about autoscling and application loadbalancer with two targetgroups and two listener rule

AWSTemplateFormatVersion: 2010-09-09

Description: ProjectX scalable stack of frontal nodes

Parameters:

InstanceType:

Description: Type of EC2 instance to launch

Type: String

Default: t2.micro

EnvironmentName:

Description: An environment name that will be prefixed to resource names

Type: String

SSHKeyName:

Description: The EC2 Key Pair to allow SSH access to the instances

Type: String

Default: dev

Password:

NoEcho: 'true'

Type: String

Description: New account password

MinLength: '1'

MaxLength: '41'

ConstraintDescription: the password must be between 1 and 41 characters

MyVpc:

Description: aboutvpc

Type: String

Default: vpc-fa5a4292

Subnets:

Description: about subnet

Type: 'List<AWS::EC2::Subnet::Id>'

AWSRegion:

Type: String

Default: us-east-2

InstanceName:

Description: Logical instance name seen in AWS console

Type: String

Default: ProjectX-Live-Node

AvailabilityZones:

Type: CommaDelimitedList

Default: 'us-east-2a,us-east-2b,us-east-2c'

SecurityGroup:

Description: Select the Security Group to apply to the Application Load Balancer

Type: 'AWS::EC2::SecurityGroup::Id'

Mappings:

AWSInstanceType2Arch:

t2.micro:

Arch: '64'

m1.small:

Arch: '32'

m1.large:

Arch: '64'

m1.xlarge:

Arch: '64'

m2.xlarge:

Arch: '64'

m2.2xlarge:

Arch: '64'

m2.4xlarge:

Arch: '64'

c1.medium:

Arch: '32'

c1.xlarge:

Arch: '64'

cc1.4xlarge:

Arch: '64'

AWSRegionArch2AMI:

us-east-2:

'32': ami-b89842d1

'64': ami-0653e888ec96eab9b

us-west-1:

'32': ami-d5712a90

'64': ami-069339bea0125f50d

eu-west-1:

'32': ami-25e8d351

'64': ami-3b65664f

Resources:

CFNUser: // ABOUT IAM User,group,Policy

Type: AWS::IAM::User

Properties:

LoginProfile:

Password: !Ref 'Password'

CFNUserGroup:

Type: AWS::IAM::Group

CFNAdminGroup:

Type: AWS::IAM::Group

Users:

Type: AWS::IAM::UserToGroupAddition

Properties:

GroupName: !Ref 'CFNUserGroup'

Users: [!Ref 'CFNUser']

Admins:

Type: AWS::IAM::UserToGroupAddition

Properties:

GroupName: !Ref 'CFNAdminGroup'

Users: [!Ref 'CFNUser']

CFNUserPolicies:

Type: AWS::IAM::Policy

Properties:

PolicyName: CFNUsers

PolicyDocument:

Statement:

- Effect: Allow

Action: ['cloudformation:Describe\*', 'cloudformation:List\*', 'cloudformation:Get\*']

Resource: '\*'

Groups: [!Ref 'CFNUserGroup']

CFNAdminPolicies:

Type: AWS::IAM::Policy

Properties:

PolicyName: CFNAdmins

PolicyDocument:

Statement:

- Effect: Allow

Action: cloudformation:\*

Resource: '\*'

Groups: [!Ref 'CFNAdminGroup']

CFNKeys:

Type: AWS::IAM::AccessKey

Properties:

UserName: !Ref 'CFNUser'

NodeGroup:

Type: 'AWS::AutoScaling::AutoScalingGroup'

Properties:

AvailabilityZones: !Ref AvailabilityZones

LaunchConfigurationName: !Ref LaunchConfig

MinSize: '2'

MaxSize: '4'

TargetGroupARNs:

- !Ref ALBTargetGroup

- !Ref TargetGroup2

|  |
| --- |
| ScheduledActionUp: |
|  | Type: AWS::AutoScaling::ScheduledAction |
|  | Properties: |
|  | AutoScalingGroupName: !Ref 'WebServerGroup' |
|  | MaxSize: '10' |
|  | MinSize: '5' |
|  | Recurrence: 0 7 \* \* \* |
|  | ScheduledActionDown: |
|  | Type: AWS::AutoScaling::ScheduledAction |
|  | Properties: |
|  | AutoScalingGroupName: !Ref 'WebServerGroup' |
|  | MaxSize: '1' |
|  | MinSize: '1' |
|  | Recurrence: 0 19 \* \* \* |

LaunchConfig:

Type: 'AWS::AutoScaling::LaunchConfiguration'

Properties:

ImageId: !FindInMap

- AWSRegionArch2AMI

- !Ref 'AWS::Region'

- !FindInMap

- AWSInstanceType2Arch

- !Ref InstanceType

- Arch

UserData: !Base64

'Fn::Join':

- ''

