ProTechGurus

AWS Certified Solution Architect –

Associate

“Lab Manual Guide”

Step By Step Lab Exercises and Best Practices

Contents

About this Book About the Author Disclaimer Copyright

1. ​Sign Up for AWS Free Tier Account

2. ​Getting Familiarized with AWS Console

3. ​Creating an AWS IAM User Enabling MFA for IAM User Enabling Virtual MFA for IAM User

4. ​Managing Virtual Private Cloud (VPC)

Creating VPC in AWS Cloud

Creating and Adding Private Subnet in the Existing VPC Deleting VPC

5. ​Creating and Configuring Internet Gateways

6. ​Creating and Configuring NAT Gateways

7. ​Configuring Routing Tables

Creating Custom Routing Table

8. ​VPC Peering Between Two VPCs Creating VPCs and Subnets Creating Subnets

Creating VPC Peering Connection

Updating Routing Tables and Routes

9. ​Working with Amazon Elastic Cloud Compute (EC2) Creating Amazon EC2 Linux Instance

Selecting Instance Type

Specify the Instance Configuration Options

Adding Storage Volumes Selecting Security Group Creating/Selecting Key Pair

10. ​Creating and Configuring Security Groups

Creating a Security Group Security Group Rules Modifying Security Group

11. ​Managing Elastic IP Addresses

Elastic IP Pricing

Assigning Elastic IPs to EC2 Instances

12. ​Connecting EC2 Linux Instance Using PuTTy, Gitbash, and Console

Connecting EC2 Instance Using Web Browser Connecting EC2 Linux Instance Using PuTTY Default EC2 Instance Users

Connecting EC2 Instance Using GitBash

13. ​Connecting Private Instance using SSH Agent Forwarding

Allow SSH connection from one security group to another security group

Access SSH using SSH Agent Forwarding

14. ​Accessing EC2 Linux Instance Using RDP with GUI Interface Installing MATE Desktop Environment in EC2 Ubuntu Instance Allow RDP Rule in the Security Group

Connecting EC2 Ubuntu using MSTSC (RDP)

15. ​Recovering and connecting EC2 instance if the SSH key is lost

Stopping Original EC2 Linux Instance

Launching New Temporary Instance

Detaching Root Volume from Original Instance Attaching Root Volume to Temporary Instance Mounting Attached Volume

Modifying the authorized\_keys File and Updating the New Private Key

Detaching Volume from Temporary Instance and Re-attaching With the Original Instance

Connect EC2 Linux Instance (Original Instance) With New Private Key

16. ​Changing Instance type, security groups, volumes & other settings

Changing instance type

Accidental Termination Protection

Specifying Auto Start Script with EC2 Instance

Changing/Adding Security Groups to EC2 Instance

17. ​Start, Stop, Reboot, and Terminate EC2 Instance

18. ​Creating and configuring Elastic Load Balancer

Load Balancer Modes

AWS Load Balancer Types

Create an Application Load Balancer

Domain Registration for ELB

19. ​Scheduling Auto Snapshot of Volumes

Creating Rule

Defining Event Source

Cron Expressions Syntax and Examples

Specifying Target

Configure Rule and Details

20. ​Creating Amazon Machine Image (AMI) and recovering EC2 instance using AMI

Creating some users and files in the source Instance

Creating Image of an EC2 instance

Specifying AMI settings

Launching an EC2 Instance using AMI Configuring Instance Details Configuring Security Group

Reviewing and launching Instance

Login in to the New Instance

21. ​Configuring CloudWatch Monitoring Amazon CloudWatch Pricing Selecting an Instance

Creating a CloudWatch Alarm

22. ​Configuring Amazon Simple Notification Service (SNS)

Amazon SNS Free Tier vs Paid Pricing

Creating a Topic

Subscribing to the topic

23. ​Configuring Centralized Log Management Using CloudWatch Log

Group

Creating a CloudWatch Log Group

Granting Permission to EC2 Instance to Use CloudWatch Logs

Attaching policy to IAM Role Assigning Role to an EC2 Instance Installing AWS CloudWatch Log Agent Verifying Logs in CloudWatch Log Group

25. ​Schedule Auto, Start, Stop, and Reboot EC2 Instances

Create an Auto Start EC2 Instance Lambda Function Creating an Auto Start-Stop Event Schedule Rule Test and Validate Auto Start-Stop EC2 Schedule

26. ​Creating and Recovering EC2 Instance Using Snapshots

Creating Snapshot of Attached Volume Creating Volume of EC2 Snapshot Attaching Volume to an Instance Mounting the Attached Volume

