Activities AWS

Activity Routing 3

Activity Peering 4

Activity Auto scaling group and Load Balancer 5

Activity RDS 7

Activity S3 9

Activity Policies 12

Activity SSM 13

Activity Routing

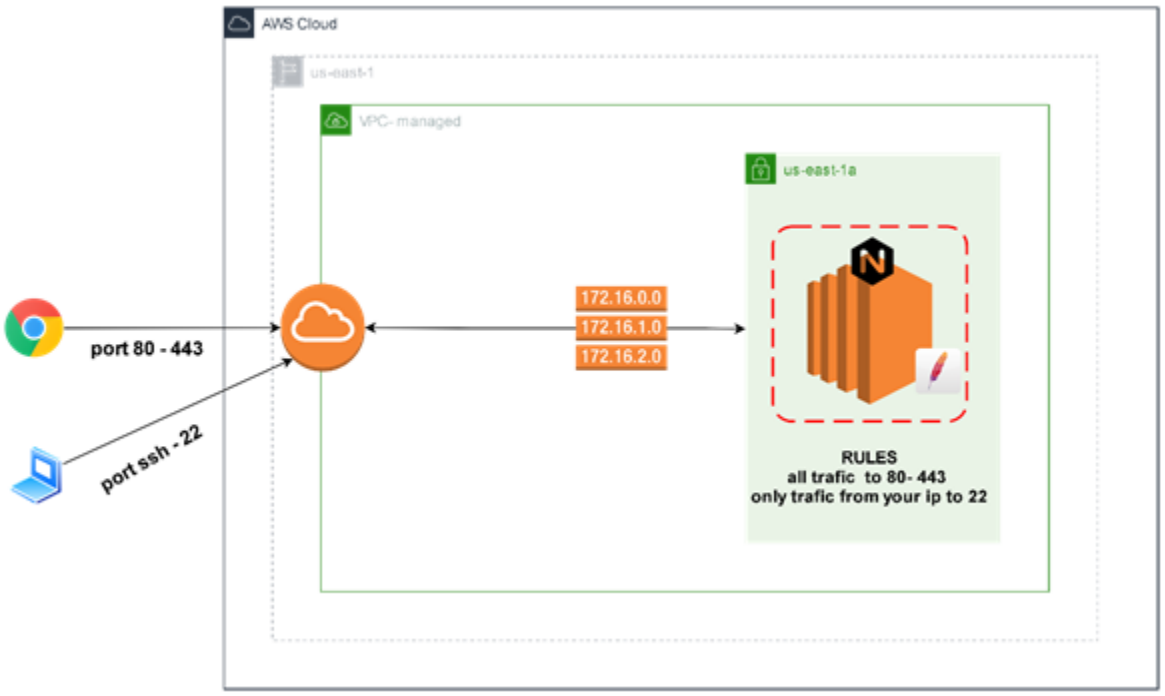
Routing

Customer needs to translocate your infrastructure of servers to AWS, but they have punctual needs to resolve about a server with NginX or Apache which has the website of company:

Need to expose your web service to all Internet only in the HTTP and HTTPS ports.

Only one developer has needed access to the server by way of SSH port and from an IP address specific.

You must solve the customer's needs on AWS, as indicated in the following diagram.