- - |

#!/bin/bash

- |

sudo apt-get updtae && install apache2 -y

sudo service start apache2

sudo usermod -a -G ubuntu root

sudo chown -R root:ubuntu /var/www

sudo chmod 2755 /var/www

sudo mkdir -p /var/www/html/in

sudo mkdir -p /var/www/html/im

sudo echo "hello everytwo">>/var/www/html/im/index.html

sudo echo "hello everyone">> /var/www/html/in/image

SecurityGroups:

- !Ref ProjectXNodeSecurityGroup

InstanceType: !Ref InstanceType

KeyName: !Ref SSHKeyName

WebServerScaleUpPolicy:

Type: 'AWS::AutoScaling::ScalingPolicy'

Properties:

AdjustmentType: PercentChangeInCapacity

AutoScalingGroupName: !Ref NodeGroup

Cooldown: '300'

ScalingAdjustment: '100'

WebServerScaleDownPolicy:

Type: 'AWS::AutoScaling::ScalingPolicy'

Properties:

AdjustmentType: ChangeInCapacity

AutoScalingGroupName: !Ref NodeGroup

Cooldown: '600'

ScalingAdjustment: '-1'

CPUAlarmHigh:

Type: 'AWS::CloudWatch::Alarm'

Properties:

AlarmDescription: Scale-up if CPU > 40% for 10 minutes

MetricName: CPUUtilization

Namespace: AWS/EC2

Statistic: Average

Period: '120'

EvaluationPeriods: '5'

Threshold: '40'

AlarmActions:

- !Ref WebServerScaleUpPolicy

Dimensions:

- Name: AutoScalingGroupName

Value: !Ref NodeGroup

ComparisonOperator: GreaterThanThreshold

CPUAlarmLow:

Type: 'AWS::CloudWatch::Alarm'

Properties:

AlarmDescription: Scale-down if CPU <20% for 60 min

MetricName: CPUUtilization

Namespace: AWS/EC2

Statistic: Average

Period: '120'

EvaluationPeriods: '30'

Threshold: '20'

AlarmActions:

- !Ref WebServerScaleDownPolicy

Dimensions:

- Name: AutoScalingGroupName

Value: !Ref NodeGroup

ComparisonOperator: LessThanThreshold

ALBListener:

Type: 'AWS::ElasticLoadBalancingV2::Listener'

Properties:

DefaultActions:

- Type: forward

TargetGroupArn: !Ref ALBTargetGroup

LoadBalancerArn: !Ref ALBElasticLoadBalancer

Port: 80

Protocol: HTTP

ALBListenerRule:

Type: 'AWS::ElasticLoadBalancingV2::ListenerRule'

DependsOn: ALBListener

Properties:

Actions:

- Type: forward

TargetGroupArn: !Ref ALBTargetGroup

Conditions:

- Field: path-pattern

Values:

- /im/index.html

ListenerArn: !Ref ALBListener

Priority: 2

ListenerRule:

Type: 'AWS::ElasticLoadBalancingV2::ListenerRule'

DependsOn: ALBListener

Properties:

Actions:

- Type: forward

TargetGroupArn: !Ref TargetGroup2

Conditions:

- Field: path-pattern

Values:

- /in/image

ListenerArn: !Ref ALBListener

Priority: 1

ALBElasticLoadBalancer:

Type: 'AWS::ElasticLoadBalancingV2::LoadBalancer'

Properties:

Name: !Ref EnvironmentName

Subnets: !Ref Subnets

SecurityGroups:

- !Ref SecurityGroup

Tags:

- Key: Name

Value: !Ref EnvironmentName

ALBTargetGroup:

Type: 'AWS::ElasticLoadBalancingV2::TargetGroup'

Properties:

Name: sujitjht

VpcId: !Ref MyVpc

Port: 80

Protocol: HTTP

HealthCheckIntervalSeconds: 15

HealthCheckPath: /im/index.html

HealthCheckProtocol: HTTP

HealthCheckTimeoutSeconds: 5

HealthyThresholdCount: 5

UnhealthyThresholdCount: 2

TargetGroup2:

Type: 'AWS::ElasticLoadBalancingV2::TargetGroup'

Properties:

Name: sujitj

VpcId: !Ref MyVpc

Port: 80

Protocol: HTTP

HealthCheckIntervalSeconds: 15

HealthCheckPath: /in/image

HealthCheckProtocol: HTTP

HealthCheckTimeoutSeconds: 5

HealthyThresholdCount: 5

UnhealthyThresholdCount: 2

ProjectXNodeSecurityGroup:

Type: 'AWS::EC2::SecurityGroup'

Properties:

GroupDescription: Enable SSH access and HTTP access on the inbound port

SecurityGroupIngress:

- IpProtocol: tcp

FromPort: '22'

ToPort: '22'

CidrIp: 0.0.0.0/0

- IpProtocol: tcp

FromPort: '80'

ToPort: '80'

CidrIp: 0.0.0.0/0

Outputs:

URL:

Description: The URL of the website

Value: !Join

- ''

- - 'http://'

- !GetAtt

- ALBElasticLoadBalancer

- DNSName