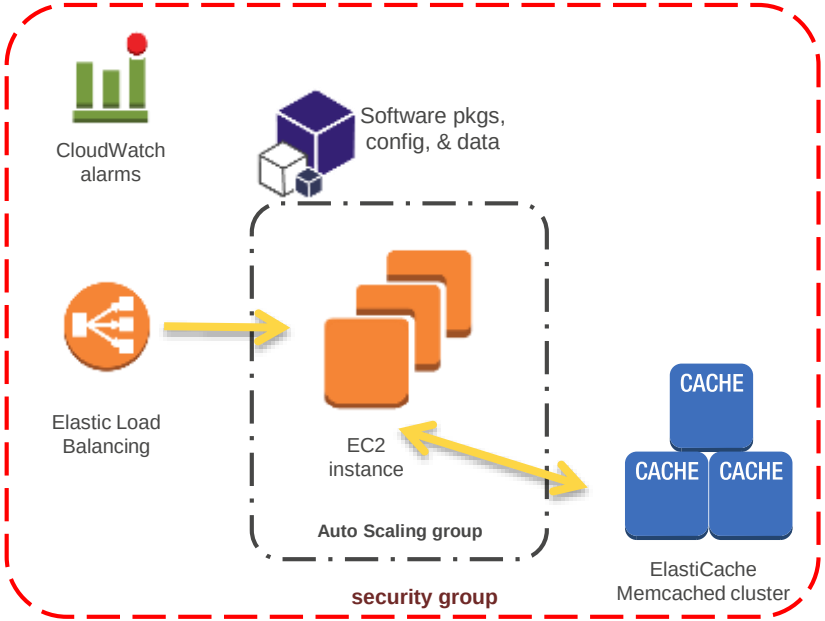
Design–Imagine building a food ordering service



Create template–example for the Food Catalog website



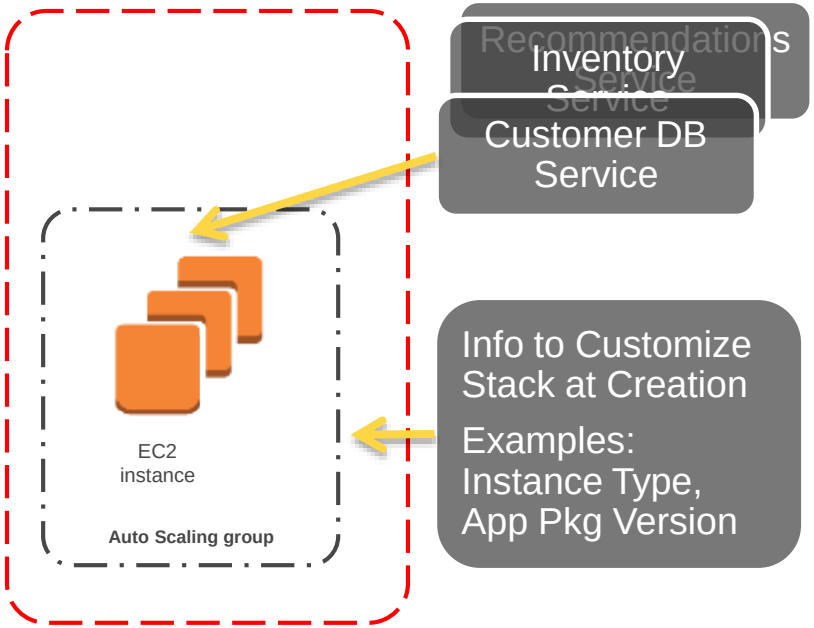
Create template–Resources



CloudFormation template

```
"Resources" : {
    "SecurityGroup" : {},
    "WebServerGroup" : {
        "Type" : "AWS::AutoScaling::AutoScalingGroup",
        "Properties" : {
            "MinSize" : "1",
            "MaxSize" : "3",
            "LoadBalancerNames" : [ { "Ref" :
"LoadBalancer" } ],
            ...
        }
    },
    "LoadBalancer" : {},
    "CacheCluster" : {},
    "Alarm" : {}
},
```