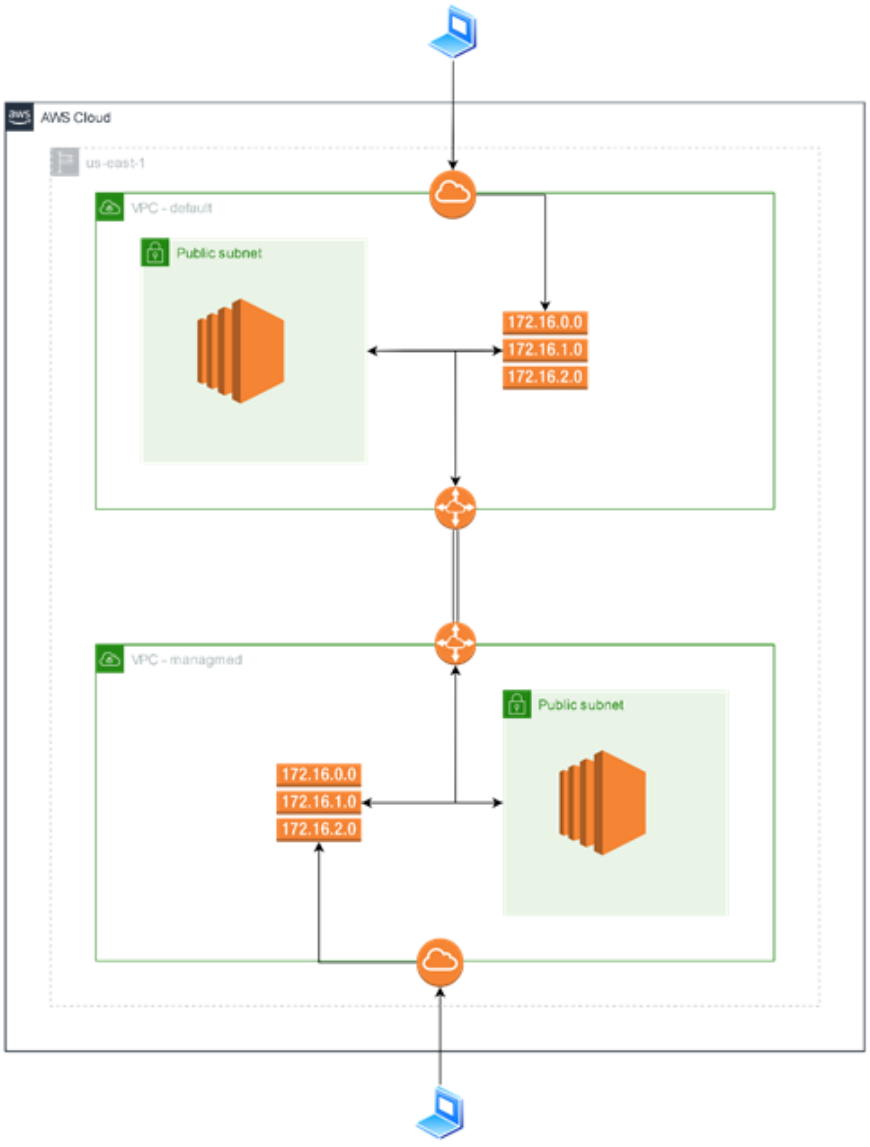
Activity Peering

Peering

The customer has two different applications to serve independent web services, each application is in a separate VPC, but now the customer needs to communicate these applications internally.

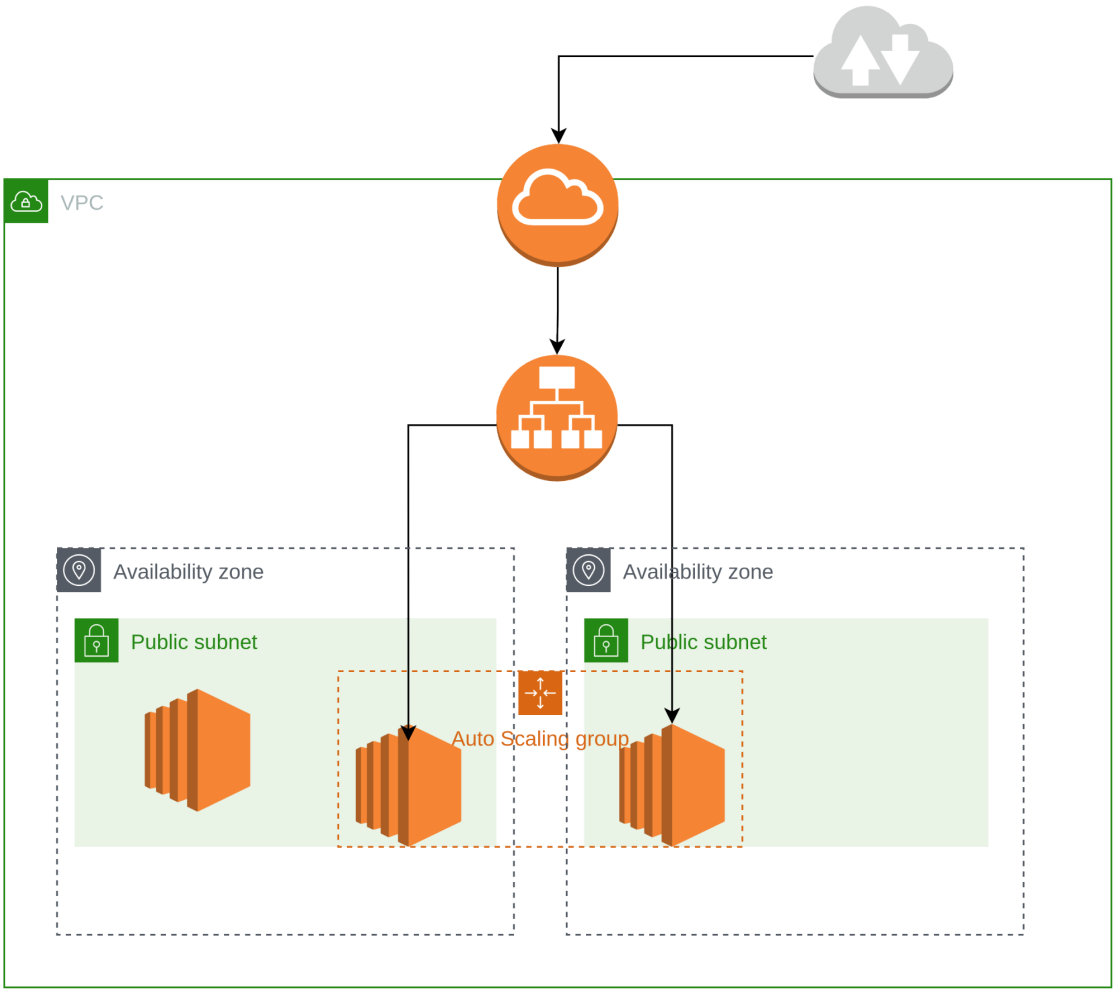
The security of the applications must be guaranteed, none of the applications must go out to the internet to communicate with each other