27. ​Working with IAM User Properties

Permissions

Groups

Security Credentials

Access Advisor

28. ​Creating and Using an IAM Role

Attaching IAM role to an EC instance

29. ​Configuring Password Policies for IAM Users

Changing IAM Password Policy Settings

30. ​Installing and configuring AWS CLI Download AWS CLI for Windows System

Installing AWS CLI on Windows System Installing AWS CLI on Linux system Setup Credentials for AWS CLI

31. ​Configuring OpenVPN Server to Securely Access Instances

Launching Pre-configured AMI from AWS Market Place

Assigning Elastic IP

Configuring OpenVPN Initial Settings

Activating License.

Creating OpenVPN user accounts. Configuring VPN settings.

Enabling Google Authenticator for VPN Clients

32. ​Connecting OpenVPN Server

Two-Factor Authentication

Setting up Two Factor Configuration Connecting Using OpenVPN Client Tool For Linux Platform

For Windows OS

33. ​Configuring Linux Bastion Server for Securely Access SSH of Private

Instances

Launching a New Bastion Server Configuring Security Groups Connecting SSH using Bastion Server

34. ​Working with S3 Buckets

Recommended links

Creating an S3 Bucket Using the AWS Console

Creating Folders under S3 Bucket Upload Content/Files/Data in S3 Bucket Deleting S3 Bucket

35. ​Configuring Permissions and Policy for S3 Buckets

Customizing S3 Bucket Policies

Access Control List Tab

Bucket Policy Tab

36. ​Configuring S3 Bucket Policies for Specific IAM Users Creating Custom S3 Policy for a Specific IAM User Attaching Policy to IAM Role, Group, or User Verifying Custom S3 Bucket Policies

Verifying S3 Bucket Policy Using AWS CLI Tool

37. ​Configuring S3 Bucket Versioning and Logging.

Configuring S3 Bucket Versioning

Configuring Logging for S3 Buckets

38. ​Configuring S3 Bucket Alerting and Notifications

39. ​Configuring S3 Bucket Lifecycle Rule

40. ​ImplementingCross-RegionS3Replication Configuring cross-replication of S3 bucket Creating Replication Rule

Verifying Cross-region S3 Bucket Replication

Troubleshooting S3 Cross-region Replication

41. ​Enabling and configuring AWS CloudTrail

Enabling CloudTrail

Creating a CloudTrail

42. ​Working with Auto Scaling Group Creating a Launch Configuration Group Creating Auto Scaling Group

Verifying Auto Scaling Group Configuration

Modifying Auto Scaling Group Settings

43. ​Configuring Amazon Route 53

Registering a domain name

44. ​Working with Amazon WorkDocs Creating an AWS WorkDocs Site Deleting WorkDocs Site

45. ​Working with AWS Trusted advisor

Thank You

About this Book

This book has been written for the candidates who want to learn and use the AWS cloud platform, candidates who have just started his career with AWS Cloud services, who is already working with AWS cloud services. This is also dedicated for those who are preparing themselves for one of the most popular cloud certifications AWS Certified Solution Architect – Associate Exam.

This is mainly focused on the hands-on lab exercises and real-world best practices rather than deep theoretical and conceptual lectures. There are hundreds of theoretical documentation are available on the AWS official documentation site and few other sites on the Internet community. But, there are very few articles on the step by step how to guide to implement, use, and configure AWS Cloud services. This guide will help you to become handy and expert on most of the AWS Cloud services that come under the syllabus of AWS Certified Solution Architect. However, this guide also contains various real world, enterprise-level, best practices to implement and use for the production services.

These step by step lab exercises will help you to design a highly-secure, scalable, and well-architecture enterprise-level Cloud solutions and designs.

We assume that you know the basic terminology of knowledge about the various AWS Cloud services. This book is not focused on the deep theory and concepts about the AWS Cloud services. However, sufficient official documentation links are given before almost each of the lab exercise. So you can use those links to get familiar with the respective AWS Cloud service that is described and implemented in that particular lab exercise.

About the Author

This book is written by an Author who has over 9+ years of experience in various IT domains such as Microsoft, Red Hat, Cisco, AWS, OpenStack, and CompTIA. He has worked with many reputed organizations as various profiles such as Content Designer, Subject Matter Expert, Technical Expert, and Solution Architect. He currently holds 15 + IT Global certifications. Some of the major Global certifications hold by him includes MCP, MCTP, MCITP, MCSA, MS Hyper-V, CCNA, RHCSA, OpenStack, and AWS Certified Solution Architect – Associate.

