

# AWS Auto Scaling Web Environment with CloudFormation

Instruction  
By @Chihiro-001

# Scenario

- A company wants to create a staging environment and a production environment for its website using a template. They want to be able to use the same environment for both validation and production.
- The website must be accessible via a secure connection when browsing.
- The template environment should be used as a basis for adding features such as EC2 auto-scaling and alarm notification based on status monitoring.

# 1 : Requirement

Create template and setup environment with the following requirements:

- automatically build an environment that fulfills the following 1-3:
  1. Create public subnets in two Availability Zones (AZ).
  2. Deploy EC2 instances in each public subnet, using Apache as the web server.
  3. Use ALB (Application Load Balancer) and configure each EC2 instance in the target group.

The following steps are done manually after setting up the stack:

- have secure encrypted communication using a domain for the web page.
  - Hint: Issue a certificate using AWS services during step 1.
- make EC2 automatically scale out when the CPU load exceeds 70%, and scale in when it falls below 20%.
- configure so that when the threshold is exceeded in monitoring, alerts can be sent to responsible personnel via email.

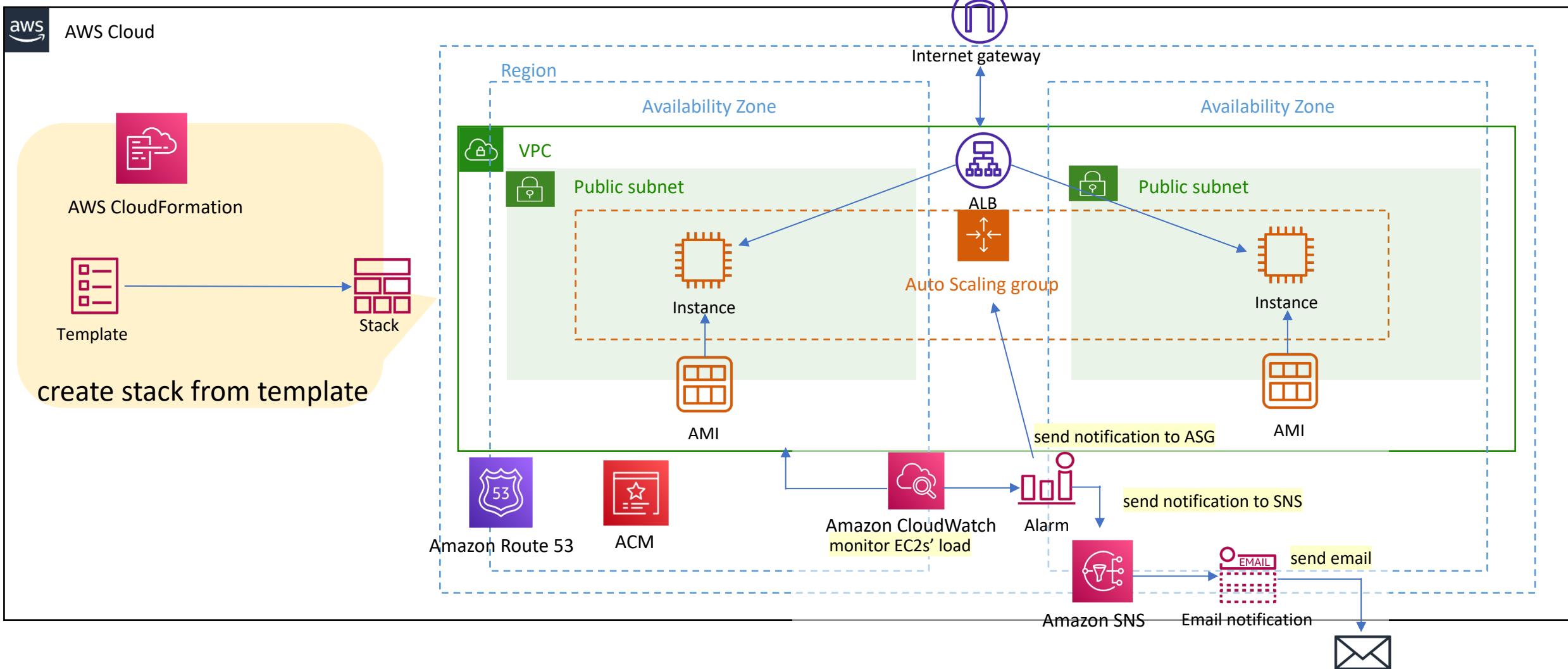
## 2. Design

- Specification (Solution) for Requirements:

No	Requirement	Specification (Solution)
1	Automatically build an environment that fulfills the following conditions: ① Create public subnets in two Availability Zones (AZ). ② Deploy EC2 instances in each public subnet, using Apache as the web server. ③ Use ALB to configure each EC2 instance in the target group.	Use CloudFormation to automatically build the environment. ① Create YAML files for environment construction such as VPC and Internet Gateway. ② Prepare YAML files to place AL2 AMIs that support Apache in each public subnet. ③ Use the YAML files created in step ② to configure ALB and set EC2 instances in the target group.
2	have secure encrypted communication using a domain for the web page.	As preparation, create a Route53 Hosted Zone and acquire an external domain. Configure to issue an SSL certificate in ACM for the acquired domain name during step 1.
3	make EC2 automatically scale out when the CPU load exceeds 70%, and scale in when it falls below 20%.	Use ASG (Auto Scaling Group) with Simple Scaling Policy to scale out when CPU load exceeds 70% and scale in when it falls below 20%.
4	Configure so that when the threshold is exceeded in monitoring, alerts can be sent to responsible person(s) via email.	Set up CloudWatch to monitor CPU thresholds. Set up SNS (Simple Notification Service) to send alert information via email to responsible personnel.

# 2. Design

Diagram: CloudFormation + Auto Scaling Group + CloudWatch + SNS



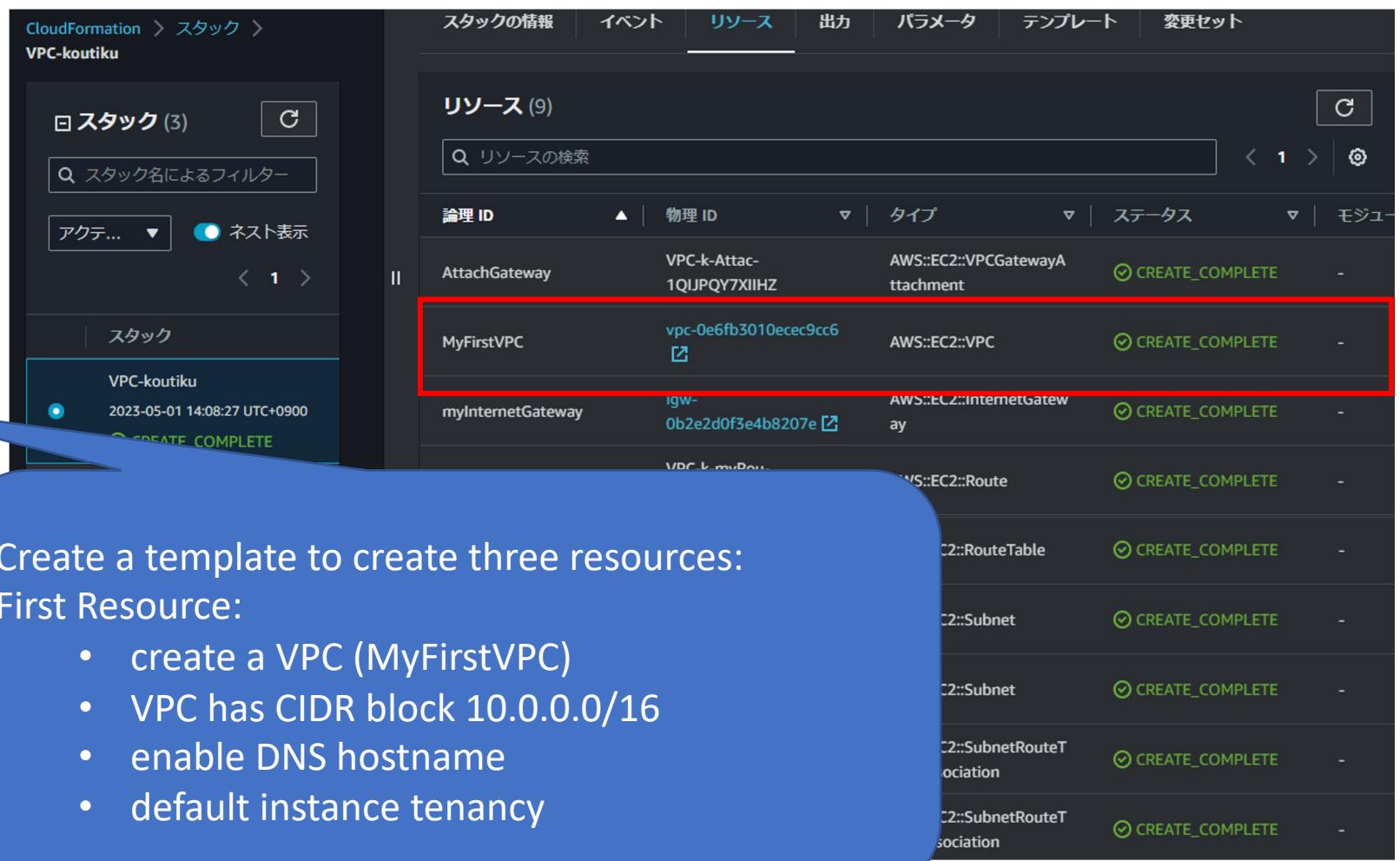
# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

- ① Create public subnets in two Availability Zones (AZ). Create a template to create three resources:

Create VPC environment

```
Resources:  
  MyFirstVPC:  
    Type: AWS::EC2::VPC  
    Properties:  
      CidrBlock: 10.0.0.0/16  
      EnableDnsSupport: 'true'  
      EnableDnsHostnames: 'true'  
      InstanceTenancy: default  
    Tags:  
      - Key: Name  
        Value: CF-VPC  
  
  PublicRouteTable:  
    Type: AWS::EC2::RouteTable  
    Properties:  
      VpcId: !Ref MyFirstVPC  
      Tags:  
        - Key: Name  
          Value: CF-VPC-PublicRT  
  
  PublicSubnet1A:  
    Type: AWS::EC2::Subnet  
    Properties:  
      VpcId: !Ref MyFirstVPC  
      CidrBlock: 10.0.0.0/24
```



Create a template to create three resources:  
First Resource:

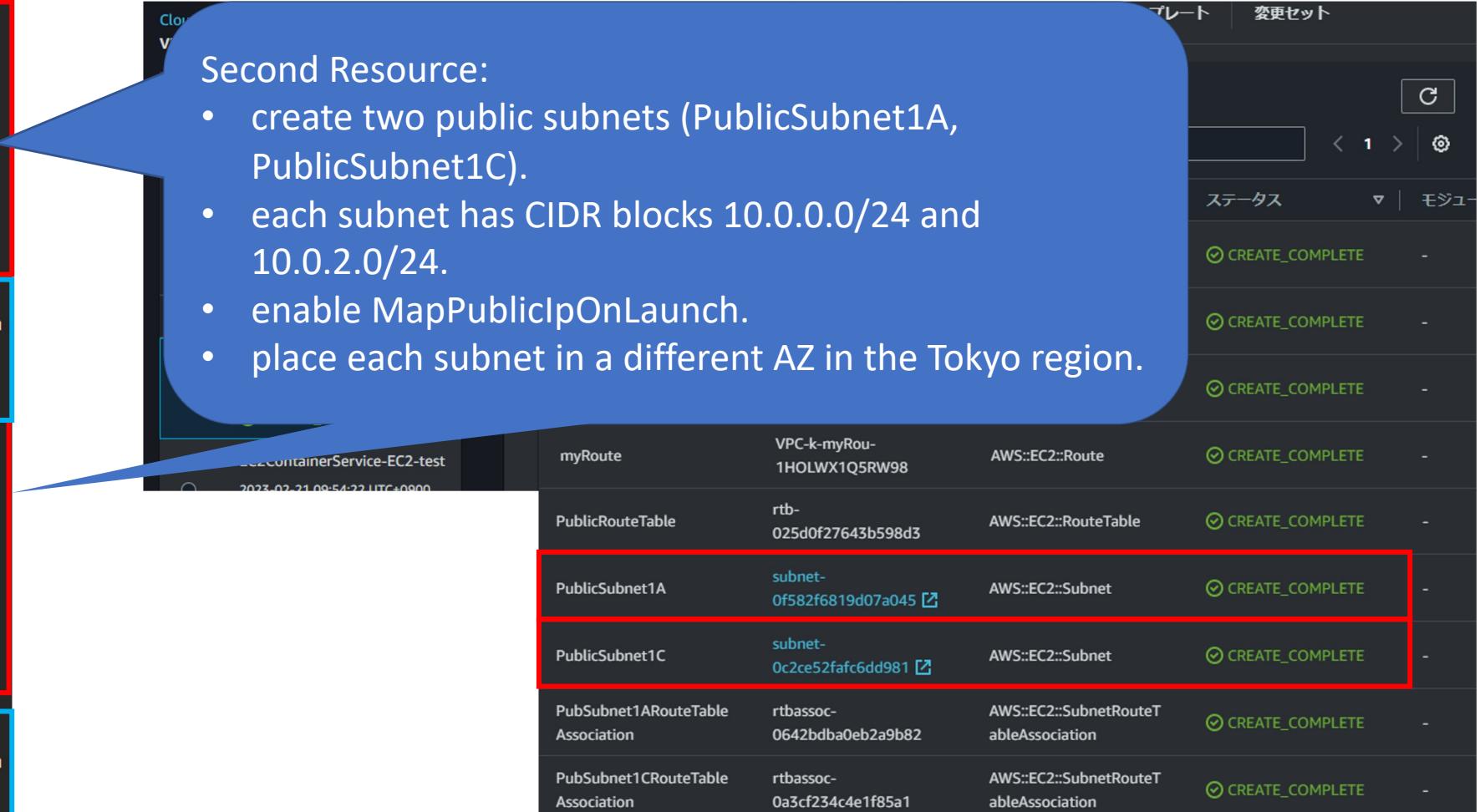
- create a VPC (MyFirstVPC)
- VPC has CIDR block 10.0.0.0/16
- enable DNS hostname
- default instance tenancy

# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

- ① Create public subnets in two Availability Zones (AZ). Create a template to create three resources:

```
PublicSubnet1A:  
  Type: AWS::EC2::Subnet  
  Properties:  
    VpcId: !Ref MyFirstVPC  
    CidrBlock: 10.0.0.0/24  
    MapPublicIpOnLaunch: true  
    AvailabilityZone: "ap-northeast-1a"  
  Tags:  
    - Key: Name  
      Value: CF-publicsubnet-1a  
  
PubSubnet1ARouteTableAssociation:  
  Type: AWS::EC2::SubnetRouteTableAssociation  
  Properties:  
    SubnetId: !Ref PublicSubnet1A  
    RouteTableId: !Ref PublicRouteTable  
  
PublicSubnet1C:  
  Type: AWS::EC2::Subnet  
  Properties:  
    VpcId: !Ref MyFirstVPC  
    CidrBlock: 10.0.2.0/24  
    MapPublicIpOnLaunch: true  
    AvailabilityZone: "ap-northeast-1c"  
  Tags:  
    - Key: Name  
      Value: CF-publicsubnet-1c  
  
PubSubnet1CRouteTableAssociation:  
  Type: AWS::EC2::SubnetRouteTableAssociation  
  Properties:  
    SubnetId: !Ref PublicSubnet1C  
    RouteTableId: !Ref PublicRouteTable
```



リソース名	状態	説明	操作
myRoute	VPC-k-myRou-1HOLWX1Q5RW98	AWS::EC2::Route	CREATE_COMPLETE
PublicRouteTable	rtb-025d0f27643b598d3	AWS::EC2::RouteTable	CREATE_COMPLETE
PublicSubnet1A	subnet-0f582f6819d07a045	AWS::EC2::Subnet	CREATE_COMPLETE
PublicSubnet1C	subnet-0c2ce52fafc6dd981	AWS::EC2::Subnet	CREATE_COMPLETE
PubSubnet1ARouteTableAssociation	rtbassoc-0642bdb40eb2a9b82	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE
PubSubnet1CRouteTableAssociation	rtbassoc-0a3cf234c4e1f85a1	AWS::EC2::SubnetRouteTableAssociation	CREATE_COMPLETE

# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

- ① Create public subnets in two Availability Zones (AZ). Create a template to create three resources:

Create VPC  
environment

```
myInternetGateway:  
  Type: "AWS::EC2::InternetGateway"  
Properties:  
  Tags:  
    - Key: Name  
      Value: CF-INGW  
AttachGateway:  
  Type: AWS::EC2::VPCGatewayAttachment  
Properties:  
  VpcId: !Ref MyFirstVPC  
  InternetGatewayId: !Ref myInternetGateway  
  
myRoute:  
  Type: AWS::EC2::Route  
  DependsOn: myInternetGateway  
Properties:  
  RouteTableId: !Ref PublicRouteTable  
  DestinationCidrBlock: 0.0.0.0/0  
  GatewayId: !Ref myInternetGateway
```

CloudFormation > スタック > VPC-koutiku

スタックの情報 | イベント | リソース | 出力 | パラメータ | テンプレート | 记録

リソース (9)

論理 ID	物理 ID	タイプ	ステータス
AttachGateway	VPC-k-Attach-1QJPQY7XIIHZ	AWS::EC2::VPCGatewayAttachment	CREATE_COMPLETE
MyFirstVPC	vpc-0e6fb3010ecec9cc6	AWS::EC2::VPC	CREATE_COMPLETE
myInternetGateway	igw-0b2e2d0f3e4b8207e	AWS::EC2::InternetGateway	CREATE_COMPLETE
		VPC-k-myRoute	CREATE_COMPLETE
			CREATE_COMPLETE

Create a template to create three resources:  
Third Resource:  
• create an Internet Gateway, attach it to the VPC, and add a route for 0.0.0.0/0.

# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

- ① Create public subnets in two Availability Zones (AZ). Create a template to create three resources:

Create VPC  
environment

```
Outputs:  
StackVPC:  
  Description: The ID of the VPC  
  Value: !Ref MyFirstVPC  
  Export:  
    Name: !Sub "${AWS::StackName}-VPCID"  
  
StackPublicSubnet1A:  
  Description: The ID of the VPC Subnet  
  Value: !Ref PublicSubnet1A  
  Export:  
    Name: !Sub "${AWS::StackName}-PublicSubnet1A"  
  
StackPublicSubnet1C:  
  Description: The ID of the VPC Subnet  
  Value: !Ref PublicSubnet1C  
  Export:  
    Name: !Sub "${AWS::StackName}-PublicSubnet1C"
```

1. export the ID of the created VPC.

2. export the ID of the created  
PublicSubnet1A.

3. export the ID of the created  
PublicSubnet1C.

Exported IDs can be  
used in other  
CloudFormation  
stacks

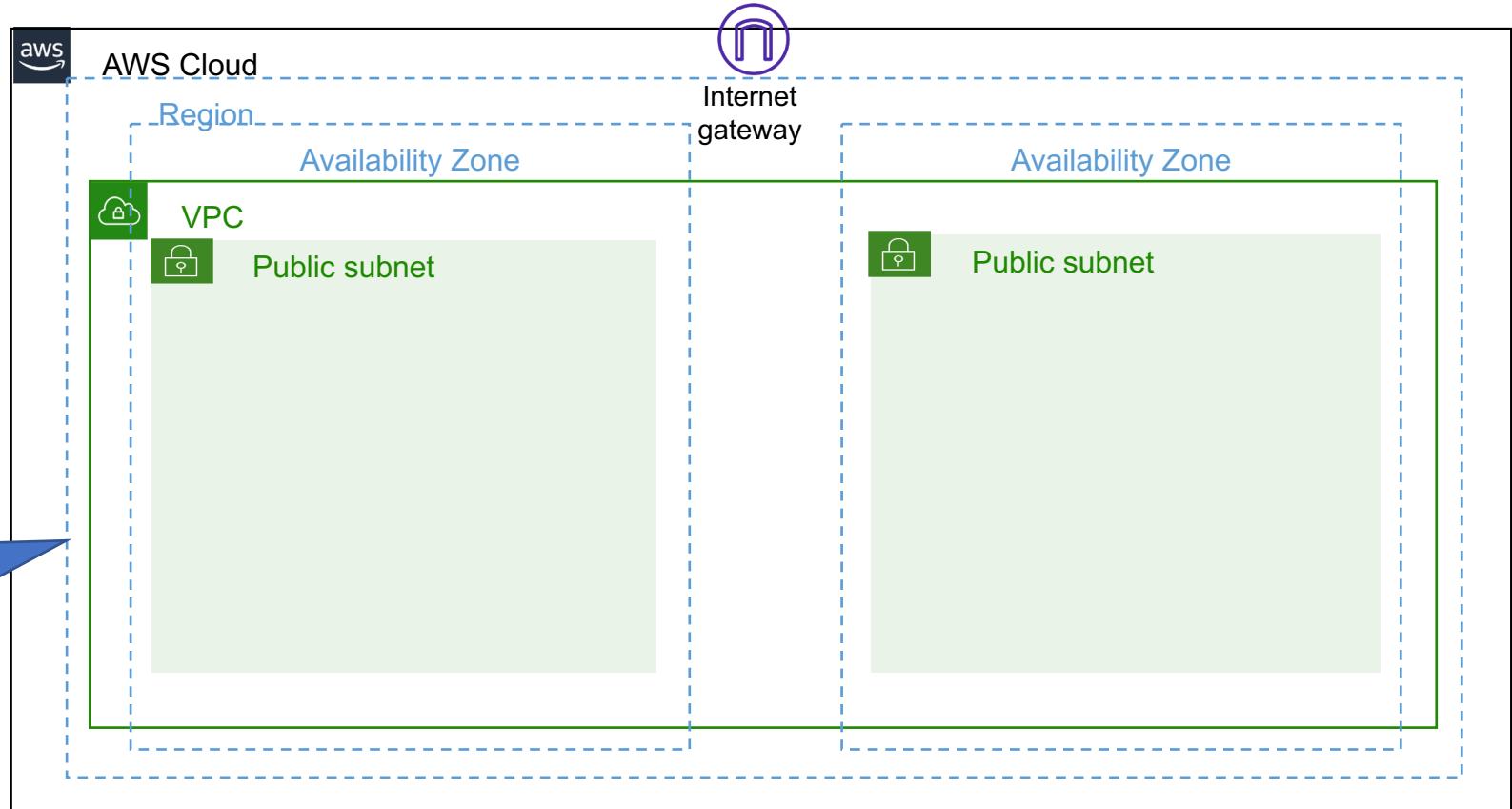
# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

- ① Create public subnets in two Availability Zones (AZ). Create a template to create three resources:

Create VPC  
environment

Now it looks like this



# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

- ②-a: Deploy EC2 instances in each public subnet.

```
MyEC2Instance1:  
  Type: AWS::EC2::Instance  
  Properties:  
    ImageId: !Ref EC2AMI  
    InstanceType: t2.micro  
    SubnetId: !ImportValue VPC-koutiku-PublicSubnet1A  
    UserData: !Base64  
      #!/bin/bash  
      systemctl start httpd.service  
  BlockDeviceMappings:  
    - DeviceName: /dev/xvda  
      Ebs:  
        VolumeType: gp2  
        VolumeSize: 8  
  Tags:  
    - Key: Name  
      Value: CF-public-1a  
  KeyName: !Ref KeyName  
  SecurityGroupIds:  
    - !GetAtt "InstanceSecurityGroup.GroupId"
```

Refer to the SubnetId from the stack where the VPC was built.