You must develop this activity in AWS as indicated in the next diagram.



Activity Auto scaling group and Load Balancer

Auto scaling group and Load Blanacer



Step 1 - create VPC with subnets

Create a new VPC with CIDR 10.36.0.0/16 (ac-academy-demo-vpc)

Add 2 public subnets in differents availability zones

- Public-ac-academy-demo-sn1 with 10.36.1.0/24
- public-ac-academy-demo-sn2 with 10.36.2.0/24

Create an internet gateway and attach it to your (vpc ac-academy-demo-igw)

Modify the main route table to:

- add route for the internet gateway
- Associate both the subnets to the main route

Step 2 - create bastion host

Launch 1 EC2 linux instance with public ip in your vpc (vpc ac-academy-demo-ec2)

- Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type - t2.micro
- User data:

```
#!/bin/bash
yum install httpd -y
EC2_AVAIL_ZONE=`curl -s http://169.254.169.254/latest/meta-data/placement/availability-zone`
echo "<h1> Instance Launched Successfully!! <br/> SINCE:  $EC2_AVAIL_ZONE</h1>"  > /var/www/html/index.html
systemctl start httpd
systemctl enable httpd
```

- Create and associate a security group with ports 80 and 22 open from anywhere (0.0.0.0/0) (vpc ac-academy-demo-ec2-sg)

Create an AWS AMI using this EC2 instance (vpc ac-academy-demo-AMI)

Step 3 - Elastic Load Balancer

Create an application load balancer (ac-academy-demo-alb):

- Ensure that the load balancer is internet-facing
- Select availability zones of the 2 plublic subnets
- Create a new security group an open port 80 from anywhere (ac-academy-demo-alb-sg)
- Create a new target group (ac-academy-demo-alb-tg) with
    - Target type as instance
    - Protocol http and port 80
    - Health check protocol as http an path as /index.html
    - No targets to registry
- Copy your application load balancer DNS Name
    - DNS: ac-academy-demo-ELB-1582473484.us-east-1.elb.amazonaws.com
    - Arn: arn:aws:elasticloadbalancing:us-east-1:509130302659:loadbalancer/app/ac-academy-demo-ELB/09bd57b64ac64d98

Step 4 - Launch Configuration and Auto Scaling group

Create security group for launch configuration (ac-academy-demo-lc-sg)

Open Port 80 should be accessible by the security group of your ELB

Open Port 22 should be accessible by the security group of your bastion host

Create a launch configuration (ac-academy-demo-lc) using your AMI and security group

Create a new Auto scaling group

Using your launch configuration

Group size 1

Network - your VPC

Subnets - choose both subnets

Scaling policies

Application Load Balancer request count per target

Target value - 1

Instance need - 10

Activity RDS

Create Networking

Create VPC

The first step in configuring your environment is to create a virtual private cloud (VPC) to hold the resources for both your Amazon Elastic Compute Cloud (Amazon EC2) instance and the Amazon RDS MySQL database.

