

**1. Start a EC2 Instance with Parameter KEY NAME, VPC, Subnet using Cloudformation Template.**

**2. Install Nginx in EC2 and put it in a ASG.**

**i) First by Userdata**

**ii) By Metadata**

**3. Create a Sample Index file and copy this file using MetaData into EC2 Instance**

**4. Changing the content of Index should reload the nginx config automatically in EC2 Instance**

**5. Perform ASG Rolling Update with the change in UserData in above Cloudformation Template**

**Cloudformation Tasks (Hint : use cloudformation:: init and cfn-hup)**

**REFERENCE :**

**<https://github.com/aws-labs/aws-cloudformation-templates/blob/master/aws/services/AutoScaling/AutoScalingRollingUpdates.yaml>**

**Look Instance.yml given for cloud formation refrence**

## Specify stack details

### Stack name

Stack name

CF-EC2

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

### Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

KeyName

ayush

SubnetIds

subnet-8d696de5 (172.31.0.0/20)

VPCIds

vpc-1255497a (172.31.0.0/16)

Cancel

Previous

Next

Parameters:

KeyName:

Type: AWS::EC2::KeyPair::KeyName

VPCIds:

Type: AWS::EC2::VPC::Id

SubnetIds:

Type: AWS::EC2::Subnet::Id

Resources:

#EC2 INSTANCE LAUNCH

Ec2Instance:

Type: AWS::EC2::Instance

Properties:

InstanceType: t2.micro

KeyName: KeyName

SecurityGroupIds:

- sg-895c5de7

SubnetId: SubnetIds

ImageId: ami-0fc20dd1da406780b

Tags:

- Key: Name

Value: CF-EC2

- Key: Owner

Value: Ayush

UserData:

Fn::Base64:

!Sub |

#!/bin/bash

echo "update"

apt update -y

echo "install nginx"

apt install -y nginx

echo "install python setup-tools"

apt install -y python-setuptools

echo "mkdir"

mkdir -p /opt/aws/bin

echo "download-s3"

wget https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.tar.gz

```
UserData:
  Fn::Base64:
    !Sub |
      #!/bin/bash
      echo "update"
      apt update -y
      echo "install nginx"
      apt install -y nginx
      echo "install python setup-tools"
      apt install -y python-setuptools
      echo "mkdir"
      mkdir -p /opt/aws/bin
      echo "download-s3"
      wget https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.tar.gz
      echo "python easy_install"
      python /usr/lib/python2.7/dist-packages/easy_install.py --script-dir /opt/aws/bin aws-cfn-bootstrap-latest.tar.gz
      echo "cfn-init"
      /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --region ${AWS::Region} --resource Ec2Instance --configsets SetupEnvironment,UpdateEnvironment
      /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --region ${AWS::Region} --resource Ec2Instance

Metadata:
  AWS::CloudFormation::Init:
```

```
Metadata:
  AWS::CloudFormation::Init:

    configSets:
      SetupEnvironment:
        - setupCfnHup
      UpdateEnvironment:
        - updateHostsFile

    setupCfnHup:
      files:
        '/var/www/html/index.html':
          content: !Sub |
            <h1>Deployed via Cloud formation</h1>
          mode: '000644'
          owner: root
          group: root

        '/etc/cfn/cfn-hup.conf':
          content: !Sub |
            [main]
            stack=${AWS::StackId}
            region=${AWS::Region}
            interval=1
          mode: '000400'
          owner: root
          group: root
```

```

'/etc/cfn/hooks.d/cfn-auto-reloader.conf':
content: !Sub |
[cfn-auto-reloader-hook]
triggers=post.update
path=Resources.Ec2Instance.Metadata.AWS::CloudFormation::Init
action=/opt/aws/bin/cfn-init --verbose --stack=${AWS::StackName} --region=${AWS::Region} --resource=Ec2Instance --configsets UpdateEnvironment
runas=root
mode: '000400'
owner: root
group: root

'/lib/systemd/system/cfn-hup.service':
content: !Sub |
[Unit]
Description=cfn-hup daemon
[Service]
Type=simple
ExecStart=/opt/aws/bin/cfn-hup
Restart=always
[Install]
WantedBy=multi-user.target
mode: '000400'
owner: root
group: root

commands:
01enable_cfn_hup:
  command : systemctl enable cfn-hup.service
02start_cfn_hup:
  command : systemctl start cfn-hup.service