Note: Output the SubnetId beforehand.

Create a separate stack to create EC2 instances.

リソース (9)				
論理 ID	物理 ID	タイプ	ステータス	
lancer/app/LancerAppElb	ngvz::LoadBalancer			
MyEC2Instance1	i-040184ad8ba599e14	AWS::EC2::Instance	CREATE_COMPLETE	
MyEC2Instance2	i-0ed1dcad88a37ae39	AWS::EC2::Instance	CREATE_COMPLETE	
Kada4ACM				

EC2 instance is created as a CloudFormation resource

# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

- ②-a: Deploy EC2 instances in each public subnet.

```
MyEC2Instance2:  
  type: AWS::EC2::Instance  
  Properties:  
    ImageId: !Ref EC2AMI  
    InstanceType: t2.micro  
    SubnetId: !ImportValue VPC-koutiku-PublicSubnet1C  
    UserData: !Base64 |  
      #!...  
      #!/bin/bash  
      echo "Hello, CloudFormation!">/etc/motd
```

Refer to the SubnetId from the stack where the VPC was built.

Note: Output the SubnetId beforehand.

```
  systemctl start httpd.service  
  BlockDeviceMappings:  
    - DeviceName: /dev/xvda  
      Ebs:  
        VolumeType: gp2  
        VolumeSize: 8  
    Tags:  
      - Key: Name  
        Value: CF-public-1c  
    KeyName: !Ref KeyName  
    SecurityGroupIds:  
      - !GetAtt "InstanceSecurityGroup.GroupId"
```

リソース (9)				
論理 ID	物理 ID	タイプ	ステータス	
Kancer/app/CF-Subnet-A-ELB	elb/56f09e2d86a660d4	aws::LoadBalancer	CREATE_COMPLETE	
Kancer/app/CF-Subnet-B-ELB	elb/56f09e2d86a660d4	aws::LoadBalancer	CREATE_COMPLETE	
arn:aws:acm:ap-				
KadaI4ACM				
MyEC2Instance1		AWS::EC2::Instance	CREATE_COMPLETE	
MyEC2Instance2	i-0ed1dcad88a37ae39	AWS::EC2::Instance	CREATE_COMPLETE	

EC2 instance is created as a CloudFormation resource

# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

②-b: Use Apache as the web server

```
AWSTemplateFormatVersion: "2010-09-09"
Description: Create EC2 Instance
Parameters:
  KeyName:
    Description: Name of an existing EC2 KeyPair.
    Type: AWS::EC2::KeyPair::KeyName
    ConstraintDescription: Can contain only ASCII characters.
  EC2AMI:
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2
```

To ensure Apache is available, use the Amazon Linux 2 AMI. Utilize AWS Systems Manager Parameter Store to fetch the latest Amazon Linux AMI.

```
MyEC2Instance1:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref EC2AMI
    InstanceType: t2.micro
    SubnetId: !ImportValue VPC-koutiku-Subnet1C
    UserData: !Base64 |
      #!/bin/bash
      yum update -y
      yum -y install httpd
      touch /var/www/html/index.html
      chown apache:apache index.html
      systemctl enable httpd.service
      systemctl start httpd.service
MyEC2Instance2:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref EC2AMI
    InstanceType: t2.micro
    SubnetId: !ImportValue VPC-koutiku-PublicSubnet1C
    UserData: !Base64 |
      #!/bin/bash
      yum update -y
      yum -y install httpd
      touch /var/www/html/.check_alive
      chown apache:apache -R /var/www/html
      systemctl enable httpd.service
      systemctl start httpd.service
```

Install Apache using the Yum command:  
yum -y install httpd

# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

③: Use ALB and configure each EC2 instance in the target group.

```
InternetELB:  
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer  
  Properties:  
    Name: CF-elb  
    SecurityGroups:  
      - !Ref InstanceSecurityGroup  
    Subnets:  
      - !ImportValue VPC-koutiku-PublicSubnet1A  
      - !ImportValue VPC-koutiku-PublicSubnet1C
```

```
ELBtargetGP:  
  Type: AWS::ElasticLoadBalancingV2::TargetGroup  
  Properties:  
    Name: ELBtargetGP  
    Port: 80  
    Protocol: HTTP  
    Targets:  
      - Id: !Ref MyEC2Instance1  
      - Id: !Ref MyEC2Instance2  
    TargetType: instance  
    VpcId: !ImportValue VPC-koutiku-VPCID
```

③ Configure the ALB

InternetELB

- configure the ALB.
- set the ALB's security group to the one specified for EC2.
- import the Subnets Output from the stack where the VPC was built.

ELBtargetGP

- configure the ALB's target group.
- specify Port 80 with the HTTP protocol.
- specify the EC2 instances to which the ELB forwards requests.

# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

③ Use ALB and configure each EC2 instance in the target group.

## ③ Configure the ALB

```
ELBLListenerHTTP: # 80=>443に転送
  Type: AWS::ElasticLoadBalancingV2::Listener
  DependsOn:
    - InternetELB
  Properties:
    DefaultActions:
      - Type: redirect
        RedirectConfig:
          Protocol: HTTPS
          Port: 443
          Host: '#{host}'
          Path: '/#{path}'
          Query: '#{query}'
          StatusCode: HTTP_301
  LoadBalancerArn: !Ref InternetELB
  Port: 80
  Protocol: HTTP
```

```
ELBLListenerHTTPS: # SSL設定してTargetGroupに転送
  Type: AWS::ElasticLoadBalancingV2::Listener
  DependsOn: InternetELB
  Properties:
    Certificates:
      - CertificateArn: !Ref Kadai4ACM
  DefaultActions:
    - TargetGroupArn: !Ref ELBtargetGP
      Type: forward
    LoadBalancerArn: !Ref InternetELB
    Port: 443
    Protocol: HTTPS
    SslPolicy: ELBSecurityPolicy-2016-08
  ListenerRule1:
    Type: AWS::ElasticLoadBalancingV2::ListenerRule
    Properties:
      ListenerArn: !Ref ELBLListenerHTTPS
      Priority: 2
      Conditions:
        - Field: host-header
          HostHeaderConfig:
            Values:
              - !Ref DomainName
      Actions:
        - Type: forward
          TargetGroupArn: !Ref ELBtargetGP
```

1 ELBLListenerHTTP:  
• Create a listener for HTTP port 80.  
• Set up HTTPS redirect.

2 ELBLListenerHTTPS:  
• Create a listener for HTTPS port 443.  
• Specify an SSL certificate and configure forwarding to the target group.

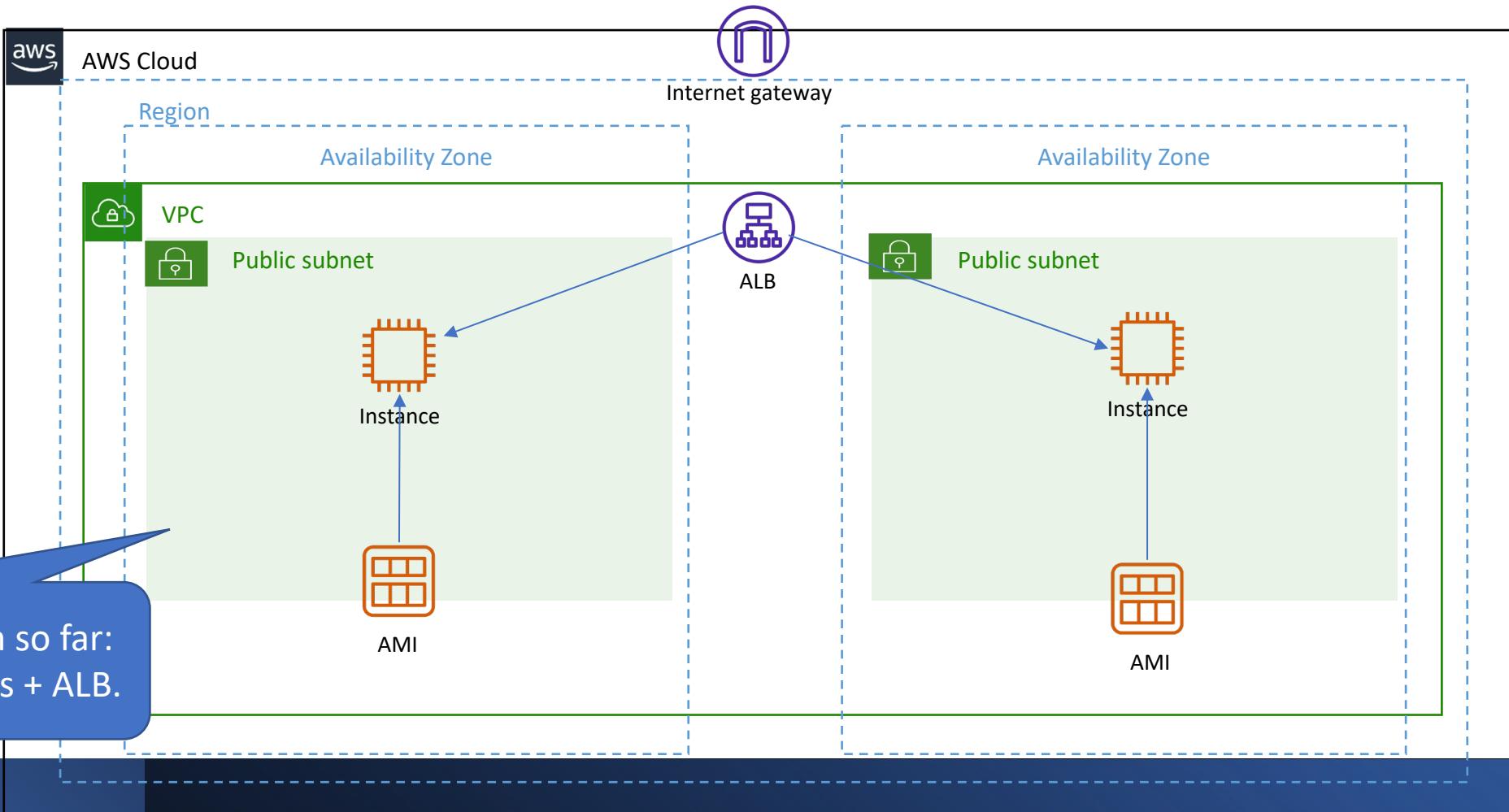
3 ListenerRule1:  
• Create a listener rule.  
• Configure forwarding to the target group if the Host header of the HTTP request matches the DomainName.

# 3. Implementation

Requirement 1: Automatically build an environment that fulfills the following conditions:

- ③ Use ALB and configure each EC2 instance in the target group.

Set up EC2 and ALB in the VPC environment.



# 3. Implementation

Requirement 2: Enable secure encrypted communication using a domain for the web page.

## Preparation Step ①: Setting up Route53

### Steps

- create a hosted zone in Route53.
- purchase a domain name from "お名前.com" (Onamae.com)
- issue a certificate in ACM (AWS Certificate Manager)

Route 53 > ホストゾーン

ホストゾーン (1)

Automatic モードは最適なフィルタ結果に最適化された現在の検索動作です。モードを変更するには、[設定] に移動します。

C 詳細を表示 編集 削除 ホストゾーンの作成

Q プロパティまたは値でレコードをフィルタリングする

ホストゾーン名	タイプ	作成者	レコード数	説明	ホストゾーン ID
yumetech.net	パブリック	Route 53	5	For project 4	Z03917251J40DXFLPJW6J

During the preparation phase, there will be two records:

- NS record
- SOA record

# 3. Implementation

Requirement 2: Enable secure encrypted communication using a domain for the web page.

preparation Step ②: Purchase a domain name from "お名前.com" (Onamae.com)

Route53

レコード (5) 情報  
Automatic モードは最適なフィルタ結果に最適化された現在の検索動作です。モードを変更するには、[設定] に移動します。

レコードを削除 ゾーンファイルをインポート レコードを作成

プロパティまたは値でレコードをフィルタリングする NS ルーティング エイリアス

タイプ = NS フィルタのクリア

レコード名	タ...	ルーティング	差別化	エイリアス
yumetech.net	NS	シンプル	-	いいえ

値/トラフィックのルーティング

- ns-1213.awsdns-23.org.
- ns-22.awsdns-02.com.
- ns-882.awsdns-46.net.
- ns-1562.awsdns-03.co.uk.