Disclaimer

While writing the content of this book, author (publisher) has tried his best to keep content as up-to-date and accurate as possible. However, being a

human, there might be some typo or grammatical errors. It might also be possible that due to the rapid changes of IT technologies and services, some of the features, limitations, and terms might get change or updated by the respective IT vendors. Author or publisher does not provide any guarantee for such changes. The lab exercises mentioned in this book should be performed in the testing environment, not on the productions servers or environment. Author or publisher will not be liable for any loss of data, services interruptions caused by candidates either intentionally or unintentionally.

Copyright

All the content of this entire book is completely copyrighted by the author. No content of this book can be republished, copied, and reproduced in any format including paper, e-paper, electronic media, PDF, word, and/or any in other digital media without the prior written permissions of the author.

*Note: Since few of the information (such as account number, email ID, public IP address, instance ID etc.) in the mentioned screenshots are private and confidential, so those might be greyed out or marked with blur. However and obviously, this will not impact your learning experience at anyhow.*

1. Sign Up for AWS Free Tier Account

The best thing about getting started with any cloud platform is that almost every cloud provider provides the free tier services with limited features and limit. AWS provides you free tire account for 12 months with limited features and selected services. However, for the learning purpose, this free tier

account is more than enough.

If you are interested to know what you will get with free tier account, please visit the following link and spent some time to know various free services and their limitations.

[AWS Free Tier Limitations](https://aws.amazon.com/free/)

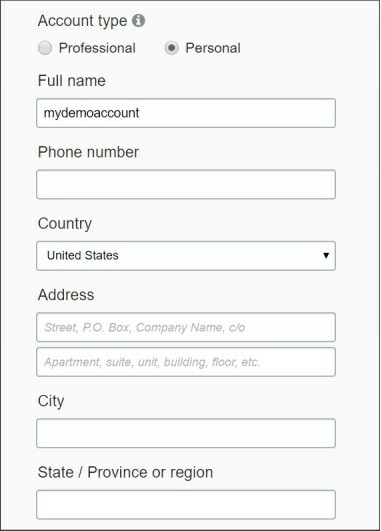
1. To sign up for AWS free tier account, click the below link and complete the Sign Up process.

[AWS Free Tier Sign UP Link](https://portal.aws.amazon.com/billing/signup#/account)

2. In the Sign Up screen, you need to fill the basic information as shown in the following figure.



3. On the next screen, select the **Account Type** and fill the required information.



4. On the next screen, you need to provide the credit card details and billing address information.

5. On the next screen, you may need to validate your contact number and you are almost done to get started with free tier account.

*Note: Here, you have to wait for few hours as your account and associated services will be created in the background. You will get a welcome and get started mail from the AWS support team once your account is ready.*

6. By default, the email address you used to Sign Up is also used to login to AWS console as the root account. Please visit the following link to sign in to AWS console.

[AWS Management Console Sign In Link](https://console.aws.amazon.com/console/home)

2. Getting Familiarized with AWS Console

Once signed in to your AWS console, you can start using AWS cloud services. However, AWS has hundreds of services and features. So it might be difficult for a new user to find and use the appropriate features.

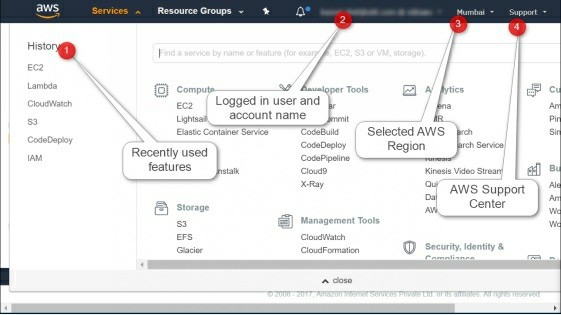
Here are the most common and basic AWS cloud services that you may like to get familiarized as a beginner.

Elastic Cloud Compute (EC2) Simple Storage Service (S3) Relational Database (RDS) Route 53

Virtual Private Cloud (VPC)

CloudWatch, and many more depending on your specific need and requirement.

The following figure shows the common AWS Console options.



We recommend spending some time to get familiarized with navigating AWS management console. You can always use the **Search** box to search any specific service or feature you may wish to explore.

3. Creating an AWS IAM User

Identity and Access Management (IAM) is one of the most important factors to manage and set access control for the AWS cloud resources. By default, when you sign up for AWS account, a ROOT account has already created and has super admin control. Although, root account should not be used for every task and or by every person.