In this project, the following settings were used:

- Name tag: ac_academy_aws_vpc
- IPv4 CIDR block: 10.37.0.0/16

Create the subnets

Amazon RDS requires two subnets in two different Availability Zones for high availability. This project also uses a public subnet for the EC2 instance itself.

In this project, the following settings were used:

- Private subnet 1
    - Name: ac_academy_aws_priv_sb01
    - CIDR: 10.37.101.0/24
- Private subnet 2
    - Name: ac_academy_aws_priv_sb02
    - CIDR: 10.37.102.0/24
- Public subnet
    - Name: ac_academy_aws_pub_sb
    - CIDR: 10.37.201.0/24

Create an Amazon EC2 instance

You can use any Amazon Machine Image (AMI) for this.

In this project, the following settings were used:

- The instance was placed in the ac_academy_aws_vpc.
- The instance was placed in the ac_academy_aws_pub_sb.

Create security groups

Security groups are what controls who have access to the Amazon EC2 instance and Amazon RDS database.

In this project, the following settings were used:

- Group 1
    - Security group name: sg-ec2-to-rds
    - Description: SSH access to EC2 for RDSProject
    - Add a rule using the SSH port 22 and select My IP.
- Group 2
    - Security group name: sg-rds-db-access
    - Description: Access from EC2
    - Add a rule using the MySQL/Aurora port 3306 and select the EC2 for RDS security group.

Create RDS

Create subnet groups

To create a new subnet group, give it a name and a description, and choose an existing VPC. You will then be able to add subnets related to that VPC.

- Name: ac_academy_aws_rds
- Description: ac_academy_aws_rds
- VPC: ac_academy_aws_vpc
- Subnets
    - Availability Zones: the Availability Zones that include the subnets you want to add.
    - Subnets:
        - Ac_academy_aws_pub_sb
        - ac_academy_aws_priv_sb01

Create database

- Engine options
    - Engine type: MariaDB
    - Version: latest
- Templates : Free tier
- Settings
    - DB instance identifier: db-demo
    - Credentials Settings
        - Master username: root
        - Master password:
- DB instance class
    - DB instance class: Burstable classes (includes t classes)
    - Db.t3.micro
- Storage
    - Storage type: General purpose
    - Allocated storage: 20
- Storage autoscaling : NO
- Availability & durability: Do not create a standby instance

Connect

From the instance created run the following command, if you do not have mysql client installed you must install it

```
mysql -h Endpoint-URL --user admin --password
```

if a test database does not exist you must create one

Create example table

```
CREATE TABLE employees (
    emp_no      INT             NOT NULL,  -- UNSIGNED AUTO_INCREMENT??
    birth_date  DATE            NOT NULL,
    first_name  VARCHAR(14)     NOT NULL,
    last_name   VARCHAR(16)     NOT NULL,
    gender      ENUM ('M','F')  NOT NULL,  -- Enumeration of either 'M' or 'F'
    hire_date   DATE            NOT NULL,
    PRIMARY KEY (emp_no)                   -- Index built automatically on primary-key column
                                -- INDEX (first_name)
                                -- INDEX (last_name)
);
```

Download an example database from: https://dev.mysql.com/doc/index-other.html and load in the RDS instance
Direct link : https://downloads.mysql.com/docs/world_x-db.zip

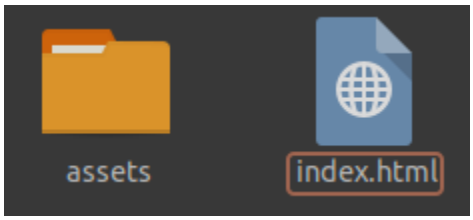
Activity S3

Static site
GitHub Repo: https://github.com/BuckyMaler/global

Create a bucket
Create a s3 bucket

- Name: ac-acadamy-aws-s3-USER_EMAIL
- Region: us-east-1
- ACLs disabled

Load static site
After download the static site from repo



To upload the static site to your bucket, do one of the following:

- Drag and drop the files into the console bucket listing.
- Choose Upload, and follow the prompts to choose and upload files.

Enable static website hosting
In the Buckets list, choose the name of the bucket that you want to enable static website hosting for.