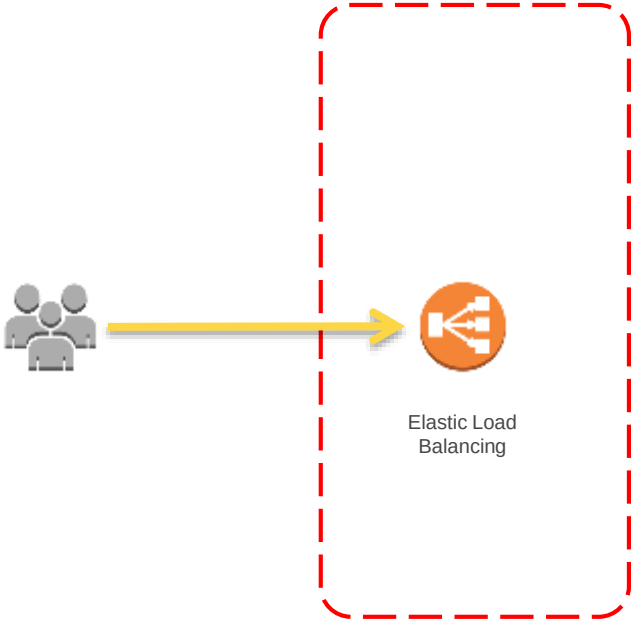
Create template–Parameters



CloudFormation template

```
"Parameters" : {
    "CustomerDBServiceEndPoint" : {
        "Description" : "URL of the Customer DB Service",
        "Type" : "String"
    },
    "CustomerDBServiceKey" : {
        "Description" : "API key for the Customer DB
Service",
        "Type" : "String",
        "NoEcho" : "true"
    },
    "InstanceType" : {
        "Description" : "WebServer EC2 instance type",
        "Type" : "String",
        "Default" : "m3.medium",
        "AllowedValues" :
["m3.medium","m3.large","m3.xlarge"],
        "ConstraintDescription" : "Must be a valid
instance type"
```

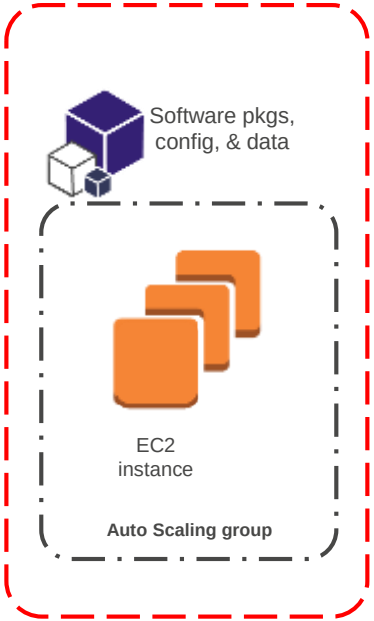
Create template–Outputs



CloudFormation template

```
"Resources" : {
    "LoadBalancer" : {},
    ...
},
"Outputs" : {
    "WebsiteDNSName" : {
        "Description" : "The DNS name of the website",
        "Value" :  {
            "Fn::GetAtt" : [ "LoadBalancer", "DNSName" ]
        }
    }
}
```