```

```

updateHostsFile:
commands:
  reload_nginx:
    command: nginx -s reload

```

## #AUTOSCALING CONFIGURATION

```

AutoScalingGroup:
  Type: AWS::AutoScaling::AutoScalingGroup
  Properties:
    AutoScalingGroupName: CF-autoscaling
    LaunchConfigurationName: !Ref 'LaunchConfig'
    AvailabilityZones:
      - us-east-2b
      - us-east-2a
    MaxSize: 2
    MinSize: 1
    DesiredCapacity: 1
    Tags:
      - Key: Name
        Value: CF-EC2
        PropagateAtLaunch: "true"
      - Key: Owner
        Value: Ayush
        PropagateAtLaunch: "true"
    UpdatePolicy:
      AutoScalingRollingUpdate:
        MaxBatchSize: '1'
        MinInstancesInService: '1'
        PauseTime: PT15M
        WaitOnResourceSignals: 'true'

```

Stacks (1)

Filter by stack name

Active View nested

CF-EC2

2020-04-24 17:35:06 UTC+0530

CREATE\_COMPLETE

CF-EC2

Delete Update Stack actions

Stack info Events Resources Outputs Parameters Template Change sets

Resources (3)

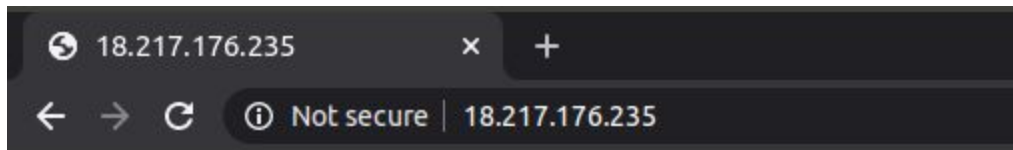
Search resources

Logical ID	Physical ID	Type	Status	Status reason
AutoScalingGroup	CF-autoscaling	AWS::AutoScaling::AutoScalingGroup	CREATE_COMPLETE	-
Ec2Instance	i-06ad75b1efd224aa3	AWS::EC2::Instance	CREATE_COMPLETE	-
LaunchConfig	CF-EC2-LaunchConfig-1DXCFYSFOX54Y	AWS::AutoScaling::LaunchConfiguration	CREATE_COMPLETE	-

Launch Instance Connect Actions

Instance State : Running Add filter

	Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
	CF-EC2	i-02b7cb78845c221c6	t2.micro	us-east-2b	running	Initializing
	CF-EC2	i-06ad75b1efd224aa3	t2.micro	us-east-2b	running	Initializing



Deployed via Cloud formation



Deployed via Cloud formation

## Updating userdata for update rolling

```
UserData:
  Fn::Base64:
    !Sub |
      #!/bin/bash
      echo "update"
      apt update -y
      echo "install nginx"
      echo "this is for rolling update"      #for rolling update
      apt install -y nginx
      echo "install python setup-tools"
      apt install -y python-setuptools
      echo "mkdir"
      mkdir -p /opt/aws/bin
      echo "download-s3"
      wget https://s3.amazonaws.com/cloudformation-examples/aws-cfn-bootstrap-latest.tar.gz
      echo "python easy_install"
      python /usr/lib/python2.7/dist-packages/easy_install.py --script-dir /opt/aws/bin aws-cfn-bootstrap-latest.tar.gz
      echo "cfn-innit"
      /opt/aws/bin/cfn-init -v --stack ${AWS::StackName} --region ${AWS::Region} --resource LaunchConfig --configsets SetupEnvironment,UpdateEnvironment
      /opt/aws/bin/cfn-signal -e $? --stack ${AWS::StackName} --region ${AWS::Region} --resource AutoScalingGroup

Metadata:
  AWS::CloudFormation::Init:
    configSets:
      SetupEnvironment:
        - setupCfnHup
      UpdateEnvironment:
        - updateHostsFile

    setupCfnHup:
      files:
        '/var/www/html/index.html':
          content: !Sub |
            <h1>Deployed via Cloud formation updated</h1>
          mode: '000644'
          owner: root
```

