

Project Documentation

Stage A : create infraStructure using CloudFormation “project9.yml”

● Code

AWSTemplateFormatVersion: '2010-09-09'

Description: VPC with EC2 instances, auto scaling, ELB, CloudWatch agent, SSM agent, and NGINX installation

Parameters:

Region:

Type: String

Default: "eu-west-1" #irland

Description: "region :irland"

VpcCIDR:

Type: String

Description: CIDR block for the VPC

Default: 10.0.0.0/16

Subnet1CIDR:

Type: String

Description: CIDR block for subnet 1

Default: 10.0.0.0/24

Subnet2CIDR:

Type: String

Description: CIDR block for subnet 2

Default: 10.0.1.0/24

Resources:

VPC:

Type: AWS::EC2::VPC

Properties:

CidrBlock: !Ref VpcCIDR

EnableDnsSupport: true

EnableDnsHostnames: true

Subnet1:

Type: AWS::EC2::Subnet

Properties:

VpcId: !Ref VPC

CidrBlock: !Ref Subnet1CIDR

MapPublicIpOnLaunch: true

AvailabilityZone: eu-west-1a

Subnet2:

Type: AWS::EC2::Subnet

Properties:

VpcId: !Ref VPC

CidrBlock: !Ref Subnet2CIDR

MapPublicIpOnLaunch: true

AvailabilityZone: eu-west-1b

InternetGateway:

Type: AWS::EC2::InternetGateway

AttachGateway:

Type: AWS::EC2::VPCGatewayAttachment

Properties:

VpcId: !Ref VPC

InternetGatewayId: !Ref InternetGateway

RouteTable:

Type: AWS::EC2::RouteTable

Properties:

VpcId: !Ref VPC

PublicRoute:

Type: AWS::EC2::Route

DependsOn: AttachGateway

Properties:

RouteTableId: !Ref RouteTable

DestinationCidrBlock: "0.0.0.0/0"

GatewayId: !Ref InternetGateway

Subnet1RouteTableAssociation:

Type: AWS::EC2::SubnetRouteTableAssociation

Properties:

SubnetId: !Ref Subnet1

RouteTableId: !Ref RouteTable

Subnet2RouteTableAssociation:

Type: AWS::EC2::SubnetRouteTableAssociation

Properties:

SubnetId: !Ref Subnet2

RouteTableId: !Ref RouteTable

SecurityGroup:

Type: AWS::EC2::SecurityGroup

Properties:

GroupDescription: Allow all traffic on port 80

VpcId: !Ref VPC

SecurityGroupIngress:

- IpProtocol: tcp
 - FromPort: 80
 - ToPort: 80
 - CidrIp: 0.0.0.0/0

TargetGroup:

Type: AWS::ElasticLoadBalancingV2::TargetGroup

Properties:

- TargetType: instance
- Protocol: HTTP
- Port: 80
- VpcId: !Ref VPC
- HealthCheckPath: /
- HealthCheckProtocol: HTTP
- HealthCheckIntervalSeconds: 30
- HealthCheckTimeoutSeconds: 10
- HealthyThresholdCount: 5
- UnhealthyThresholdCount: 2

LoadBalancer:

Type: AWS::ElasticLoadBalancingV2::LoadBalancer

Properties:

- Name: MyLoadBalancer
- Subnets:
 - !Ref Subnet1
 - !Ref Subnet2
- SecurityGroups:
 - !Ref SecurityGroup
- Scheme: internet-facing
- Type: application

LoadBalancerListener:

Type: AWS::ElasticLoadBalancingV2::Listener

Properties:

- DefaultActions:
 - TargetGroupArn: !Ref TargetGroup
- Type: forward
- LoadBalancerArn: !Ref LoadBalancer
- Port: 80
- Protocol: HTTP

LaunchConfiguration:

Type: AWS::AutoScaling::LaunchConfiguration

Properties:

- ImageId: ami-0c1c30571d2dae5c9
- InstanceType: t2.micro
- KeyName: key1

SecurityGroups:

- !Ref SecurityGroup

UserData:

```
Fn::Base64: |  
#!/bin/bash  
wget
```

```
https://s3.amazonaws.com/amazoncloudwatch-agent/linux/amd64/latest/AmazonCloudWatchAgent.zip
```

```
sudo apt install unzip  
sudo ./install.sh  
sudo systemctl start amazon-cloudwatch-agent  
sudo systemctl enable amazon-cloudwatch-agent
```

```
wget
```

```
https://s3.amazonaws.com/ec2-downloads-windows/SSMAgent/latest/debian_amd64/amazon-ssm-agent.deb
```

```
sudo snap refresh amazon-ssm-agent  
sudo snap remove amazon-ssm-agent  
sudo dpkg -i amazon-ssm-agent.deb  
sudo systemctl start amazon-ssm-agent  
sudo systemctl enable amazon-ssm-agent
```

IamInstanceProfile: !Ref InstanceProfile

InstanceProfile:

Type: AWS::IAM::InstanceProfile

Properties:

Roles:

- !Ref EC2Role

EC2Role:

Type: AWS::IAM::Role

Properties:

AssumeRolePolicyDocument:

Statement:

- Effect: Allow

Principal:

Service:

- ec2.amazonaws.com

Action: sts:AssumeRole

Policies:

- PolicyName: EC2Policy

PolicyDocument:

Version: "2012-10-17"

Statement:

- Effect: Allow

Action:

- "ssm:DescribeAssociation"
- "ssm:GetDeployablePatchSnapshotForInstance"
- "ssm:GetDocument"
- "ssm:DescribeDocument"
- "ssm:GetManifest"
- "ssm:GetParameter"
- "ssm:GetParameters"
- "ssm:ListAssociations"
- "ssm:ListInstanceAssociations"
- "ssm:PutInventory"
- "ssm:PutComplianceItems"
- "ssm:PutConfigurePackageResult"
- "ssm:UpdateAssociationStatus"
- "ssm:UpdateInstanceAssociationStatus"
- "ssm:UpdateInstanceInformation"
- "ssmmessages:CreateControlChannel"
- "ssmmessages:CreateDataChannel"
- "ssmmessages:OpenControlChannel"
- "ssmmessages:OpenDataChannel"

Resource: '*'

- Effect: Allow

Action:

- "cloudwatch:PutMetricData"
- "cloudwatch:PutMetricAlarm"
- "cloudwatch:DescribeAlarms"
- "cloudwatch:GetMetricStatistics"
- "logs:CreateLogGroup"
- "logs:CreateLogStream"
- "logs:PutLogEvents"
- "logs:DescribeLogStreams"

Resource: "*"

AutoScalingGroup:

Type: AWS::AutoScaling::AutoScalingGroup

Properties:

DesiredCapacity: '4'

LaunchConfigurationName: !Ref LaunchConfiguration

MaxSize: '7'

MinSize: '4'

VPCZoneIdentifier:

- !Ref Subnet1 # First subnet

- !Ref Subnet2 # Second subnet

● Output

1- 2 Ec2 in each subnet

The screenshot shows the AWS Management Console 'Instances' page. The left sidebar is open, showing the navigation menu. The main content area displays a table of four EC2 instances. All instances are in the 'Running' state. The instances are distributed across two availability zones: eu-west-1a and eu-west-1b. The table columns are: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, and Availability Zone.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
	i-046d14cc1debcc41b	Running	t2.micro	2/2 checks passed	View alarms +	eu-west-1a
	i-03f0c8be691097af4	Running	t2.micro	2/2 checks passed	View alarms +	eu-west-1b
	i-015603721cde22f3d	Running	t2.micro	2/2 checks passed	View alarms +	eu-west-1b
	i-05aa5e31f85c3b00d	Running	t2.micro	2/2 checks passed	View alarms +	eu-west-1a

2- Load blancer

The screenshot shows the AWS Management Console 'Load Balancers' page. The left sidebar is open, showing the navigation menu. The main content area displays a table of one Elastic Load Balancing instance. The instance is named 'MyLoadBalancer' and is in the 'Active' state. The table columns are: Name, State, VPC ID, and Availability Zones.