Create template–deploy and configure software

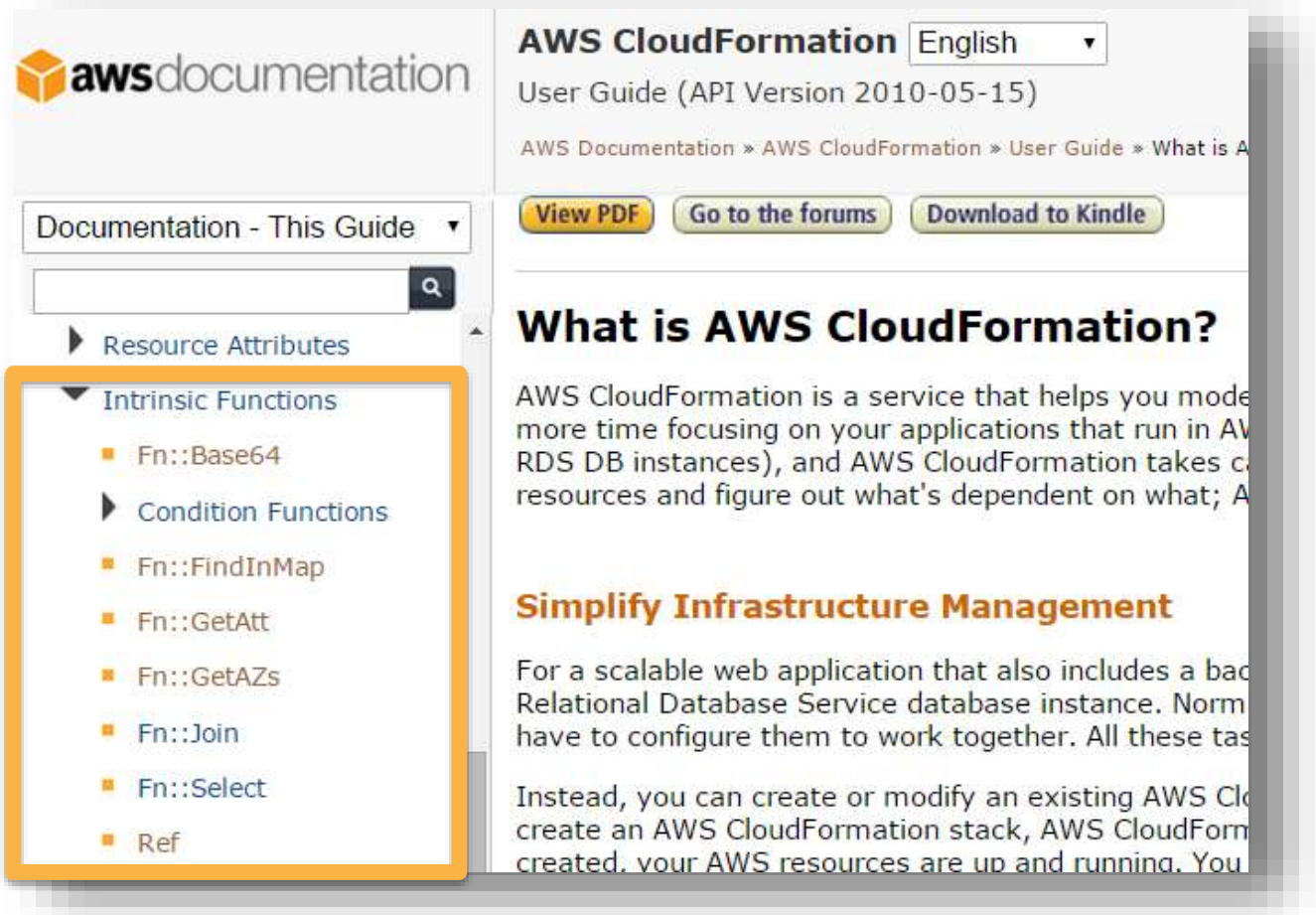


CloudFormation template

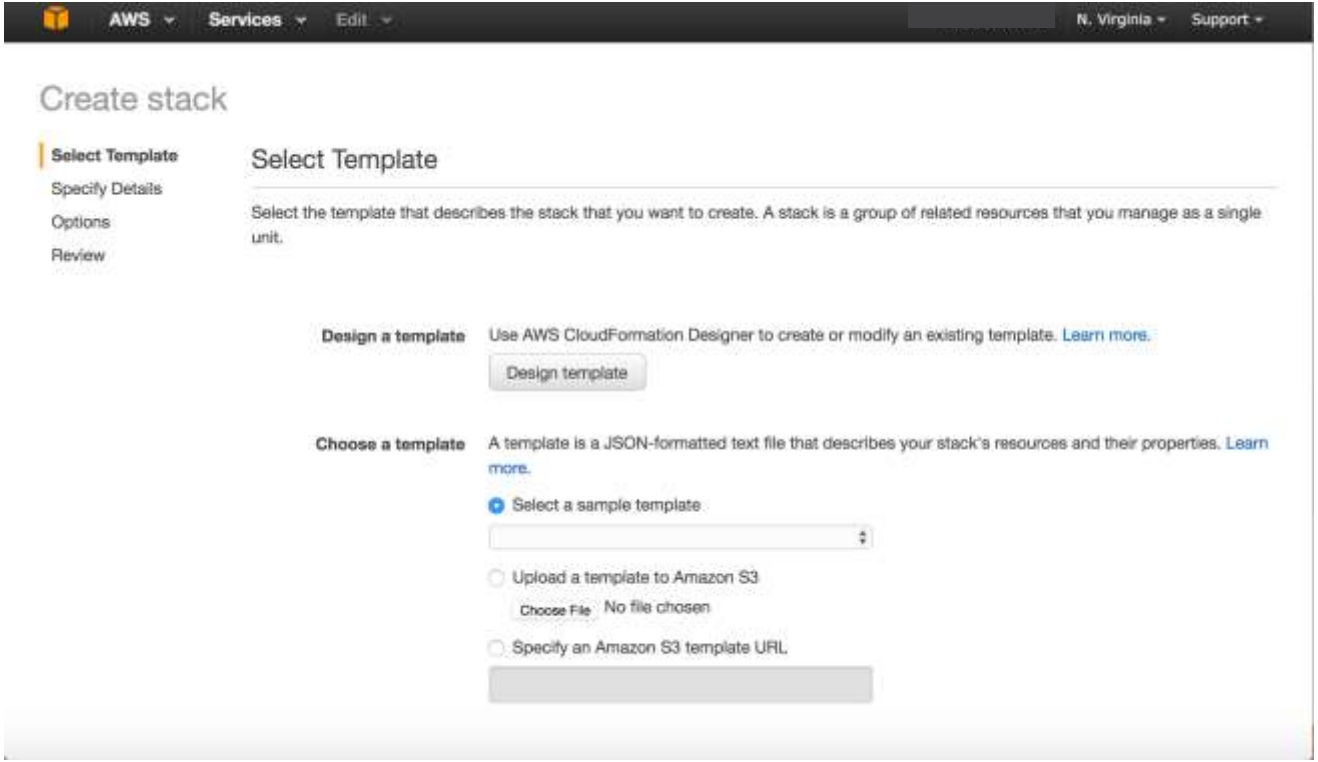
```
"AWS::CloudFormation::Init": {
    "webapp-config": {
        "packages" : {}, "sources" : {}, "files" : {},
        "groups" : {}, "users" : {},
        "commands" : {}, "services" : {}
      },

    "chef-config" : {}
}
```

- ✓ Declarative
- ✓ Debuggable
- ✓ Updatable

Create template–language features



Create stack



Operate stack

AWS Services Edit N. Virginia Support

Create Stack | Actions | Design template

Filter: Active By Name: Showing 1 stack

Stack Name	Created Time	Status	Description
My-test-stack	2016-06-08 07:30:31 UTC-0400	CREATE_COMPLETE	My test cloudformation

Overview | Outputs | Resources | Events | Template | Parameters | Tags | Stack Policy | Change Sets