- Choose Properties.
- Under Static website hosting, choose Edit.
- Choose Use this bucket to host a website.
- Under Static website hosting, choose Enable.

Note: Amazon S3 enables static website hosting for your bucket. At the bottom of the page, under Static website hosting, you see the website endpoint for your bucket.

Edit Block Public Access settings
By default, Amazon S3 blocks public access to your account and buckets. If you want to use a bucket to host a static website, you can use these steps to edit your block public access settings.

- Choose Permissions.
- Under Block public access (bucket settings), choose Edit.
- Clear Block all public access, and choose Save changes.

Add a bucket policy that makes your bucket content publicly available

After you edit S3 Block Public Access settings, you can add a bucket policy to grant public read access to your bucket. When you grant public read access, anyone on the internet can access your bucket.

- Choose Permissions.
- Under Bucket Policy, choose Edit.
- To grant public read access for your website, copy the following bucket policy, and paste it in the Bucket policy editor.

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "PublicReadGetObject",
         "Effect": "Allow",
         "Principal": "*",
         "Action": [
            "s3:GetObject"
         ],
         "Resource": [
            "arn:aws:s3:::<your bucket name here>/*"
         ]
      }
   ]
}
```

- Update the Resource to your bucket name.
- Choose Save changes.

Test your website endpoint

After you configure static website hosting for your bucket, you can test your website endpoint. Under Buckets, choose the name of your bucket.

- Choose Properties.
- At the bottom of the page, under Static website hosting, choose your Bucket website endpoint.