For the best practices and better control, create separate IAM users for a separate purpose and set appropriate permissions and policies as per the requirements.

For the more details on the AWS IAM management, please visit the following link.

[Getting started with AWS Identity and Access Management (IAM)](http://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html) To create an IAM user, you need to perform the following steps:

1. Open the **IAM console** and click **Users** in the left pane.

2. In the right pane, click **Add user** to add a new user.

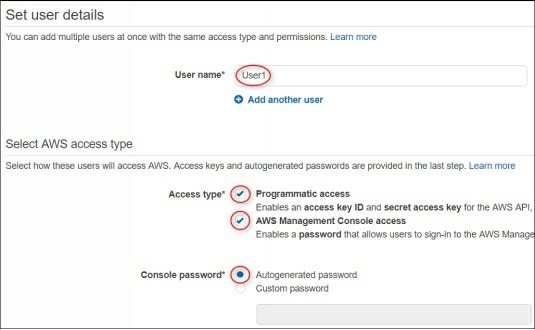
3. On the **Set user details** page, provide the following details:

**Username**: Name of the IAM user

**Access Type**: whether you want to give Console access or CLI

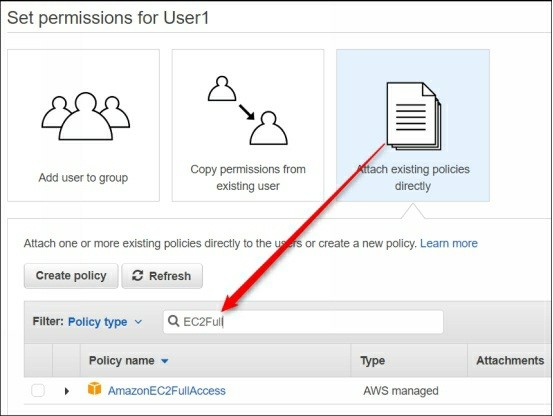
access or both access.

**Console password**: Auto-generated password or custom password.



4. Click **Next: Permissions** to proceed to the permissions page.

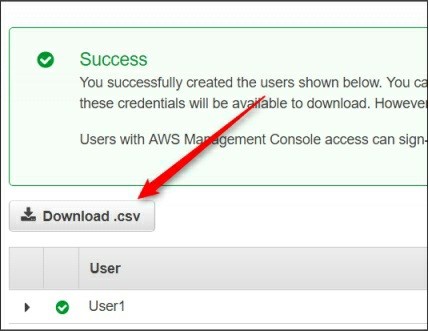
5. On the **Set Permissions for User1** page, you can either add a user to a group, copy existing permissions from a user, or attach the existing policies directly to this IAM user. For example, select attach the existing policies directly, select **AmazonEC2FullAccess** as the permissions and proceed to the next page.



6. On the **Review** page, review the options you have selected and click

**Create User** to finish the wizard.

7. Once the user is created, click **Download .csv** file to download it and keep it in a secret place. This file contains all the required credentials details such as Username, password, access key, secret key and console login link.



8. Click the AWS console link and use the username and password to login to AWS console. For the AWS CLI (covered in the later sections), use the access key and secret key stored in the .csv file.

**Enabling MFA for IAM User**

Security for AWS console is the prime concern for the Cloud administrators. Every user, including AWS Root Account, should be enabled with Multi-Factor Authentication (MFA) for secure AWS console login. The MFA feature adds an additional layer of security while login to

AWS console. You can enable MFA for AWS IAM or root user using either a hardware-based MFA device or a virtual MFA application .

Please visit the following link to know more about the AWS Multi-Factor

Authentication.

[Enabling MFA for AWS IAM Users](http://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_mfa_enable_virtual.html)

Here, we are going to explain how to enable virtual MFA for the IAM

users. There are various virtual

MFA applications are available to use depending on the mobile platform you are using.

The following virtual MFA applications are available for mobile devices:

Android: [Google Authenticator;](https://play.google.com/store/apps/details?id=com.google.android.apps.authenticator2&hl=en) Authy 2-Factor Authentication iPhone: [Google Authenticator;](https://itunes.apple.com/in/app/google-authenticator/id388497605?mt=8) Authy 2-Factor Authentication Windows Phone: [Authenticator+](https://www.microsoft.com/en-in/store/p/authenticator/9nblggh08h54)

Blackberry: [Google Authenticator](https://appworld.blackberry.com/webstore/content/29401059/?countrycode=IN&lang=en)

You can also use the hardware-based MFA device, however, you may need to pay something to purchase it. This article is focused on virtual

MFA application as it is the cheapest option.