## Update stack

### Prerequisite - Prepare template

#### Prepare template

Every stack is based on a template. A template is a JSON or YAML file that contains configuration information about the AWS resources you want to include in the stack.

☐ Use current template

☒ Replace current template

☐ Edit template in designer

### Specify template

A template is a JSON or YAML file that describes your stack's resources and properties.

#### Template source

Selecting a template generates an Amazon S3 URL where it will be stored.

☒ Amazon S3 URL

☐ Upload a template file

Amazon S3 URL

https://

Amazon S3 template URL

S3 URL: Will be generated when URL is provided

[View in Designer](#)



## Events (19)



Timestamp	Logical ID	Status	Status reason
2020-04-24 17:46:10 UTC+0530	AutoScalingGroup	UPDATE_IN_PROGRESS	New instance(s) added to autoscaling group - Waiting on 1 resource signal(s) with a timeout of PT15M.
2020-04-24 17:44:53 UTC+0530	AutoScalingGroup	UPDATE_IN_PROGRESS	Temporarily setting autoscaling group MinSize and DesiredCapacity to 2.
2020-04-24 17:44:53 UTC+0530	AutoScalingGroup	UPDATE_IN_PROGRESS	Rolling update initiated. Terminating 1 obsolete instance(s) in batches of 1, while keeping at least 1 instance(s) in service. Waiting on resource signals with a timeout of PT15M when new instances are added to the autoscaling group.
2020-04-24 17:44:50 UTC+0530	AutoScalingGroup	UPDATE_IN_PROGRESS	-
2020-04-24 17:44:49 UTC+0530	LaunchConfig	UPDATE_COMPLETE	-
2020-04-24 17:44:48 UTC+0530	LaunchConfig	UPDATE_IN_PROGRESS	Resource creation Initiated
2020-04-24 17:44:48 UTC+0530	LaunchConfig	UPDATE_IN_PROGRESS	Requested update requires the creation of a new physical resource; hence creating one.
2020-04-24 17:44:44 UTC+0530	CF-EC2	UPDATE_IN_PROGRESS	User Initiated
2020-04-24 17:36:38 UTC+0530	CF-EC2	CREATE_COMPLETE	-
2020-04-24 17:36:37 UTC+0530	AutoScalingGroup	CREATE_COMPLETE	-

## Events (27)



Timestamp	Logical ID	Status	Status reason
2020-04-24 18:14:59 UTC+0530	ec2-cf	UPDATE_COMPLETE	-
2020-04-24 18:14:59 UTC+0530	LaunchConfig	DELETE_COMPLETE	-
2020-04-24 18:14:59 UTC+0530	LaunchConfig	DELETE_IN_PROGRESS	-
2020-04-24 18:14:58 UTC+0530	ec2-cf	UPDATE_COMPLETE_CLEANUP_IN_PROGRESS	-
2020-04-24 18:14:56 UTC+0530	AutoScalingGroup	UPDATE_COMPLETE	-
2020-04-24 18:14:55 UTC+0530	AutoScalingGroup	UPDATE_IN_PROGRESS	Successfully terminated instance(s) [i-0f95ba069a5b78f00] (Progress 100%).
2020-04-24 18:14:54 UTC+0530	AutoScalingGroup	UPDATE_IN_PROGRESS	Terminating instance(s) [i-0f95ba069a5b78f00]; replacing with 0 new instance(s).
2020-04-24 18:14:53 UTC+0530	AutoScalingGroup	UPDATE_IN_PROGRESS	Received SUCCESS signal with Uniqueid i-07362d708a502bb18

# Deployed via Cloud formation updated