Register the value of the NS record for "yumetech.net" as the nameserver for "お名前.com" (Onamae.com).

ドメイン名 yumetech.net お名前.com

ネームサーバー情報

ネームサーバー1	ns-1213.awsdns-23.org
ネームサーバー2	ns-22.awsdns-02.com
ネームサーバー3	ns-882.awsdns-46.net
ネームサーバー4	ns-1562.awsdns-03.co.uk

# 3. Implementation

Requirement 2: Enable secure encrypted communication using a domain for the web page.

Preparation Step ③:  
Issue a certificate in ACM

Parameters:
<b>KeyName:</b>
Description: Name of an existing EC2 KeyPair.
Type: AWS::EC2::KeyPair::KeyName
ConstraintDescription: Can contain only ASCII characters.
<b>EC2AMI:</b>
Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
Default: /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2
<b>HostZoneId:</b>
Description: FQDN of the hosted zone.
Type: String
Default: 'Z03917251J40DXFLPJW6J'
<b>DomainName:</b>
Description: FQDN of the HostZone yumetech.net

Route 53 > ホストゾーン

ホストゾーン (1)

Automatic モードは最適なフィルタ結果に最適化された現在の検索動作です。モードを変更するには、[設定] に移動します。

C 詳細を表示 編集 削除 ホストゾーンの作成

Q プロパティまたは値でレコードをフィルタリングする

ホストゾーン名	タイプ	作成者	レコード数	説明	ホストゾーン ID
yumetech.net	パブリック	Route 53	5	For project 4	Z03917251J40DXFLPJW6J

Set the Hosted Zone ID to Default.

# 3. Implementation

Requirement 2: Enable secure encrypted communication using a domain for the web page.

Preparation Step ③:  
Issue a certificate in ACM

**HostZoneId:**  
Description: FQDN of the hosted zone.  
Type: String  
Default: 'Z03917251J40DXFLPJW6J'

**DomainName:**  
Description: FQDN of the HostZone yumetech.net  
Type: String  
Default: yumetech.net

**SubjectAlternativeName:**  
Description: FQDN of the Subdomains.  
Type: String  
Default: '\*.yumetech.net'

**SubDomain:**  
Description: FQDN of the Subdomain  
Type: String  
Default: www.yumetech.net

1

1: Define an FQDN to be protected by ACM.

2

2: Define additional FQDN

3

3: Define a subdomain.

Ensure users can access  
www.yumetech.net

# 3. Implementation

Requirement 2: Enable secure encrypted communication using a domain for the web page.

```
Domain1Record: # yumetech.net
  Type: AWS::Route53::RecordSetGroup
  Properties:
    HostedZoneId: !Sub ${HostZoneId}
    Comment: Zone apex alias targeted to public LoadBalancer.
  RecordSets:
    - Name: !Sub ${DomainName}
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt InternetELB.CanonicalHostedZoneID
        DNSName: !GetAtt InternetELB.DNSName

Domain1SubDomainRecord: # www.yumetec.net
  Type: AWS::Route53::RecordSet
  DependsOn: InternetELB
  Properties:
    HostedZoneId: !Sub '${HostZoneId}'
    Name: !Sub '${SubDomain}'
    Type: CNAME
    TTL: 60
  ResourceRecords:
    - !GetAtt InternetELB.DNSName

#
# -----
# ACM
# -----
```

Kadai4ACM:

```
  Type: AWS::CertificateManager::Certificate
  Properties:
    DomainName: !Sub '${DomainName}'
    SubjectAlternativeNames:
      - !Sub ${SubjectAlternativeName}
    DomainValidationOptions:
      - DomainName: !Sub '${DomainName}'
        HostedZoneId: !Sub '${HostZoneId}'
    ValidationMethod: DNS
```

1

2

3

Preparation Step ③:  
Issue a certificate in ACM

1: Create a Type A record.

2: Create a CNAME record.

3: Issue ACM.

# 3. Implementation

Requirement 2: Enable secure encrypted communication using a domain for the web page.

Preparation Step ③:  
Issue a certificate in ACM

Using the same YAML file template for EC2, Route53, etc., execute ACM issuance in the same stack.

論理 ID	物理 ID	タイプ	ステータス
Kadai4ACM	arn:aws:acm:ap-northeast-1:234056973550:certificate/5d320949-7796-40aa-94a2-9862beb9be96	AWS::CertificateManager::Certificate	CREATE_COMPLETE
ListenerRule1	arn:aws:elasticloadbalancing:ap-northeast-1:234056973550:listener-rule/app/CF-elb/9fa17c4300c2a203/ff10067331272b86/d37bec81b3219f81	AWS::ElasticLoadBalancingV2::ListenerRule	CREATE_COMPLETE
MyEC2Instance1	i-07e8d605733b5cbc9	AWS::EC2::Instance	CREATE_COMPLETE

# 3. Implementation

Requirement 3: Enable automatic scaling out when the CPU load exceeds 70% and scaling in when it falls below 20%.

The screenshot shows the AWS CloudFormation console for a stack named 'kada14'. It displays two EC2 instances and their configurations:

- EC2 Instance 1 (Left):** AMI ID: ami-05c7416867898f878. Security group: sg-0aade5df201e0bd55 (CF-instance-InstanceSecurityGroup).
- EC2 Instance 2 (Right):** AMI ID: ami-05c7416867898f878. Key pair assigned at launch: mykeypair.

A blue callout box highlights the following steps:

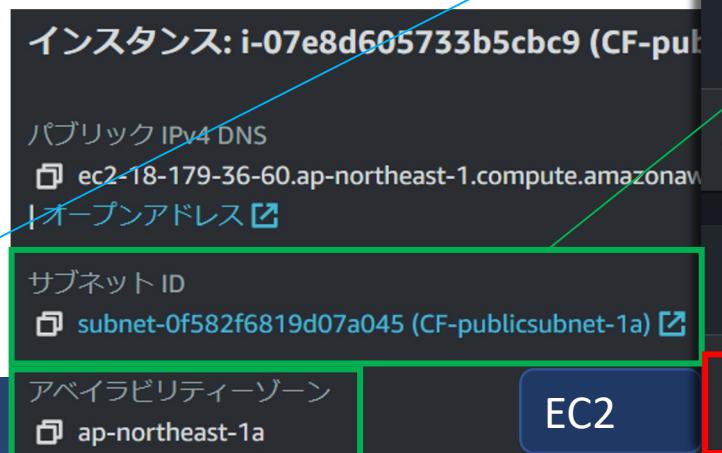
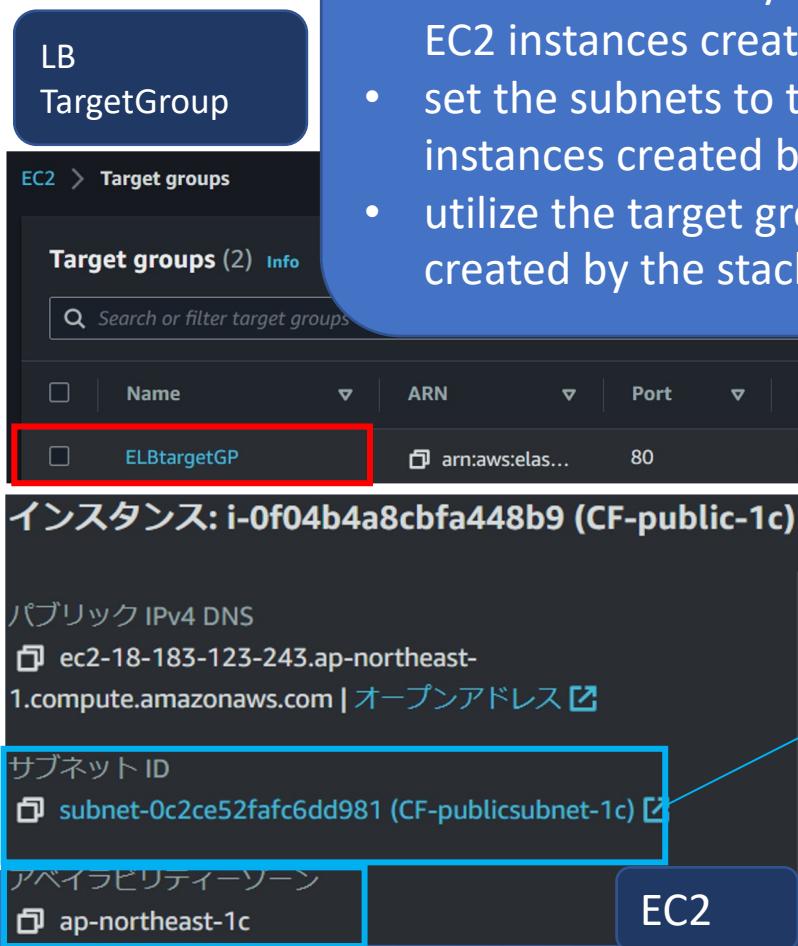
- Create a launch template necessary for creating the Auto Scaling Group:
- Use the AMI ID, security group, and key pair automatically created by the stack for EC2.

The screenshot also shows the '起動テンプレートのバージョン' (Launch Template Version) section of the CloudFormation stack, which includes fields for Version (1 (デフォルト)), Description (for kada14 web server), and the Launch Template itself, which lists the AMI ID, instance type (t2.micro), and security group (sg-0aade5df201e0bd55). A red arrow points from the AMI ID in the CloudFormation stack to the AMI ID in the Launch Template.

# 3. Implementation

Requirement 3: Enable automatic scaling out when the CPU load exceeds 70% and scaling in when it falls below 20%.

- configure the Auto Scaling Group (ASG):
- set the Availability Zones (AZs) to the AZ where the EC2 instances created by the stack are located.
- set the subnets to the subnet where the EC2 instances created by the stack are located.
- utilize the target group for the Load Balancer (LB) created by the stack.



# 3. Implementation

Requirement 3: Enable automatic scaling out when the CPU load exceeds 70% and scaling in when it falls below 20%.

The screenshot shows the AWS Auto Scaling Groups interface for the group 'asg-kadai4'. It displays two scaling policies:

- kadai4-cpu-high**: Simple scaling, active. It triggers when CPUUtilization > 70 over 300 seconds. The 'AutoScalingGroupName' is asg-kadai4. A red box highlights the '追加 1 容量ユニット' (Add 1 capacity unit) button.
- kadai4-cpu-low**: Simple scaling, active. It triggers when CPUUtilization < 20 over 300 seconds. The 'AutoScalingGroupName' is asg-kadai4. A red box highlights the '削除 1 容量ユニット' (Delete 1 capacity unit) button.

Both policies have a '別の規模の拡大や縮小を許可する前に 300 秒' (Allow other scale changes for 300 seconds) note at the bottom.

For dynamic scaling policies, set up Simple Scaling Policies:

- when CPU utilization exceeds 70%, add one EC2 instance (Scale Out).
- when CPU utilization falls below 20%, remove one EC2 instance (Scale In).

# 3. Implementation

Requirement 4: Set up alerts to be sent via email to responsible personnel when thresholds are exceeded in monitoring.