Stack name: My-test-stack

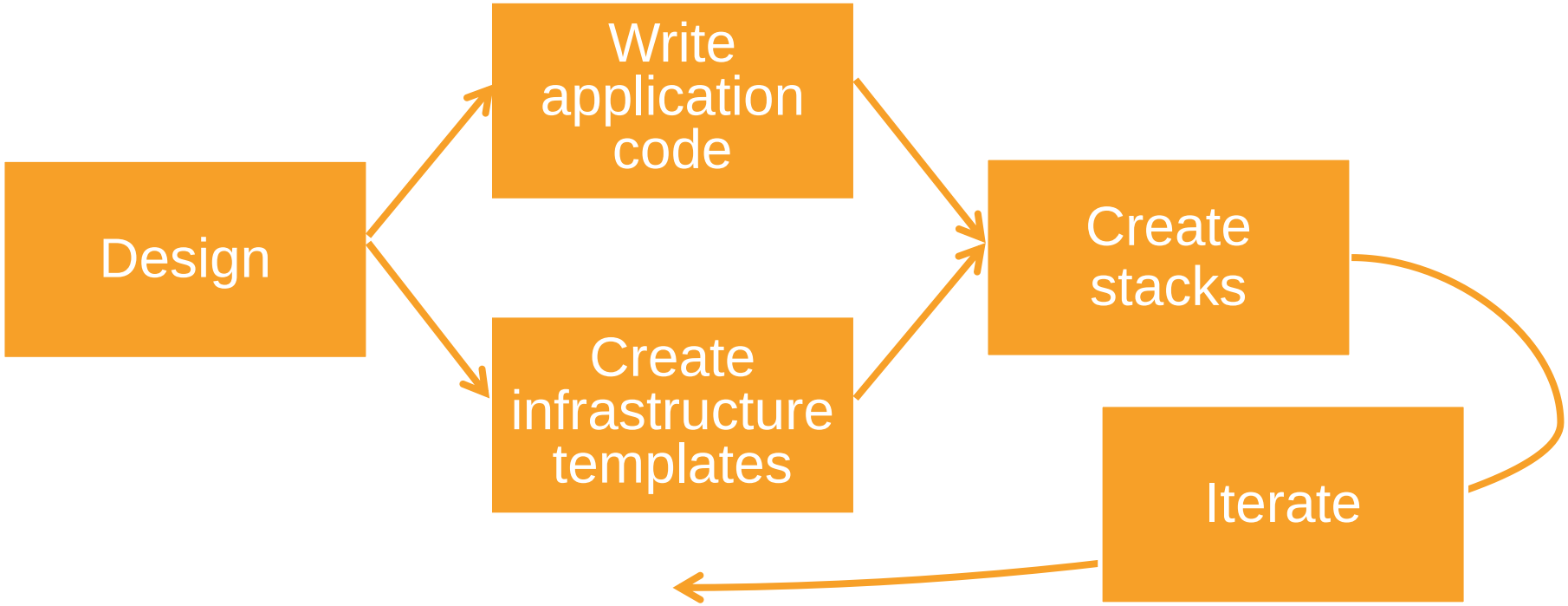
Stack ID: arn:aws:cloudformation:us-east /My-test-stack/68a15420-2d6c-11e6-801a-50d5cd26c2d2

Status: CREATE_COMPLETE

Status reason:

Description: My test cloudformation

Basic workflow



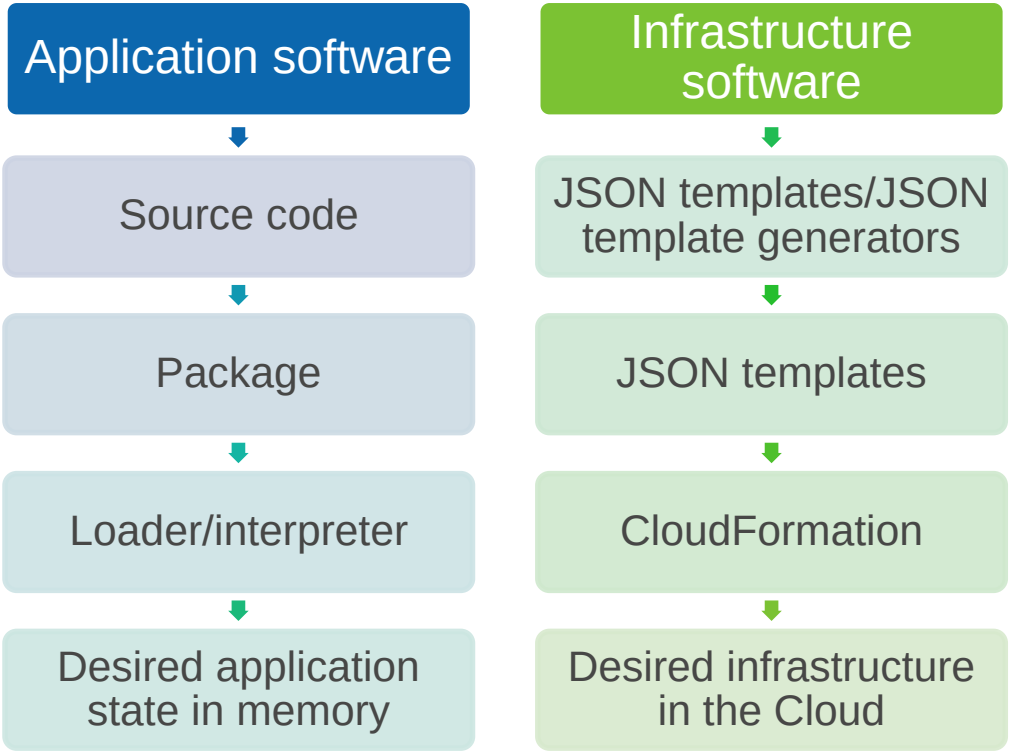
Infrastructure as code workflow

“It’s all software”

“It’s all software”–Organize like it’s software

Front-end Services	• Consumer Website, Seller Website, Mobile Backend
Back-end Services	• Search, Payments, Reviews, Recommendations
Shared Services	• CRM DBs, Common Monitoring, Alarms, Subnets, Security Groups
Base Network	• VPCs, Internet Gateways, VPNs, NATs
Identity	• IAM Users, Groups, Roles