- Your index document opens in a separate browser window.

Note:

Amazon S3 does not support HTTPS access to the website. If you want to use HTTPS, you can use Amazon CloudFront to serve a static website hosted on Amazon S3.

With Route 53 you can configure a static website using a custom domain registered.

Clean up

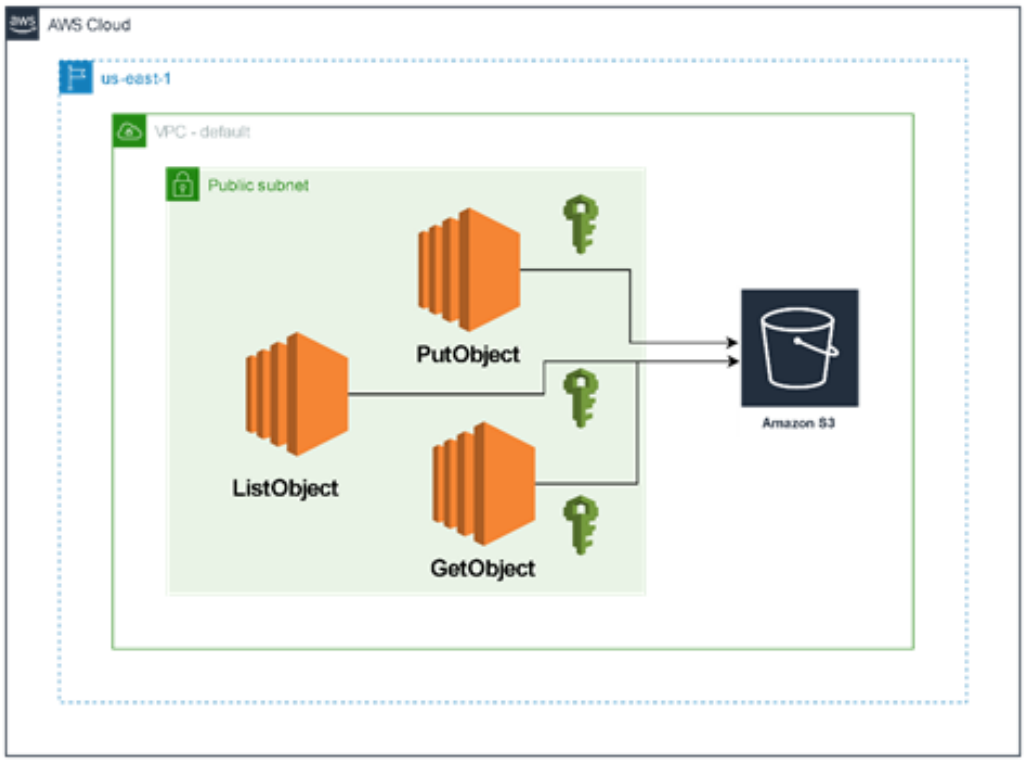
If you created your static website only as a learning exercise, delete the AWS resources that you allocated so that you no longer accrue charges. After you delete your AWS resources, your website is no longer available. For more information, see Deleting a bucket.

Activity Policies

IAM Policy Simulator: https://policysim.aws.amazon.com/home/index.jsp?#

Customer has 3 applications, these applications have connection with an object storage (storage of static objects), an application reads objects from storage, another application creates and puts objects in the storage and the last application gets objects from the storage.

You must resolve the needs of the client in AWS, it's very important ensured the principle of the least privilege.



Note: Using high-level (s3) commands with the AWS CLI
https://docs.aws.amazon.com/cli/latest/userguide/cli-services-s3-commands.html

Activity SSM

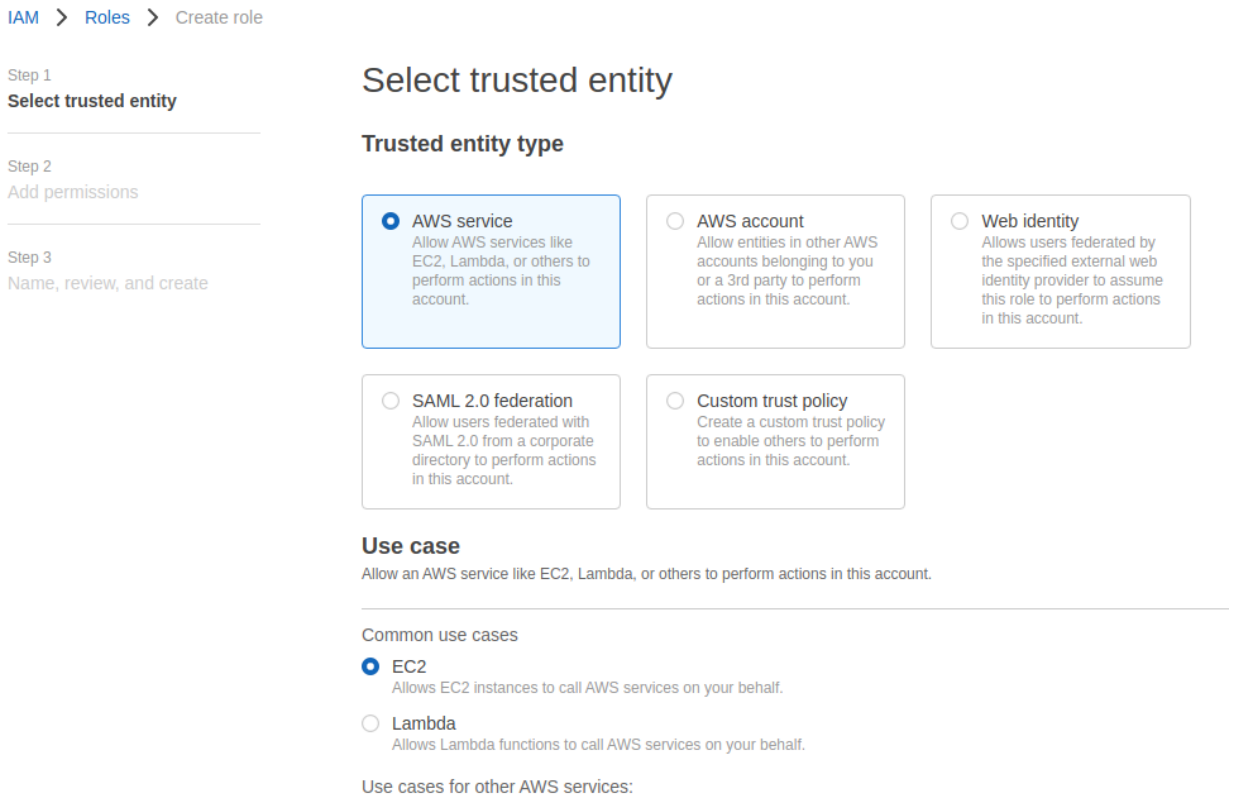
If not exits, create a policy for Systems Manager

police name: SSMTagEditorAndResourceGroupAccess or ResourceGroupsandTagEditorFullAccess

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "tag:getResources",
            "tag:getTagKeys",
            "tag:getTagValues",
            "tag:TagResources",
            "tag:UntagResources",
            "resource-groups:*",
            "cloudformation:DescribeStacks",
            "cloudformation:ListStackResources"
         ],
         "Resource": "*"
      }
   ]
}
```