CloudWatch > アラーム

アラーム (2)

検索

Auto Scaling アラームを非表示

選択をクリア C 複合アラームの作成 アクション アラームの作成

名前	状態	最終状態の更新	条件	アクション
kada4-low-20	△ アラーム状態	2023-05-03 20:37:49	5 分内の1データポイントのCPUUtilization < 20	アクションが有効になっています
kada4-cpu-high-70	○ OK	2023-05-03 12:38:24	5 分内の1データポイントのCPUUtilization > 70	アクションが有効になっています

アラーム (1)

名前: kada4-cpu-high-70

状態: ○ OK

しきい値: 5 分内の1データポイントの CPUUtilization > 70

説明: 説明がありません

最終変更: 2023-05-03 03:38:24

アクション: アクションが有効になっています

名前空間: AWS/EC2

メトリクス名: CPUUtilization

AutoScalingGroupName: asg-kada4

統計: 平均値 evaluate

期間: 5 分

ARN: arn:aws:cloudwatch:ap-northeast-1:234056973550:alarm:kada4-cpu-high-70

Configure an alert to be triggered when CPU utilization exceeds 70%:

- metric: CPUUtilization
- threshold: CPUUtilization > 70
- associate with ASG

# 3. Implementation

Requirement 4: Set up alerts to be sent via email to responsible personnel when thresholds are exceeded in monitoring.

CloudWatch > アラーム

アラーム (2)

検索

名前 状態 最終状態の更新 條件 アクション

名前	状態	最終状態の更新	条件	アクション
kadai4-low-20	△ アラーム状態	2023-05-03 20:37:49	5 分内の1データポイントのCPUUtilization < 20	アクションが有効になっています
kadai4-cpu-high-70	○ OK	2023-05-03 12:38:24	5 分内の1データポイントのCPUUtilization > 70	アクションが有効になっています

アラームの作成

検索

任意の状態 ▾ 任意のタイプ ▾ すべてのア... ▾

アラームの詳細

名前: kadai4-low-20

状態: アラーム状態

しきい値: 5 分内の1データポイントのCPUUtilization < 20

メトリクス名: CPUUtilization

AutoScalingGroupName: asg-kadai4

最終変更: 2023-05-03 11:37:49

アクション: アクションが有効になっています

統計: 平均値

期間: 5 分

名前空間: AWS/EC2

アラームを実行するデータポイント: 1/1

欠落データの処理: 欠落データを見つかりませんとして処理

サンプル数が少ないパーセンタイル: evaluate

ARN: arn:aws:cloudwatch:ap-northeast-1:234056973550:alarm:kadai4-low-20

Configure an alert to be triggered when CPU utilization falls below 20%:

- metric: CPUUtilization
- threshold: CPUUtilization < 20
- associate with ASG

# 3. Implementation

Requirement 4: Set up alerts to be sent via email to responsible personnel when thresholds are exceeded in monitoring.

The screenshot shows the AWS Auto Scaling Groups interface for the group 'asg-kadai4'. It displays two scaling policies:

- kadai4-cpu-high**:
  - シンプルなスケーリング
  - 有効
  - kadai4-cpu-high-70**:
    - アラームのしきい値を超過: 次のメトリクスディメンションに対して 300 秒間の 1 連続期間で CPUUtilization > 70:
    - AutoScalingGroupName = asg-kadai4
  - 追加 1 容量ユニット
  - 別の規模の拡大や縮小を許可する前に 300 秒
- kadai4-cpu-low**:
  - シンプルなスケーリング
  - 有効
  - kadai4-low-20**:
    - アラームのしきい値を超過: 次のメトリクスディメンションに対して 300 秒間の 1 連続期間で CPUUtilization < 20:
    - AutoScalingGroupName = asg-kadai4
  - 削除 1 容量ユニット
  - 別の規模の拡大や縮小を許可する前に 300 秒

When setting up the dynamic scaling policies with Simple Scaling Policy in ASG, you can confirm that CloudWatch alarms are configured.

# 3. Implementation

Requirement 4: Set up alerts to be sent via email to responsible personnel when thresholds are exceeded in monitoring.

To enable alert notifications via email:

- Create an SNS topic.
  - Topic Type: Standard
  - Endpoint: Email Address

ID	エンドポイント	ステータス	プロトコル
cafd2e98-4e47-43f7-aaaa-5...	c-izumeru@yume-tec.com	確認済み	EMAIL

When configuring CloudWatch alarms, ensure that alarms are set up to notify the SNS topic. Configure notifications to the SNS topic for both CPU utilization thresholds: below 20% and above 70%.

CloudWatch > アラーム > アラームの作成

## アクションの設定

### 通知

アラーム状態トリガー  
このアクションをトリガーするアラームの状態

アラーム状態  
メトリクスまたは式が定義された値の範囲外にあります

次の SNS トピックに通知を送信する通知を受信する SNS (Simple Notification Service)

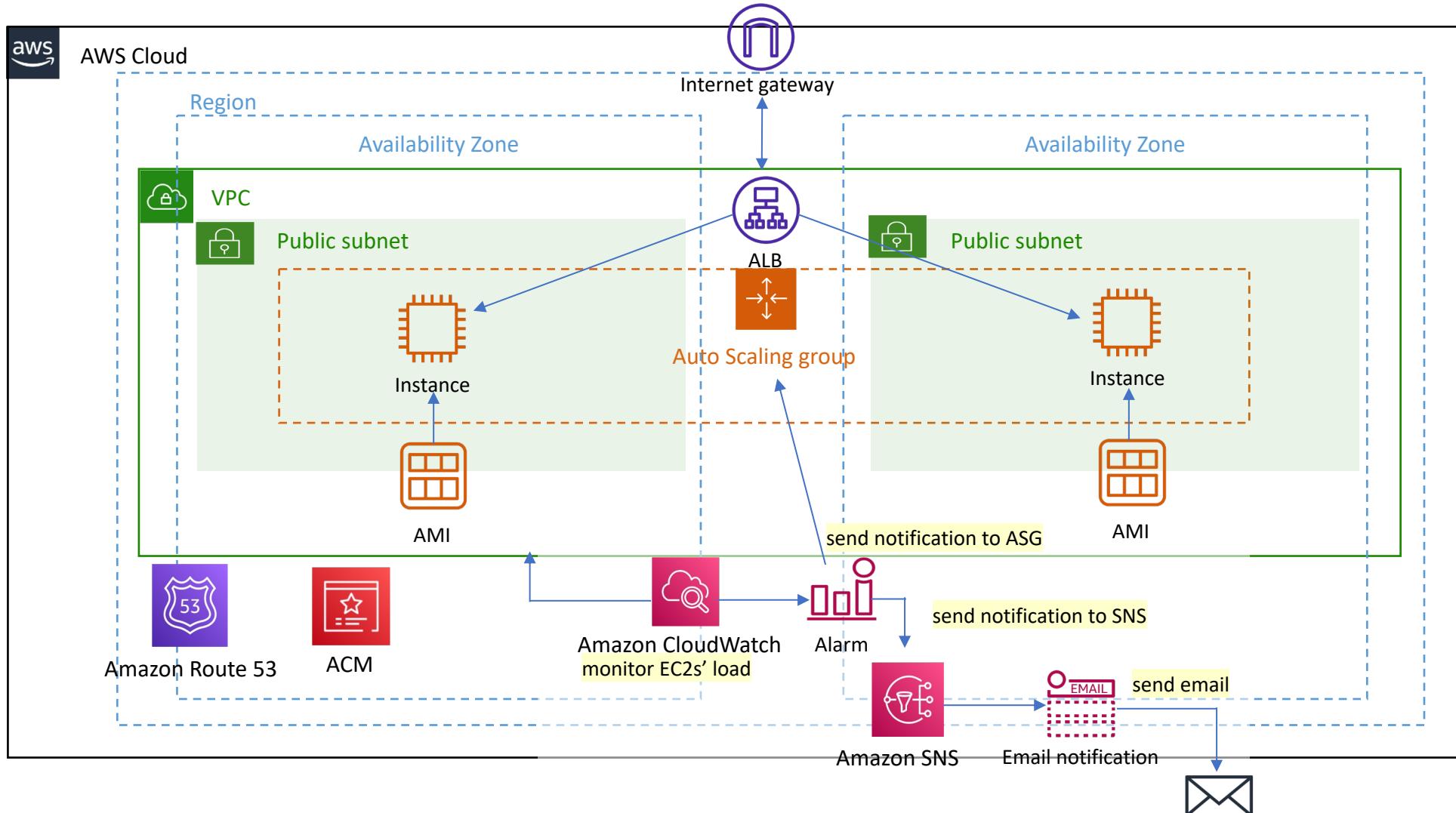
既存の SNS トピックを選択

新しいトピックの作成

トピック ARN を使用して他のアカウントで通知を受信する

通知の送信先:  
Kadai04\_CloudWatch\_Alarms\_Topic

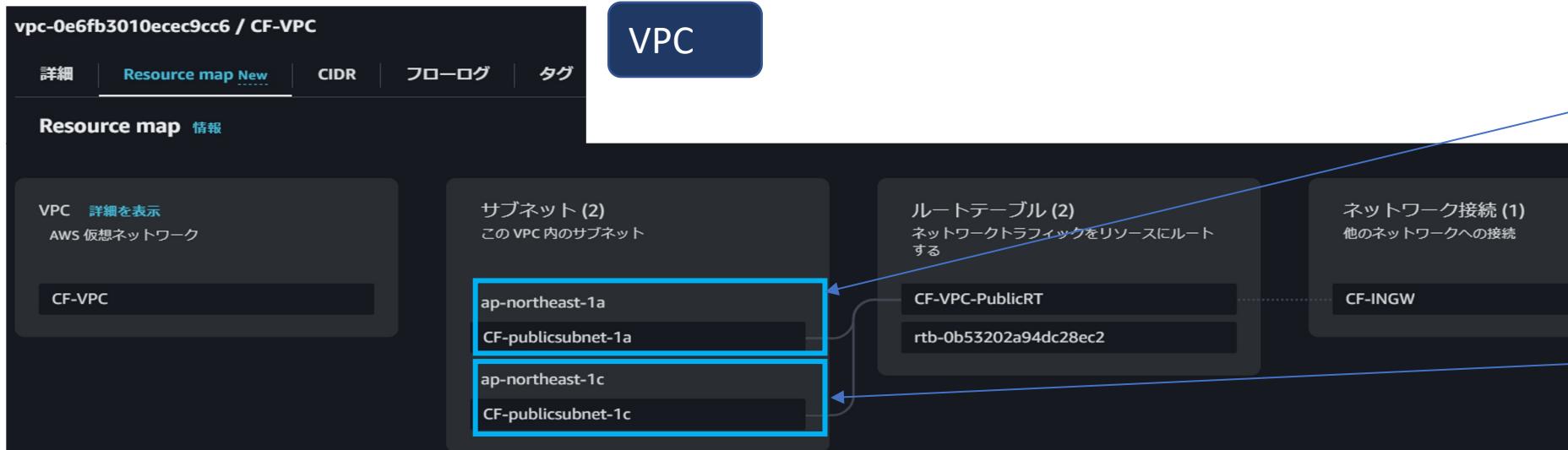
# 3. Implementation



# 4. Test Results:

Requirement 1: Automatically build an environment that fulfills the following conditions:

1. Create public subnets in two Availability Zones (AZs).



PublicSubnet1A:
Type: AWS::EC2::Subnet
Properties:
VpcId: !Ref MyFirstVPC
CidrBlock: 10.0.0.0/24
MapPublicIpOnLaunch: true
AvailabilityZone: "ap-northeast-1a"
Tags:
- Key: Name
Value: CF-publicsubnet-1a

PublicSubnet1C:
Type: AWS::EC2::Subnet
Properties:
VpcId: !Ref MyFirstVPC
CidrBlock: 10.0.2.0/24
MapPublicIpOnLaunch: true
AvailabilityZone: "ap-northeast-1c"
Tags:
- Key: Name
Value: CF-publicsubnet-1c

Resources:
MyFirstVPC:
Type: AWS::EC2::VPC
Properties:
CidrBlock: 10.0.0.0/16
EnableDnsSupport: 'true'
EnableDnsHostnames: 'true'
InstanceTenancy: default
Tags:
- Key: Name
Value: CF-VPC

詳細			
Resource map			
CIDR			
フローログ			
タグ			
詳細			
VPC ID vpc-0e6fb3010ecec9cc6	状態 Available	DNS ホスト名 有効	DNS 解決 有効
デナシ Default	DHCP オプションセット dopt-00499efc7d1821d91	メインルートテーブル rtb-0b53202a94dc28ec2	メインネットワーク ACL acl-0add4399538c500469
デフォルト VPC いいえ	IPv4 CIDR 10.0.0.0/16	IPv6 プール -	IPv6 CIDR (ネットワークポーダーグループ) -
ネットワークアドレスの使用状況メトリクス 無効	Route 53 リソルバー DNS ファイアウォールルール グループ -	所有者 ID 234056973550	-

# 4. Test Results:

Requirement 1: Automatically build an environment that fulfills the following conditions:

1. Create public subnets in two Availability Zones (AZs).

②-a: Deploy EC2 instances in each public subnet.