“It’s all software”–Build and operate like it’s software



Iterate on infrastructure: Update stack

In-place

Faster

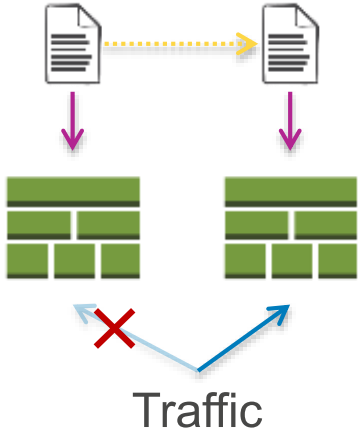
Cost-efficient

Simpler state and data migration

Templates

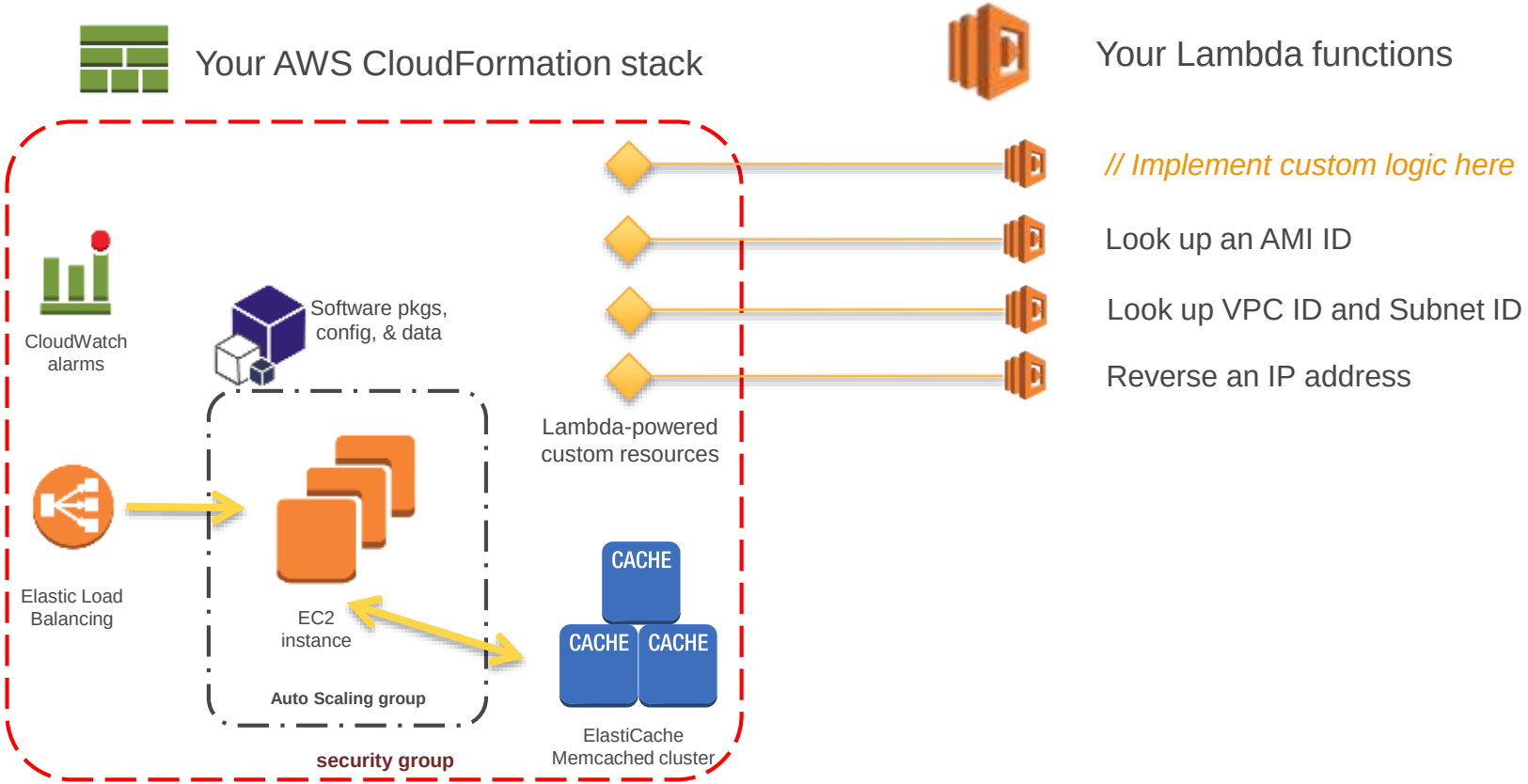
Stacks

Blue/Green

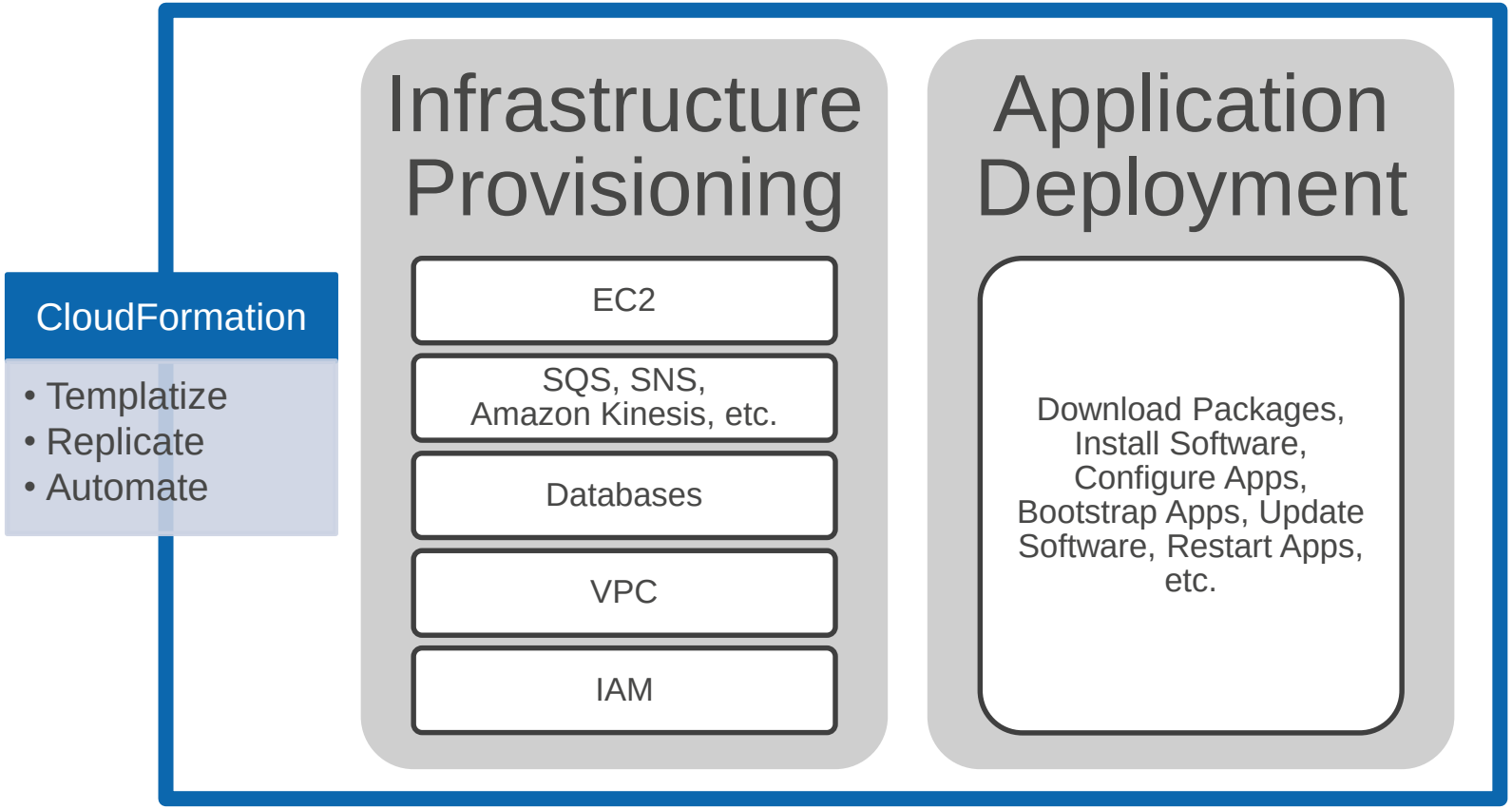


Working stack not touched

Extending CloudFormation: Lambda-backed custom resources



Application deployment as code



Best practices (1 of 2)

- Planning and organizing
  - Organize your stacks by lifecycle and ownership
  - Reuse templates to replicate stacks in multiple environments
  - Verify quotas for all resource types
  - Use nested stacks to reuse common template patterns
- Creating templates
  - Do not embed credentials in your templates
  - Use AWS-specific parameter types
  - Use parameter constraints
  - Validate templates before using them

See: http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html

Best practices (2 of 2)

- Managing stacks
  - Manage all stack resources through CloudFormation
  - Create change sets before updating your stacks
  - Use stack policies
  - Use CloudTrail to log CloudFormation calls
  - Use code reviews and revision controls to manage your templates

See: http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html