Before proceeding to the next, process, make sure that you have installed the appropriate virtual MFA application for your mobile device.

**Enabling Virtual MFA for IAM User**

To enable MFA for AWS IAM user, you need to perform the following steps:

1. Login to the **AWS Management Console** with the admin privileges.

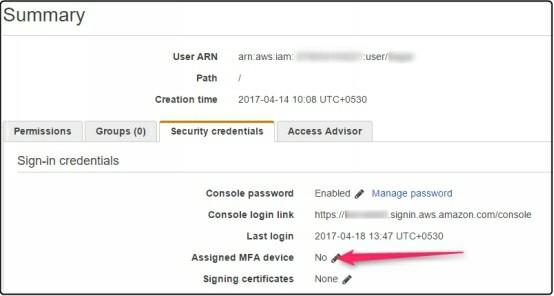
2. Search and open the IAM user dashboard.

3. In the left pane, click **Users** and select an **IAM user** for which you want to enable MFA.

4. I n t h e I A M u s e r **S u m m a r y** p a g e , s e l e c t

t h e **S e c u r i t y C r e d e n t i a l s** t a b a n d t h e n c l i c k **A s s i g n**

**M F A** d e v i c e e d i t b u t t o n a s s h o w n i n t h e f o l l o w i n g f i g u r e .



5. O n t h e **M a n a g e M F A D e v i c e** w i n d o w , s e l e c t t h e t y p e o f M F A d e v i c e t o a c t i v a t e . F o r t h i s e x e r c i s e , w e w i l l s e l e c t **A v i r t u a l M F A d e v i c e** o p t i o n a s s h o w n i n t h e f o l l o w i n g f i g u r e .



6. Click **Next Step** to proceed. On the warning message box, read the instruction and click the **Next Step** button to proceed.

7. On the next page, you will see a scan code that you need to scan using the Virtual MFA Application such as **Google Authenticator**.

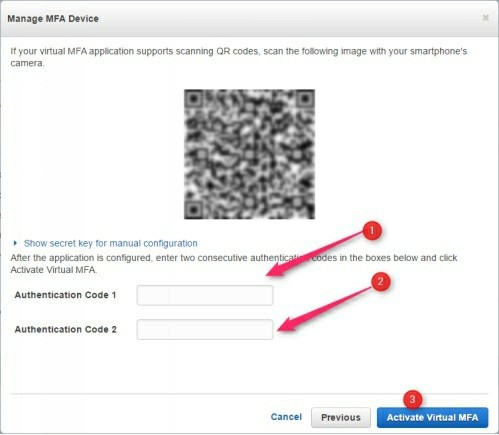
8. Once the code is scanned, the virtual MFA device (in our case

Android mobile) should be able to detect the AWS user account.

9. On the **Scan Code** page of AWS console, you also need to type two consecutive codes displayed on the Google

Authenticator application.

1 0. N o w c l i c k **A c t i v a t e M F A D e v i c e** b u t t o n t o p r o c e e d a s s h o w n i n t h e f o l l o w i n g f i g u r e .



N o t e : *The authentication code changes after every few seconds, so be careful while typing correct authentication code.*

11. Once the process is completed “The MFA device was successfully associated” message will be displayed. Click **Finish** to complete the wizard.

12. Now, whenever the IAM user will try to login to AWS console, he will need the dynamic security code along with username

and password.

That’s all you need to do to enable MFA for AWS IAM user. The same process can be followed to enable MFA for AWS root account. However, you must be logged in with the root account to do so.

4. Managing Virtual Private Cloud (VPC)

VPC is the backbone of the AWS cloud platform. In order to become AWS Solution Architect, you must have the good understanding of the AWS VPC and its components. If you are from the networking background, managing VPC might be very easier for you. However, candidates from the developing background should spend a good amount of time to get familiarized with AWS Cloud.

VPC is a separate, isolated, private network in the AWS cloud. By default, the instances from one VPC to another VPC cannot communicate to each other. For some reason, we may need to have multiple VPCs in the AWS cloud. Here, we will see how to create, manage, and delete VPCs.

For the more details of AWS VPC and its components, please visit the following link.