Name	State	VPC ID	Availability Zones
MyLoadBalancer	Active	vpc-0b1f94a25bef03497	2 Availability Zones

3- Target group

The screenshot shows the AWS Management Console 'Target Groups' page. The left sidebar is open, showing the navigation menu. The main content area displays a table of four registered targets. All targets are in the 'Healthy' state. The targets are distributed across two availability zones: eu-west-1a and eu-west-1b. The table columns are: Instance ID, Name, Port, Zone, Health status, and Health status details.

Instance ID	Name	Port	Zone	Health status	Health status details
i-03f0c8be691097af4		80	eu-west-1b	Healthy	-
i-046d14cc1debcc41b		80	eu-west-1a	Healthy	-
i-015603721cde22f3d		80	eu-west-1b	Healthy	-
i-05aa5e31f85c3b00d		80	eu-west-1a	Healthy	-

4- Stack

Stacks (1)

Filter by stack name

Filter status: Active ☐ View nested

Stacks

- project9
2024-04-07 21:51:38 UTC+0200
CREATE_COMPLETE

Resources (17)

Logical ID	Physical ID	Type	Status	Mode
AttachGateway	IGW/vpc-0b1f94a25bef03497	AWS::EC2::VPCGatewayAttachment	CREATE_COMPLETE	-
AutoScalingGroup	project9-AutoScalingGroup-9Ffz31twPFNU	AWS::AutoScaling::AutoScalingGroup	CREATE_COMPLETE	-
EC2Role	project9-EC2Role-IGG73OwS3409	AWS::IAM::Role	CREATE_COMPLETE	-
InstanceProfile	project9-InstanceProfile-205yN1M342kF	AWS::IAM::InstanceProfile	CREATE_COMPLETE	-
InternetGateway	igw-0b9dfd7e08b034b47	AWS::EC2::InternetGateway	CREATE_COMPLETE	-
LaunchConfiguration	project9-LaunchConfiguration-	AWS::AutoScaling::LaunchConfiguration	CREATE_COMPLETE	-

Stage B : install nginx using ssm

1- Installation code

install-nginx

Delete Actions Run command

Description Content Versions Details

Document version: 1 (Default) Compare versions

The content of this document is as follows:

```
1- {}
2- "schemaVersion": "1.2",
3- "description": "Installs Nginx",
4- "parameters": {},
5- "runtimeConfig": {
6-   "awsrunShellScript": {
7-     "properties": {
8-       "id": "0.awsrunShellScript",
9-       "runCommand": [
10-        "apt update",
11-        "apt install -y nginx",
12-        "sudo systemctl enable nginx",
13-        "sudo ufw allow 'Nginx HTTP'",
14-        "sudo ufw allow 'Nginx HTTPS'",
15-        "sudo ufw reload"
16-      ]
17-     }
18-   }
19- }
20- }
```

2- Fleet Manager

Command status

Overall status	Detailed status	# targets	# completed	# error	# delivery timed out
Success	Success	4	4	0	0