PublicSubnet1A:

```
Type: AWS::EC2::Subnet
Properties:
  VpcId: !Ref MyFirstVPC
  CidrBlock: 10.0.0.0/24
  MapPublicIpOnLaunch: true
  AvailabilityZone: "ap-northeast-1a"
  Tags:
    - Key: Name
      Value: CF-publicsubnet-1a
```

PubSubnet1ARouteTableAssociation:

```
Type: AWS::EC2::SubnetRouteTableAssociation
Properties:
  SubnetId: !Ref PublicSubnet1A
  RouteTableId: !Ref PublicRouteTable
```

PublicSubnet1C:

```
Type: AWS::EC2::Subnet
Properties:
  VpcId: !Ref MyFirstVPC
  CidrBlock: 10.0.2.0/24
  MapPublicIpOnLaunch: true
  AvailabilityZone: "ap-northeast-1c"
  Tags:
    - Key: Name
      Value: CF-publicsubnet-1c
```

Resources:

```
# -----
# EC2 for pub sub 1A
#
MyEC2Instance1:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref EC2AMI
    InstanceType: t2.micro
    SubnetId: !ImportValue VPC-koutiku-PublicSubnet1A
```

```
# -----
# EC2 for pub sub 1C
#
MyEC2Instance2:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref EC2AMI
    InstanceType: t2.micro
    SubnetId: !ImportValue VPC-koutiku-PublicSubnet1C
```

subnet

The EC2 instances are deployed in two different subnets as specified in the template.

インスタンス: i-07e8d605733b5cbc9 (CF-public-1a)

▼ ネットワーキングの詳細 情報

パブリック IPv4 アドレス  
□ 18.179.36.60 | オープンアドレス

パブリック IPv4 DNS  
□ ec2-18-179-36-60.ap-northeast-1.compute.amazonaws.com | オープンアドレス

サブネット ID

□ subnet-0f582f6819d07a045 (CF-publicsubnet-1a)

アベイラビリティーゾーン

□ ap-northeast-1a

インスタンス: i-0f04b4a8cbfa448b9 (CF-public-1c)

▼ ネットワーキングの詳細 情報

パブリック IPv4 アドレス  
□ 18.183.123.243 | オープンアドレス

パブリック IPv4 DNS  
□ ec2-18-183-123-243.ap-northeast-1.compute.amazonaws.com | オープンアドレス

サブネット ID

□ subnet-0c2ce52fafc6dd981 (CF-publicsubnet-1c)

アベイラビリティーゾーン

□ ap-northeast-1c

# 4. Test Results:

Requirement 1: Automatically build an environment that fulfills the following conditions:

1. Create public subnets in two Availability Zones (AZs).

②-a: Deploy EC2 instances in each public subnet.

```
PublicSubnet1A:  
  Type: AWS::EC2::Subnet  
  Properties:  
    VpcId: !Ref MyFirstVPC  
    CidrBlock: 10.0.0.0/24  
    MapPublicIpOnLaunch: true  
    AvailabilityZone: "ap-northeast-1a"  
  Tags:  
    - Key: Name  
      Value: CF-publicsubnet-1a
```

```
PubSubnet1ARouteTableAssociation:  
  Type: AWS::EC2::SubnetRouteTableAssociation  
  Properties:  
    SubnetId: !Ref PublicSubnet1A  
    RouteTableId: !Ref PublicRouteTable
```

```
PublicSubnet1C:  
  Type: AWS::EC2::Subnet  
  Properties:  
    VpcId: !Ref MyFirstVPC  
    CidrBlock: 10.0.2.0/24  
    MapPublicIpOnLaunch: true  
    AvailabilityZone: "ap-northeast-1c"  
  Tags:  
    - Key: Name  
      Value: CF-publicsubnet-1c
```

```
Resources:  
  # -----  
  # EC2 for pub sub 1A  
  # -----  
  MyEC2Instance1:  
    Type: AWS::EC2::Instance  
    Properties:  
      ImageId: !Ref EC2AMI  
      InstanceType: t2.micro  
      SubnetId: !ImportValue VPC-koutiku-PublicSubnet1A  
  
  # -----  
  # EC2 for pub sub 1C  
  # -----  
  MyEC2Instance2:  
    Type: AWS::EC2::Instance  
    Properties:  
      ImageId: !Ref EC2AMI  
      InstanceType: t2.micro  
      SubnetId: !ImportValue VPC-koutiku-PublicSubnet1C
```

The EC2 instances are launched in two different AZs as specified in the template.



インスタンス (2) 情報						
属性またはタグ (case-sensitive) で インスタンスを検索						
CF-public-1		X	フィルターをクリア			
□	Name	▼	インスタンス ID	インスタン... ▼	イ... ▼	ステータスチェック
□	CF-public-1a		i-07e8d605733b5cbc9	実行中	t2.micro	2/2 のチェックに合格
□	CF-public-1c		i-0f04b4a8cbfa448b9	実行中	t2.micro	2/2 のチェックに合格

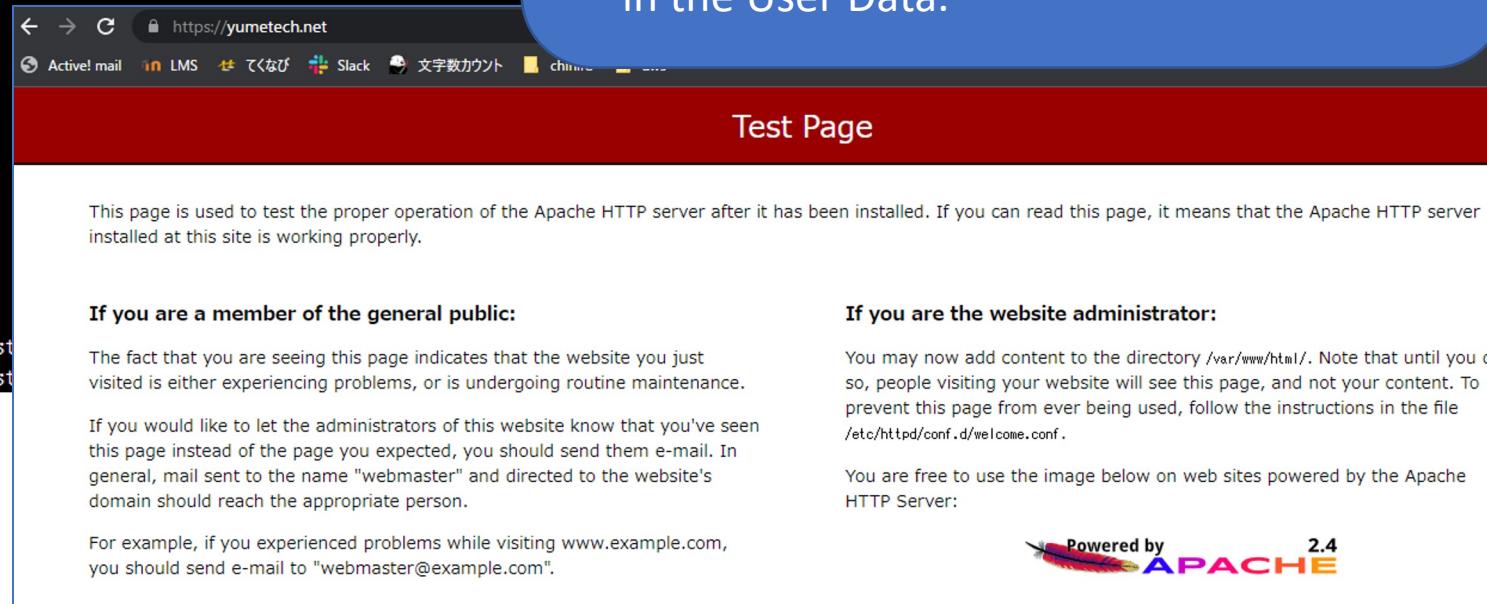
# 4. Test Results:

Requirement 1: Automatically build an environment that fulfills the following conditions:

1. Create public subnets in two Availability Zones (AZs).

②-b: Use Apache as the web server

```
https://aws.amazon.com/amazon-linux-2/  
[ec2-user@ip-10-0-0-197 ~]$ sudo service httpd status  
Redirecting to /bin/systemctl status httpd.service  
● httpd.service - The Apache HTTP Server  
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)  
  Active: active (running) since Wed 2023-05-03 00:44:44 UTC; 1 day 3h ago  
    Docs: man:httpd.service(8)  
  Main PID: 6294 (httpd)  
    Status: "Total requests: 6816; Idle/Busy workers 100/0; Requests/sec: 0.0696; Bytes served/sec:  
  CGroup: /system.slice/httpd.service  
          ├ 6294 /usr/sbin/httpd -DFOREGROUND  
          ├ 6295 /usr/sbin/httpd -DFOREGROUND  
          ├ 6296 /usr/sbin/httpd -DFOREGROUND  
          ├ 6297 /usr/sbin/httpd -DFOREGROUND  
          ├ 6298 /usr/sbin/httpd -DFOREGROUND  
          ├ 6299 /usr/sbin/httpd -DFOREGROUND  
          ├ 6378 /usr/sbin/httpd -DFOREGROUND  
          ├ 6384 /usr/sbin/httpd -DFOREGROUND  
          ├ 6385 /usr/sbin/httpd -DFOREGROUND  
          └ 9076 /usr/sbin/httpd -DFOREGROUND  
  
May  3 00:44:43 ip-10-0-0-197.ap-northeast-1.compute.internal systemd[1]: Starting The Apache HTTP Server...  
May  3 00:44:44 ip-10-0-0-197.ap-northeast-1.compute.internal systemd[1]: Started The Apache HTTP Server.
```



- connected to the EC2 instance via EC2 Instance Connect
- executed the sudo service httpd status command.
- Apache is installed as per the script written in the User Data.

# 4. Test Results:

Requirement 1: Automatically build an environment that fulfills the following conditions:

1. Create public subnets in two Availability Zones (AZs).

③: Use ALB and configure each EC2 instance in the target group.

The screenshot shows the AWS Load Balancers console with one entry: CF-elb. The details are as follows:

Name	DNS name	Status	VPC ID	Availability Zones	Type	Date
CF-elb	CF-elb-2083295779.ap-northeast-1.elb.amazonaws.com (A Record)	Active	vpc-0e6fb3010ecec9cc6	2 Availability Zones	application	May 3, 2023, (UTC+09:00)

## Load Balancer

- Load Balancer Type: Application
- AZ: Configured to span across the AZs specified in the template.