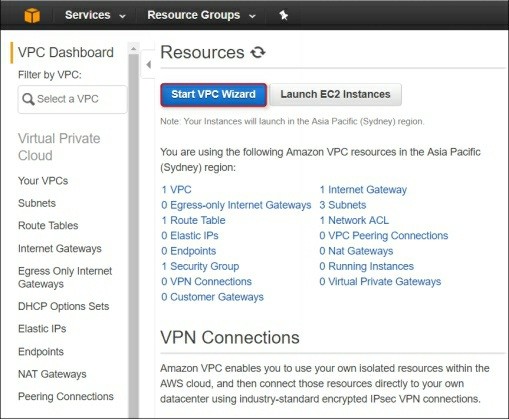
[Getting started with AWS VPC.](http://docs.aws.amazon.com/AmazonVPC/latest/GettingStartedGuide/getting-started-ipv4.html)

**Creating VPC in AWS Cloud**

In order to create VPC, you need to perform the following steps:

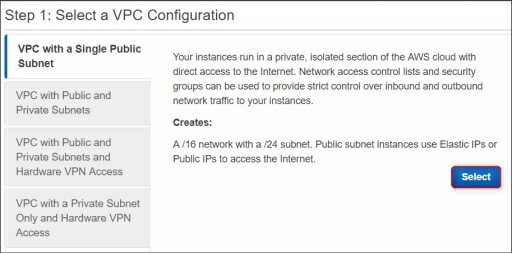
1. In the AWS console, search and open the **VPC** dashboard.

2. Click the **Start VPC Wizard** option as shown in the following figure.



3. On the **Select a VPN Configuration** page, click each of the options and review the description of the features provided by them.

4. Depending on your requirement, select the appropriate VPC configuration. Here, we will select the VPC with Public Subnet option as shown in the following figure.



***Note****: You can later add more subnets in the VPC and can customize your VPC options.*

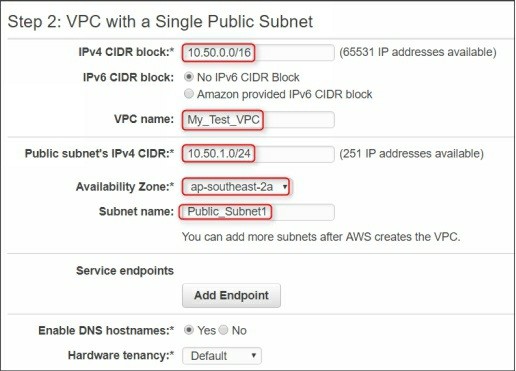
5. On the next page, specify the VPC name, subnet range, and Availability Zone etc. Here we are going to specify the following values:

IPv4 CIDR Block: **10.50.0.0/16**

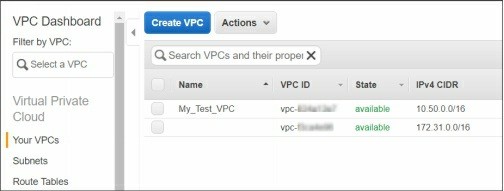
VPC Name: **My\_Test\_VPC**

Public Subnet CIDR: **10.50.1.0/24**

Availability Zone: **Select the first availability zone**. Subnet Name: **Public\_Subnet1**



6. Click the **Create VPC** button to proceed next. The VPC will be created and available in the VPC list as shown in the following figure.



**Creating and Adding Private Subnet in the Existing VPC** Since we had selected the VPC with Public Subnet option, so we need to create Private subnets separately. Private subnet does not have direct access from outside network and requires NAT gateway to access the Internet. Typically, backend and database servers should always belong to the private subnets.

If you are interested, you can visit the following link to know more about the

AWS VPC and subnets.

[AWS VPC and Subnets Getting Started.](http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Subnets.html)

To create a private subnet, you need to perform the following steps:

1. Select the Subnets option in the navigation pane and then click Create

Subnet.

2. On the Create Subnet page, specify the following values:

**Name tag**: Name of the subnet

**VPC**: Select the VPC in which you want to create subnet **Availability Zone**: Select the zone in which you want to create subnet

**IPv4 CIDR block**: Specify the subnet IP range

3. We will go with the following values:

Name tag: **Private\_Subnet1**

VPC: **My\_Test\_VPC**

Availability Zone: **ap-southeast-2b**

IPv4 CIDR block: **10.50.2.0/24**

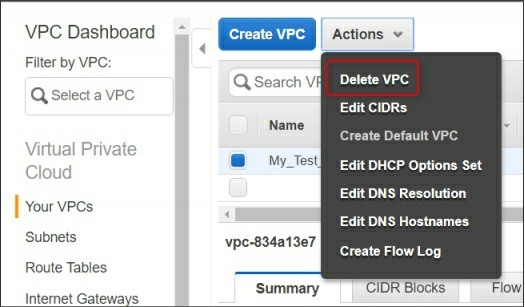


4. Click the **Yes Create** button to proceed. A new private subnet will be added to your existing VPC.

*Note: In an upcoming lab, we also will explore how to configure VPC peering between two or more VPCs to allow inter-VPC communication.*

**Deleting VPC**

If you no longer required any VPC for any reason, you can delete it anytime. For this, just select the VPC you want to delete, click **Actions** and then select **Delete VPC** to delete it as shown in the following figure.