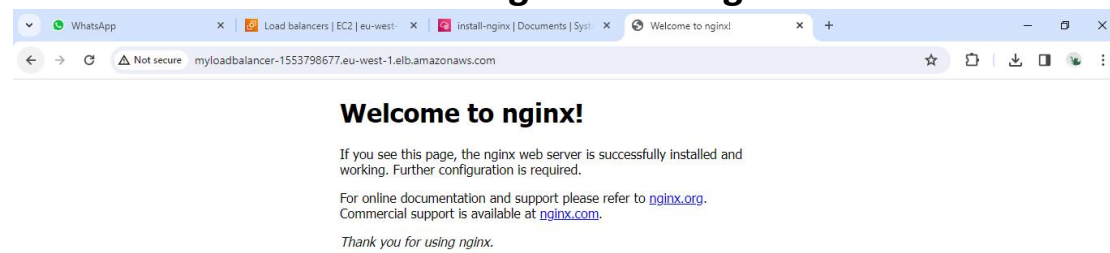
Targets and outputs

Search command invocations

Instance ID	Instance name	Status	Detailed Status	Start time	Finish time
i-05aa5e31f85c3b00d	ip-10-0-0-193.eu-west-1.compute.internal	Success	Success	Sun, 07 Apr 2024 20:41:09 GMT	Sun, 07 Apr 2024 20:41:09 GMT
i-046d14cc1debcc41b	ip-10-0-0-249.eu-west-1.compute.internal	Success	Success	Sun, 07 Apr 2024 20:41:09 GMT	Sun, 07 Apr 2024 20:41:09 GMT
i-03f0c8be691097af4	ip-10-0-1-210.eu-west-1.compute.internal	Success	Success	Sun, 07 Apr 2024 20:41:09 GMT	Sun, 07 Apr 2024 20:41:09 GMT
i-015603721cde22f3d	ip-10-0-1-231.eu-west-1.compute.internal	Success	Success	Sun, 07 Apr 2024 20:41:09 GMT	Sun, 07 Apr 2024 20:41:09 GMT

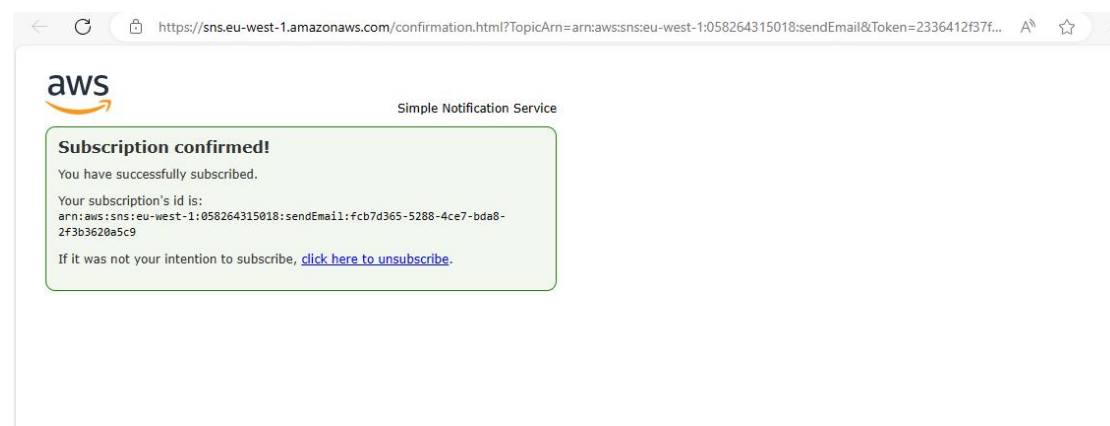
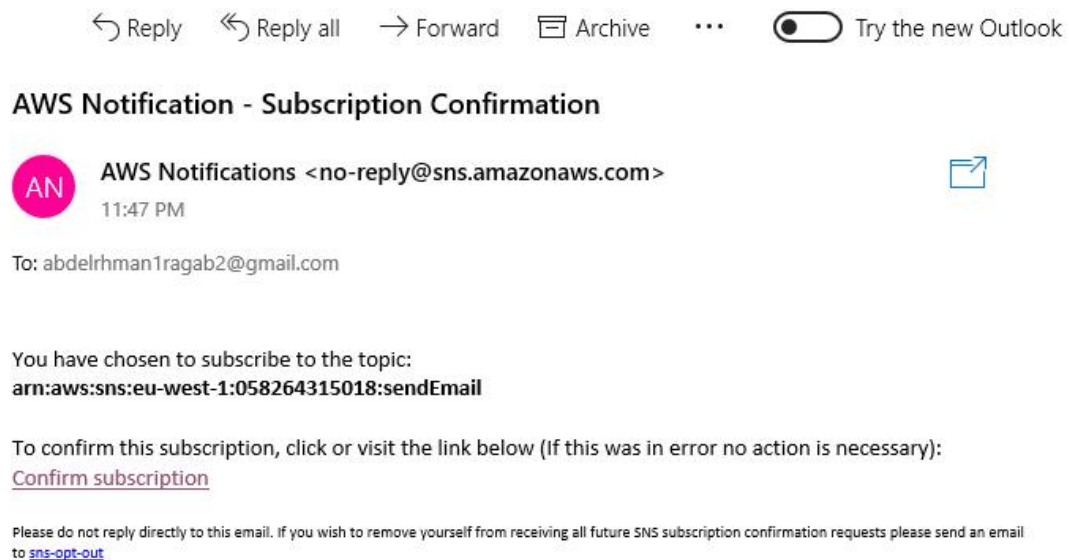
CloudWatch alarm