The screenshot shows the configuration for the CF-elb load balancer. The target group is set to ELBtargetGP. The target type is Instance, and the protocol is HTTP. The target group details are as follows:

Target type	Protocol : Port	Protocol version
Instance	HTTP: 80	HTTP1

The screenshot shows the AWS Target Groups console with one entry: ELBtargetGP. The target type is Instance, and the protocol is HTTP. The target group details are as follows:

Target type	Protocol : Port	Protocol version
Instance	HTTP: 80	HTTP1

## Target Group

- Target Group set to Instance (EC2).

## 4. Test Results:

Requirement 2: Enable secure encrypted communication using a domain for the web page.

The screenshot shows the AWS EC2 Load Balancers console. The top navigation bar includes 'EC2 > Load balancers'. A blue box highlights the 'Load Balancer > Listener' section. The main area displays a table of load balancers, with 'CF-elb' selected. Below the table, the 'Load balancer: CF-elb' details page is shown, featuring a 'Default action' table. One row in this table, 'HTTP:80' with a 'Redirect to HTTPS://#{host}:443/#{path}?#{query}' rule, is highlighted with a red box.

Name	DNS name	State	VPC ID	Availability Zones	Type	Date created	Instances
CF-elb	CF-elb-20832957...	Active	vpc-0e6fb3010ecec9cc6	2 Availability Zones	application	May 3, 2023, 09:43 (UTC+09:00)	-

Protocol:Port	Default action	Rules	ARN	Security policy	CloudWatch	Metrics	Default SSL cert
HTTP:80	Redirect to HTTPS://#{host}:443/#{path}?#{query} • Status code: HTTP_301	1 rule	ARN	Not set	HTTP redirect count		
HTTPS:443	Forward to target group • ELBtargetGP: 1 (100%) • Group-level stickiness: Off	2 rules	ARN	ELBSe			

The Default Action for HTTP port 80 is set to redirect to HTTPS instead of forwarding to the target group.



Verified from CloudWatch metrics that redirection to HTTPS is occurring.

# 4. Test Results:

Requirement 2: Enable secure encrypted communication using a domain for the web page.

The screenshot shows the AWS Certificate Manager interface. At the top, a message indicates a certificate has been issued. Below it, a table lists two domain entries: 'yumetech.net' and '\*yumetech.net', both marked as successful. A red box highlights the 'ステータス' column for these entries. On the right, a detailed view of one entry shows the CNAME record 'yumetech.net' with a lock icon indicating it's protected by HTTPS. Another red box highlights this lock icon.

- An ACM certificate has been issued
- a CNAME for DNS validation has been set up.

The screenshot shows a web browser window displaying a test page from 'www.yumetech.net'. The address bar shows the URL 'https://www.yumetech.net'. A red box highlights the lock icon in the address bar, confirming a secure connection. The page content includes a heading 'Test Page' and some descriptive text about the Apache HTTP server.

- When accessing the DNS-validated domain name, we can confirm that it's an HTTPS connection.
- The subdomain www.yumetech.net is also configured for HTTPS connection.

この接続は保護されています  
Cookie とサイトデータ  
サイトの設定  
is either experiencing problems, or is undergoing routine maintenance.  
If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.  
For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

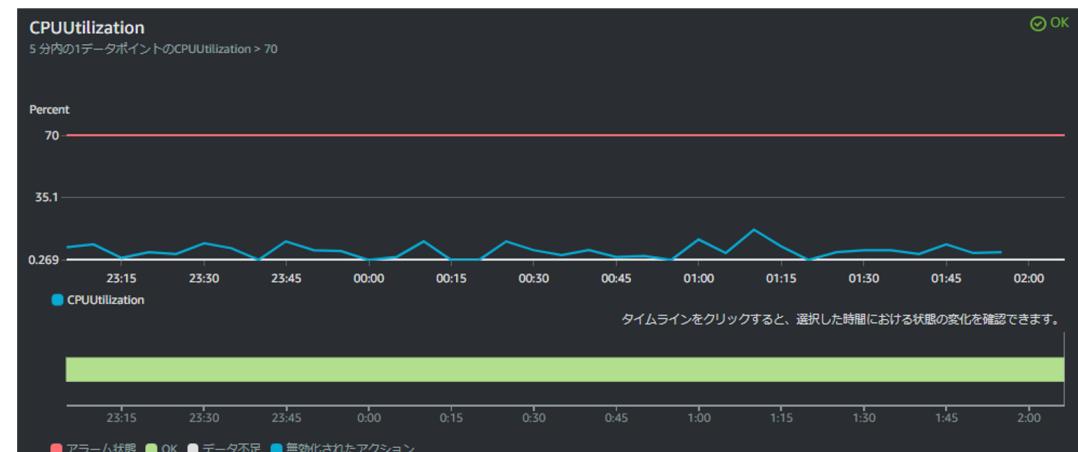
If you are a member of the general public:  
The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.  
If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.  
For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

If you are the website administrator:  
You may now add content to the directory /var/www/html/. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file /etc/httpd/conf.d/welcome.conf.  
You are free to use the image below on web sites powered by the Apache HTTP Server:

# 4. Test Results:

Requirement 3: make EC2 automatically scale out when the CPU load exceeds 70% and scale in when it falls below 20%.

Before applying any load



No instances have experienced CPU loads above 20% or 70%.

インスタンス (15) 情報

検索

接続

インスタンスの状態 ▾

	Name	インスタンス ID	インスタンス...	ステータスチェック
□	CF-public-1a	i-07e8d605733b5cbc9	○ 実行中	○ 2/2 のチェックに合格しました
□	kadai4-asg	i-05927a9f55271df72	○ 実行中	○ 2/2 のチェックに合格しました
□	CF-public-1c	i-0f04b4a8cbfa448b9	○ 実行中	○ 2/2 のチェックに合格しました
□	kadai4-asg	i-07886ca6c3a5429c	○ 実行中	○ 2/2 のチェックに合格しました

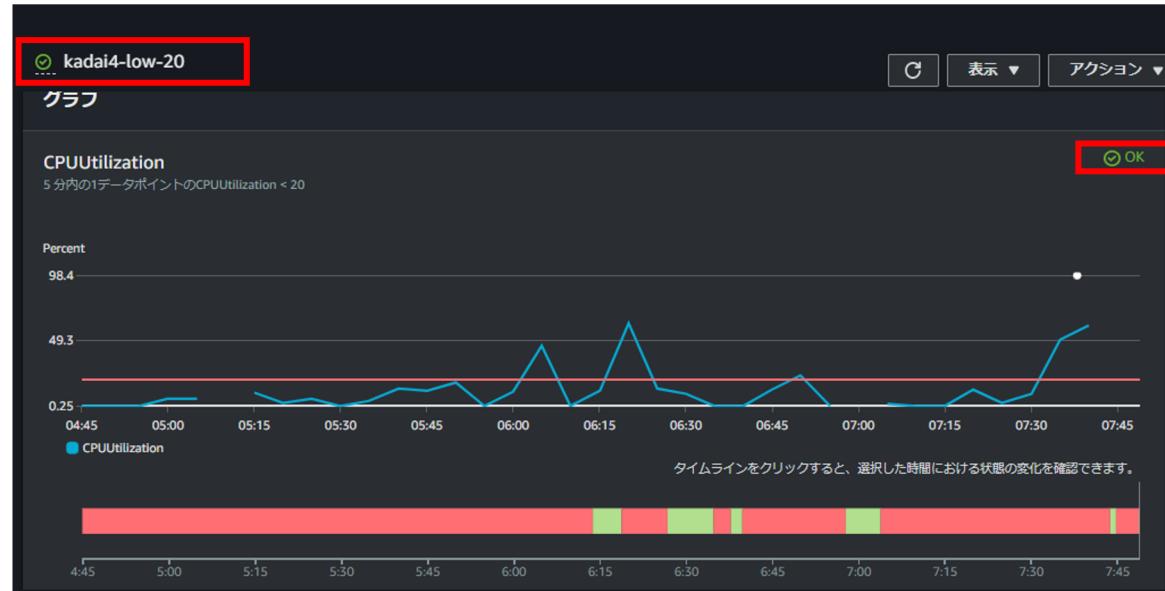
```
[ec2-user@ip-10-0-0-29 ~]$ uptime
07:20:17 up 8 min, 1 user,  load average: 0.47, 0.18, 0.10
[ec2-user@ip-10-0-0-29 ~]$ sudo stress -c 1 --timeout 3000
stress: info: [6874] dispatching hogs: 1 cpu, 0 io, 0 vm, 0 hdd
```

Applied load to these EC2 instances using the stress command.  
sudo amazon-linux-extras install epel -y  
sudo yum install stress -y

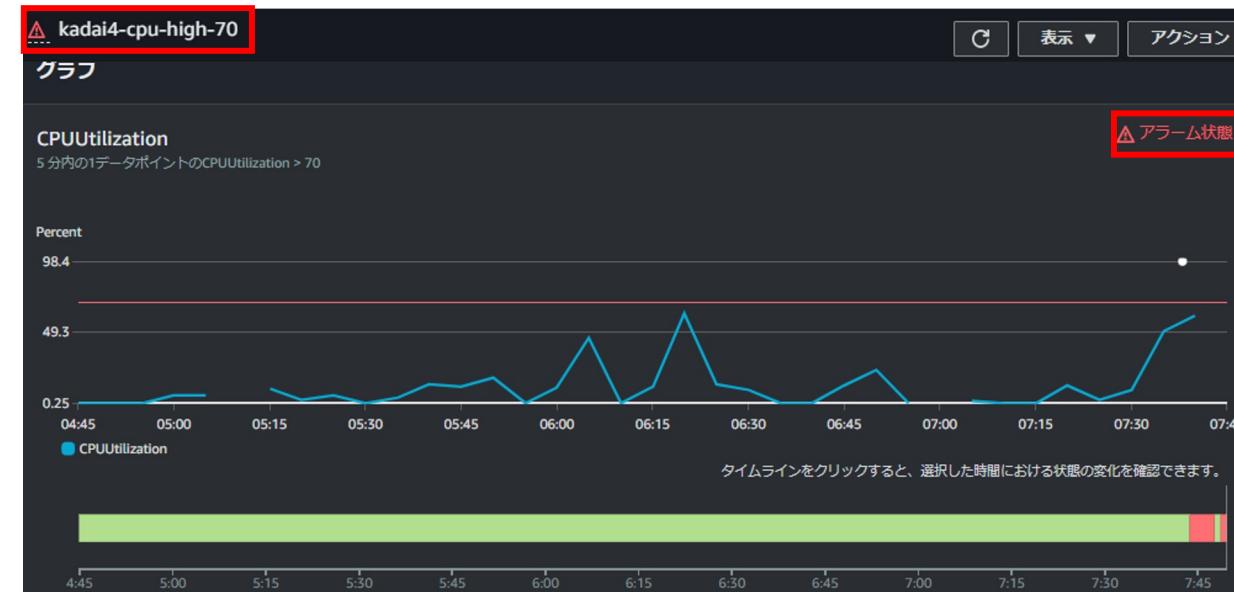
# 4. Test Results:

Requirement 3: make EC2 automatically scale out when the CPU load exceeds 70% and scale in when it falls below 20%.

## Status while applying load



CPU load exceeds 20%, causing the CloudWatch alarm set at CPU 20% to return to the "OK" state.



CPU load exceeds 70%, triggering the CloudWatch alarm set at CPU 70% to enter the "Alarm" state.

# 4. Test Results:

Requirement 3: make EC2 automatically scale out when the CPU load exceeds 70% and scale in when it falls below 20%.

scaling out

The screenshot shows the AWS Auto Scaling Groups interface. A blue callout box points to the instance count column, which is highlighted with a red border. The text inside the callout box states: "As CPU load exceeds 70%, ASG is triggered. Instance count changes from the minimum value of "2" to "3"."