Create a group “SSMUserGroup”, Add the users and the policy “ResourceGroupsandTagEditorFullAccess” or “SSMTagEditorAndResourceGroupAccess” “AmazonSSMFullAccess”

Create e role



Add the policy “AmazonSSMManagedInstanceCore”, “AmazonSSMManagedInstanceCore”, ”CloudWatchAgentServerPolicy”

For Name, enter a name to identify this role “SSMInstanceProfile”

Create as new policy to instance profile

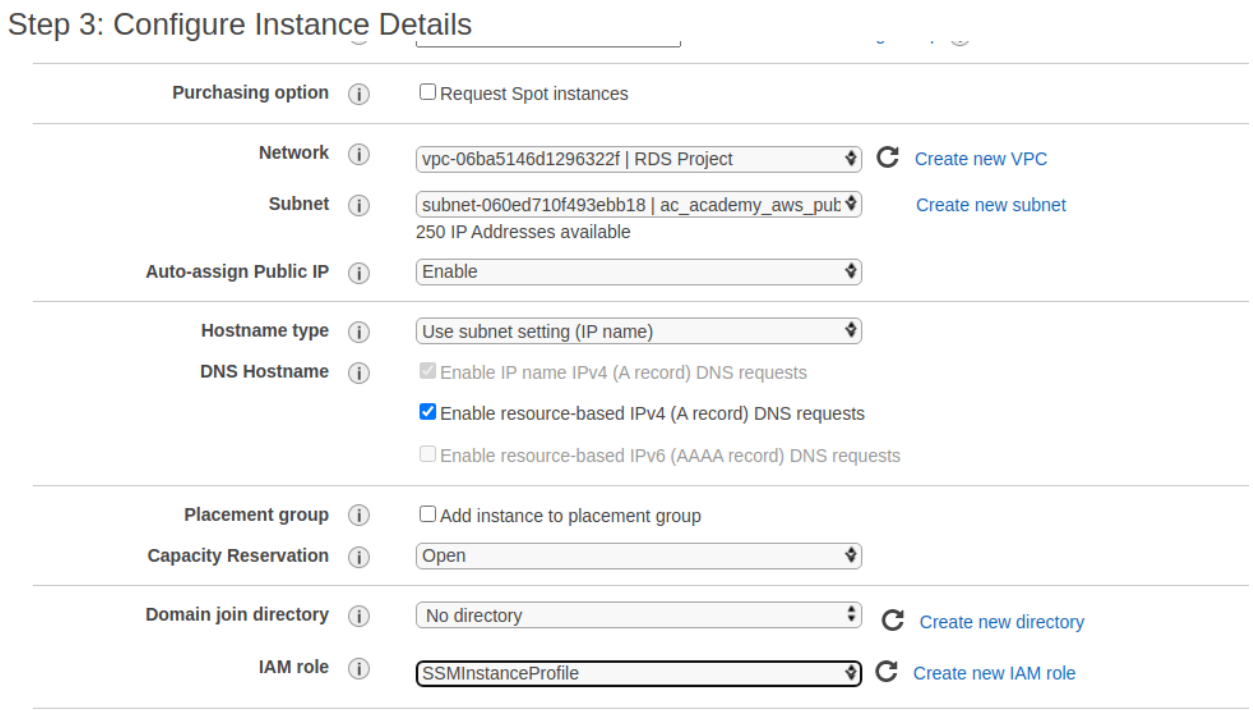
```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": "s3:GetObject",
         "Resource": [
            "arn:aws:s3:::aws-ssm-us-east-1/*",
            "arn:aws:s3:::aws-windows-downloads-region/*",
            "arn:aws:s3:::amazon-ssm-region/*",
            "arn:aws:s3:::amazon-ssm-packages-region/*",
            "arn:aws:s3:::region-birdwatcher-prod/*",
            "arn:aws:s3:::aws-ssm-distributor-file-region/*",
            "arn:aws:s3:::aws-ssm-document-attachments-region/*",
```

```
            "arn:aws:s3:::patch-baseline-snapshot-region/*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "s3:GetObject",
            "s3:PutObject",
            "s3:PutObjectAcl",
            "s3:GetEncryptionConfiguration"
        ],
        "Resource": [
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",
            "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        ]
    }
  ]
}
```