*Note: Deleting VPC will also delete its associated components such as Subnets, NAT Gateway, Routing Table, Internet Gateways etc.*

5. Creating and Configuring Internet Gateways

An Internet gateway is an exit point for internal EC2 instance and the entry point for the outside public users. In AWS Cloud, you can logically consider an Internet Gateway as a Router that distinguishes the public and private network. Each public subnet requires an Internet gateway to provide services to public users and access the services from the internet.

Please visit the following link to know more about the AWS Internet

Gateways.

[Getting Started with AWS Internet Gateways.](http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Internet_Gateway.html)

When you create a VPC, an Internet Gateway is also created by default. If you wish to add additional or different internet gateway, which typically should not be required, you can do it. For this, you need to perform the following steps:

1. Select the **Internet Gateways** option in the left pane.

2. Click the **Create Internet Gateway** option and specify the name of

Gateway.

3. Click the **Yes, Create** button to complete the task.



4. In the **Internet Gateways list**, select the created IGW, and click the

**Attach to VPC** option.

5. In the **Attach to VPC** window, select the VPC that you want to attach with this IGW and then click **Yes, Attach** as shown in the following figure.



That’s all you need to do to create and attach the Internet Gateways in for AWS VPC. An Internet gateway can only be attached to a single VPC. However, a single Internet Gateway can be attached to the multiple subnets (routing tables) inside a single VPC.

6. Creating and Configuring NAT Gateways

NAT gateways are only required when you want to provide the Internet

access to your EC2 instances that are located inside the private subnets. There are two options to use with NAT gateways: Your own EC2 instance acting as NAT gateway or AWS NAT Gateway as a service.

NAT instance should be used for the Dev, QA and testing infrastructures where you can stop, start, scale, and manage it as per your own requirements. However, for the enterprise production servers, it is recommended to use NAT gateways. Because NAT gateways are managed by AWS and auto scalable as per the need and do not require any manual interactions. Here, we will focus on NAT Gateway as a service.

Please visit the following link to know more about the AWS NAT Gateways.

[Getting Started with AWS NAT Gateways.](http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-nat-gateway.html)

To create and configure NAT gateway, you need to follow the following steps:

1. Select the NAT Gateways option in the left pane and then click

**Create NAT Gateway**.

2. On the next page, you need to provide following two settings:

**Subnet**: Select the public subnet of your VPC to which your NAT

gateway will belong.

**Elastic IP**: Generate a new EIP that will be attached to NAT Gateway.



3. Click **Create NAT Gateway** option to proceed. After few minutes, the NAT Gateway will be created.

4. On the Next page, click **Close** to finish the task.

7. Configuring Routing Tables

Understanding and configuring rioting table in AWS Cloud is a little bit tricky and required a sound knowledge of networking fundamentals as prerequisites. Hope, you are already familiar with some routing concepts. Here, we will show you how to create, edit, change and configure routing tables.

Please visit the following link to know more about the AWS Routing Tables concept and usage.

[Getting Started with AWS Routing Tables.](http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/VPC_Route_Tables.html)

**Creating Custom Routing Table**

By default when you create a VPC, a default routing table “Main Routing Table” is also created automatically. All the traffic is by default directed to this default routing table. However, you can always create additional routing tables and can attach it to desired subnets. To design and implement a robust, secure, and scalable cloud Architecture, you must have the well understanding of VPC, Subnetting, NATing, VPC Peering, ACLs, and Routing.

To create a custom Routing table, you need to perform the following steps:

1. In the VPC Dashboard, click the **Create Routing table** option.

2. In the **Create Routing Table** window, specify the name of the routing table.

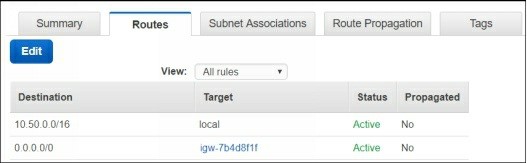
3. In the VPC drop-down list, select the VPC for which you want to create routing table and then click **Yes, Create** button.