**Auto Scaling グループ (1/1) Info**

名前	起動テンプレート/設定	インスタンス	ステータス	希望するキャパシティ	最小	最大
asg-kadai4	kadaiLaunchTemp   バージョンデフォルト	3	-	2	2	4

**Auto Scaling グループ: asg-kadai4**

Successful	Launching a new EC2 instance: i-072279aa97b60bf86	At 2023-05-06T07:43:45Z a monitor alarm kadai4-cpu-high-70 in state ALARM triggered policy kadai4-cpu-high changing the desired capacity from 2 to 3. At 2023-05-06T07:43:55Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 2 to 3.	2023 May 06, 04:43:57 PM +09:00
			06, 04:43:57 PM +09:00

# 4. Test Results:

Requirement 3: make EC2 automatically scale out when the CPU load exceeds 70% and scale in when it falls below 20%.

scaling out

The screenshot shows the AWS Auto Scaling Groups page for the group 'asg-kadai4'. It displays two log entries for successful launches of new EC2 instances. A callout bubble highlights the second entry, which corresponds to the scaling out event described in the requirement.

Successful	Launching a new EC2 instance: i-0ad49faaec13ea147	At 2023-05-06T07:49:45Z a monitor alarm <u>kadai4-cpu-high-70</u> in state <u>ALARM</u> triggered policy <u>kadai4-cpu-high</u> changing the desired capacity from 3 to 4. At 2023-05-06T07:49:55Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 3 to 4.	2023 May 06, 04:49:57 PM +09:00
Successful	Launching a new EC2 instance: i-072279aa97b60bf86	At 2023-05-06T07:43:45Z a monitor alarm <u>kadai4-cpu-high-70</u> in state <u>ALARM</u> triggered policy <u>kadai4-cpu-high</u> changing the desired capacity from 2 to 3. At 2023-05-06T07:43:55Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 2 to 3.	2023 May 06, 04:43:57 PM +09:00

As continuing to apply load, the number of Instance changes from "3" to "4".

# 4. Test Results:

Requirement 3: make EC2 automatically scale out when the CPU load exceeds 70% and scale in when it falls below 20%.

scale-out

2 instances created from the stack (CF-public-1a, CF-public-1c)  
+ 4 instances scaled out by ASG (kadai4-asg x4) = 6

インスタンス (15) 情報						
		操作		インスタンスの状態 ▾	アクション ▾	インスタンスを起動
<input type="text"/> 属性またはタグ (case-sensitive) で インスタンスを検索						
□	Name ▾	インスタンス ID	インスタンス... ▲	ステータスチェック	アラームの状態	アベイラビリティー
□	CF-public-1a	i-07e8d605733b5cbc9	⟳ 実行中	⟳ 2/2 のチェックに合格しました	アラームなし	+ ap-northeast-1a
□	kadai4-asg	i-072279aa97b60bf86	⟳ 実行中	⟳ 2/2 のチェックに合格しました	アラームなし	+ ap-northeast-1a
□	kadai4-asg	i-05927a9f55271df72	⟳ 実行中	⟳ 2/2 のチェックに合格しました	アラームなし	+ ap-northeast-1a
□	CF-public-1c	i-0f04b4a8cbfa448b9	⟳ 実行中	⟳ 2/2 のチェックに合格しました	アラームなし	+ ap-northeast-1c
□	kadai4-asg	i-0ad49faaec13ea147	⟳ 実行中	⟳ 初期化しています	アラームなし	+ ap-northeast-1c
□	kadai4-asg	i-0452f492b34e97395	⟳ 実行中	⟳ 2/2 のチェックに合格しました	アラームなし	+ ap-northeast-1c

# 4. Test Results:

Requirement 3: make EC2 automatically scale out when the CPU load exceeds 70% and scale in when it falls below 20%.

scale-in

Auto Scaling グループ (1/1) [Info](#)

Q Auto Scaling グループを検索する

<input checked="" type="checkbox"/>	名前	起動テンプレート/設定	インスタンス	ステータス	希望するキャパシティ	最小	最大
<input checked="" type="checkbox"/>	asg-kadai4	kadaiLaunchTemp   バージョンデフォルト	3	-	2	2	4

Auto Scaling グループ: asg-kadai4

Successful	Launching a new EC2 instance: i-0787fe191d14cf51c	kadai4-cpu-low changing the desired capacity from 4 to 3. At 2023-05-06T08:01:49Z a monitor alarm <u>kadai4-low-20</u> in state ALARM triggered policy <u>kadai4-cpu-low changing the desired capacity from 3 to 2</u> . At 2023-05-06T08:02:00Z availability zones ap-northeast-1a ap-northeast-1c had 0 2 instances respectively. An instance was launched to aid in balancing the group's zones.	2023 May 06, 05:02:03 PM +09:00	06, 0 PM +
------------	---	---	---------------------------------	------------

Interrupted the stress command and changed the desired capacity from "3" to "2".

# 4. Test Results:

Requirement 3: make EC2 automatically scale out when the CPU load exceeds 70% and scale in when it falls below 20%.

Scale-in

2 instances created from the stack (CF-public-1a, CF-public-1c)  
+ 2 instances scaled in by ASG (kadai4-asg x2) = 4

インスタンス (16) 情報							
名前							
状態							
<input type="checkbox"/>	Name	▼	インスタンス ID	インスタンス...	▲	ステータスチェック	アラームの状態
<input type="checkbox"/>	CF-public-1a		i-07e8d605733b5cbc9	実行中	<input checked="" type="radio"/> <input checked="" type="radio"/>	2/2 のチェックに合格しました	アラームなし
<input type="checkbox"/>	kadai4-asg		i-0479aa68d8cc05cb8	実行中	<input checked="" type="radio"/> <input checked="" type="radio"/>	2/2 のチェックに合格しました	アラームなし
<input type="checkbox"/>	CF-public-1c		i-0f04b4a8cbfa448b9	実行中	<input checked="" type="radio"/> <input checked="" type="radio"/>	2/2 のチェックに合格しました	アラームなし
<input type="checkbox"/>	kadai4-asg		i-03d24626f4ed67a78	実行中	<input checked="" type="radio"/> <input checked="" type="radio"/>	2/2 のチェックに合格しました	アラームなし
<input type="checkbox"/>	kadai4-asg		i-01264b551bb390e6e	シャットダウン	<input checked="" type="radio"/> <input checked="" type="radio"/>	-	アラームなし
<input type="checkbox"/>	kadai4-asg		i-0787fe191d14cf51c	終了済み	<input checked="" type="radio"/> <input checked="" type="radio"/>	-	アラームなし
<input type="checkbox"/>	kadai4-asg		i-09080a55e38a93f97	終了済み	<input checked="" type="radio"/> <input checked="" type="radio"/>	-	アラームなし

# 4. Test Results:

Requirement 4: Set up alerts to be sent via email to responsible personnel when thresholds are exceeded in monitoring.

Subject: ALARM: "kada14-cpu-high-70" in Asia Pacific (Tokyo) From: AWS Notifications Date: 23-05-06 16:48:47 Size: 5.03K

Subject: ALARM: "kada14-cpu-high-70" in Asia Pacific (Tokyo) From: AWS Notifications Date: 23-05-06 16:44:51 Size: 5.00K

You are receiving this email because your Amazon CloudWatch Alarm "kada14-cpu-high-70" in the Asia Pacific (Tokyo) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [84.40000000000002 (06/05/23 07:43:00)] was greater than the threshold (70.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Saturday 06 May, 2023 07:48:45 UTC".

View this alarm in the AWS Management Console:  
<https://ap-northeast-1.console.aws.amazon.com/cloudwatch/deeplink.js?region=ap-northeast-1#alarmsV2:alarm/kada14-cpu-high-70>

Alarm Details:

Subject: ALARM: "kada14-cpu-high-70" in Asia Pacific (Tokyo) From: "AWS Notifications" <no-reply@sns.amazonaws.com> Select Action

You are receiving this email because your Amazon CloudWatch Alarm "kada14-cpu-high-70" in the Asia Pacific (Tokyo) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [84.40000000000002 (06/05/23 07:43:00)] was greater than the threshold (70.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Saturday 06 May, 2023 07:48:45 UTC".

View this alarm in the AWS Management Console:  
<https://ap-northeast-1.console.aws.amazon.com/cloudwatch/deeplink.js?region=ap-northeast-1#alarmsV2:alarm/kada14-cpu-high-70>

Alarm Details:

- Name: kada14-cpu-high-70
- Description:
- State Change: OK -> ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [84.40000000000002 (06/05/23 07:43:00)] was greater than the threshold (70.0) (minimum 1 datapoint for OK -> ALARM transition).
- Timestamp: Saturday 06 May, 2023 07:48:45 UTC
- AWS Account: 234056973550
- Alarm Arn: arn:aws:cloudwatch:ap-northeast-1:234056973550:alarm:kada14-cpu-high-70

Threshold:

- The alarm is in the ALARM state when the metric is GreaterThanThreshold 70.0 for at least 1 of the last 1 period(s) of 300 seconds.

Monitored Metric:

- MetricNamespace: AWS/EC2
- MetricName: CPUUtilization
- Dimensions: [AutoScalingGroupName = asg-kada14]
- Period: 300 seconds
- Statistic: Average
- Unit: not specified
- TreatMissingData: breaching

State Change Actions:

The responsible person's email address, registered as SNS Topics, received notifications for CloudWatch alarms triggered when CPU load exceeded 70%.

# 4. Test Results:

Requirement 4: Set up alerts to be sent via email to responsible personnel when thresholds are exceeded in monitoring.

The screenshot shows an email inbox with one message highlighted. The message is from "AWS Notifications" and has the subject "ALARM: 'kada4-low-20' in Asia Pacific (Tokyo)". The message body contains a link to view the alarm in the AWS Management Console.

The screenshot shows an email message with the subject "ALARM: 'kada4-low-20' in Asia Pacific (Tokyo)". The message body provides details about the alarm, including its name, state change, reason for state change, timestamp, AWS account, and alarm ARN. It also specifies the monitored metric as "AWS/EC2 CPUUtilization [AutoScalingGroupName = asg-kada4]" with a threshold of "LessThanThreshold 20.0" over "300 seconds".

**Subject:** ALARM: "kada4-low-20" in Asia Pacific (Tokyo)

**From:** "AWS Notifications" <no-reply@sns.amazonaws.com>

You are receiving this email because your Amazon CloudWatch Alarm "kada4-low-20" in the Asia Pacific (Tokyo) region has entered the ALARM state, because "Threshold Crossed: 1 out of the last 1 datapoints [1.2691010660856084 (06/05/23 07:39:00)] was less than the threshold (20.0) (minimum 1 datapoint for OK -> ALARM transition)." at "Saturday 06 May, 2023 07:44:49 UTC".

View this alarm in the AWS Management Console:  
<https://ap-northeast-1.console.aws.amazon.com/cloudwatch/deeplink.js?region=ap-northeast-1#alarmsV2:alarm/kada4-low-20>

**Alarm Details:**

- Name: kada4-low-20
- Description:
- State Change: OK -> ALARM
- Reason for State Change: Threshold Crossed: 1 out of the last 1 datapoints [1.2691010660856084 (06/05/23 07:39:00)] was less than the threshold (20.0) (minimum 1 datapoint for OK -> ALARM transition).
- Timestamp: Saturday 06 May, 2023 07:44:49 UTC
- AWS Account: 234056973550
- Alarm Arn: arn:aws:cloudwatch:ap-northeast-1:234056973550:alarm:kada4-low-20

**Threshold:**

- The alarm is in the ALARM state when the metric is LessThanThreshold 20.0 for at least 1 of the last 1 period(s) of 300 seconds.

**Monitored Metric:**  
MetricNamespace: AWS/EC2  
CPUUtilization  
[AutoScalingGroupName = asg-kada4]

**Data:**

- AWS/EC2
- CPUUtilization
- [AutoScalingGroupName = asg-kada4]
- 300 seconds
- Average
- not specified
- missing

The responsible person's email address, registered as SNS Topics, received notifications for CloudWatch alarms triggered when CPU load fell below 20%.

Fin.