For Name, enter a name to identify this policy “SSMInstanceProfileS3Policy”

Add this policy ““SSMInstanceProfileS3Policy”” to role “SSMInstanceProfile”

Create an instance a assig ht role



SSM Agent is installed by default on the following AMIs:

- Amazon Linux
- Amazon Linux 2
- Amazon Linux 2 ECS-Optimized Base AMIs
- macOS 10.14.x (Mojave), 10.15.x (Catalina), and 11.x (Big Sur)
- SUSE Linux Enterprise Server (SLES) 12 and 15
- Ubuntu Server 16.04, 18.04, and 20.04
- Windows Server 2008-2012 R2 AMIs published in November 2016 or later
- Windows Server 2016, 2019, and 2022

Activity KMS

Generate demo file

```
aws kms generate-data-key --key-id alias/ac-demo --key-spec AES_256 --region us-east-1 > key.txt && cat key.txt
```

Example output

```
{
    "CiphertextBlob":
"AQIDAHg04up1TO5H25LKm3gqsNJPum9rEnhZIPElzJXNlvvbWwE6F6J8EiOIXAFe2HtkOeOeAAAAfjB8BgkqhkiG9w0BBwagbzBtAgEAMGgGCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQMdihZhS397rFnbT5rAgEQgDur+ds/btJ1vCrbrGyoOugpWEBrPZdlGg22QD/t/QkXcPzFhYP2GCtlhIb/MOyGJtogMDQabispm+FrPw==",
    "Plaintext": "gRs/T8nYbWoLg8klEeWQP6QMZN5j5FtgNmTHJZmg0Yg=",
    "KeyId":
"arn:aws:kms:us-east-1:509130302659:key/b3486106-2d69-4517-8872-1f48735b890e"
}
```

Decode
Use “Plaintext” data to generate datakey

```
echo "gRs/T8nYbWoLg8klEeWQP6QMZN5j5FtgNmTHJZmg0Yg=" | base64  --decode > datakey.txt  && cat datakey.txt
```

Use “CiphertextBlob” data to generate encryped datakey

```
echo
"AQIDAHg04up1TO5H25LKm3gqsNJPum9rEnhZIPElzJXNlvvbWwE6F6J8EiOIXAFe2HtkOeOeAAAAfjB8BgkqhkiG9w0BBwagbzBtAgEAMGgGCSqGSIb3DQEHATAeBglghkgBZQMEAS4wEQQMdihZhS397rFnbT5rAgEQgDur+ds/btJ1vCrbrGyoOugpWEBrPZdlGg22QD/t/QkXcPzFhYP2GCtlhIb/MOyGJtogMDQabispm+FrPw==" | base64  --decode  > encrypted-datakey.txt && cat encrypted-datakey.txt
```

Encrypt file

```
openssl enc -in ./demofile.txt  -out ./demofile-encrypted.txt -e -aes256 -k  fileb://./datakey.txt && cat demofile-encrypted.txt
```

Or using -pbkdf2

```
openssl enc -in ./demofile.txt  -out ./demofile-encrypted3.txt -e -aes256 -pbkdf2 -k fileb://./datakey.txt  && cat demofile-encrypted3.txt
```

Decrypt file
Get datakey with encrypted data key

```
aws kms decrypt --ciphertext-blob fileb://./encrypted-datakey.txt --region us-east-1
```

Example output

```
{
    "KeyId":
"arn:aws:kms:us-east-1:509130302659:key/b3486106-2d69-4517-8872-1f48735b890e",
    "Plaintext": "gRs/T8nYbWoLg8klEeWQP6QMZN5j5FtgNmTHJZmg0Yg=",
    "EncryptionAlgorithm": "SYMMETRIC_DEFAULT"
}
```

Decode
Use “Plaintext” data to generate datakey

```
echo "gRs/T8nYbWoLg8klEeWQP6QMZN5j5FtgNmTHJZmg0Yg=" | base64  --decode > datakey.txt  && cat datakey.txt
```

Decrypt

```
openssl enc -in ./demofile-encrypted.txt  -out ./demofile-decrypted.txt -d -aes256  -k fileb://./datakey.txt  && cat demofile-decrypted.txt

Or using -pbkdf2

openssl enc -in ./demofile-encrypted3.txt  -out ./demofile-decrypted3.txt -d -aes256 -pbkdf2 -k fileb://./datakey.txt  && cat demofile-decrypted3.txt
```