3- Access loadbalancer using DNS and nginx installed



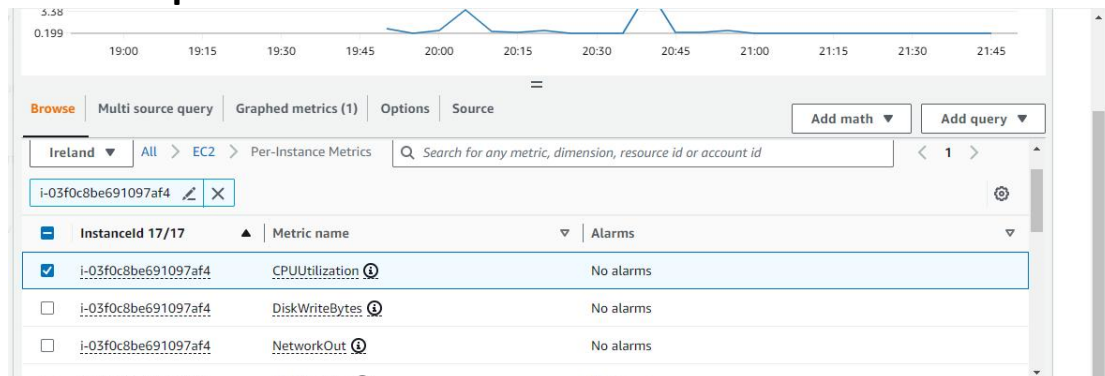
Stage C : cloudwatch send mail

1- Create sns and activate it



2- Create Alarm and select metrics and threshold

For example CPU-Utilization for EC2



The screenshot shows the 'Conditions' section of the AWS CloudWatch console. The 'Threshold type' is set to 'Static'. The condition is 'Greater/Equal' to the threshold value of 0.2. The 'Additional configuration' section is expanded, showing 'Cancel' and 'Next' buttons.

3- Output Email

The screenshot shows an email notification from AWS Notifications. The email is titled "ALARM: "CPU-Alarm" in EU (Ireland)". It contains details about the alarm state change from INSUFFICIENT_DATA to ALARM. The email includes the following information:

- Alarm Details:**
 - Name: CPU-Alarm
 - Description: INSUFFICIENT_DATA -> ALARM
 - State Change: INSUFFICIENT_DATA -> ALARM
 - Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [0.2350829232731872 (07/04/24 21:51:00)] was greater than or equal to the threshold (0.2) (minimum 1 datapoint for OK -> ALARM transition).
 - Timestamp: Sunday 07 April, 2024 21:56:30 UTC
 - AWS Account: 058264315018
 - Alarm Arn: arn:aws:cloudwatch:eu-west-1:058264315018:alarm:CPU-Alarm
- Threshold:**
 - The alarm is in the ALARM state when the metric is GreaterThanOrEqualToThreshold 0.2 for at least 1 of the last 1 period(s) of 300 seconds.