4. Select the created routing table, here, you can see the various tabs.

The **Summary** tab provides the information regarding a number of attached subnets, whether it is main routing table or not, attached VPC etc.

The **Routes** tab provide the information regarding added routes, the attached IGW (for public routing table) or NAT gateways (for private routing table).

Rest of the tabs are self-explanatory.



5. If you wish to modify the route information, click **Edit** and add the routes you want to add. If you need to enable inter-communication between two VPCs, you need to create **VPC peering** and then you need to add VPC peering connection name in the routes tab. We will cover VPC Peering in a later section.

6. The **Subnet Associations** tab allows you to view, add, and remove subnets with this routing table.

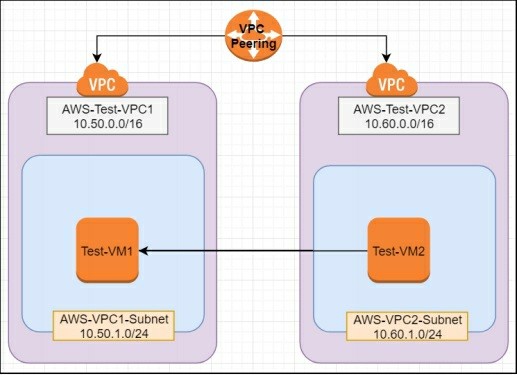
7. Simply click the **Edit** button and select or remove the checkbox in front of the desired subnets as shown in the following figure.



8. VPC Peering Between Two VPCs

As discussed earlier VPC is a separate isolated virtual network in AWS Cloud. By default, each VPC has its own Internet gateways, NAT gateways, Subnet range and scopes, Security group, and associated routing tables. In addition, resources from one VPC is not accessible from another VPC until unless you have configured VPC peering between them.

VPC peering allows one VPC to talk with another VPC inside AWS cloud. For example, let’s assume we have two VPCs named as AWS-Test-VPC1 and AWS-Test-VPC2. Each VPC has a public subnet inside which one instance is running in each subnet as shown in the following figure.



Test-VM1 cannot talk directly to Test-VM2 as it belongs to a different subnet of different VPC. However, instances from one subnet are allowed to talk

with instances of another subnet, but only if they belong to the same VPC. Please have a look at the IP scheme we have used for the above design.

In order to allow inter-VPC communication, you need to create a VPC peering connection and update routing tables associated with the subnets that you want to allow to communicate.

This lab exercise consists of the following tasks.

**Creating VPCs and Subnets**

To create a VPC and subnet you need to perform the following steps:

1. Select your desired region where you want to perform this activity such as **us-east-1**.

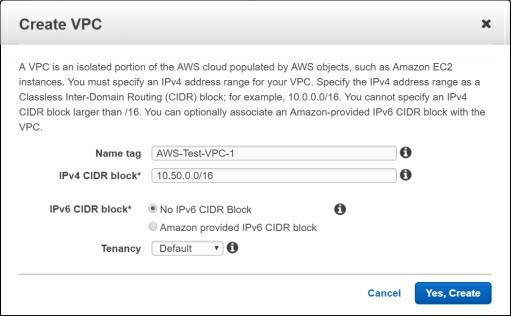
2. Open the VPC home page by navigating the following link.

<https://console.aws.amazon.com/vpc>

3. In the left pane, click **Your VPCs**. You will see a default VPC

created already.

4. Click **Create VPC** to create a new VPC. Create AWS-Test-VPC1 with settings as shown in the following figure.



5. Click **Yes Create** to complete the wizard. Similarly, create another

VPC named as AWS-Test-VPC2 with subnet scheme as 10.60.0.0/16.

**Creating Subnets**

To create a subnet, you need to perform the following steps.

1. In the VPC console, select **Subnets** in the left pane and then click

**Create Subnet**.

2. On the **Create Subnet** window, create a subnet with the following settings:

Name: **AWS-VPC1-Subnet**

VPC: **AWS-Test-VPC1**

Availability zone: **us-east-1a**

Subnet range: **10.50.1.0/24**



3. Now create another subnet with the following settings:

Name: **AWS-VPC2-Subnet**

VPC: **AWS-Test-VPC2**

Availability zone: **us-east-1b**

Subnet range: **10.60.1.0/24**

**Creating VPC Peering Connection**

Now you have two VPCs with one subnet in each VPC. But you have still not peered these VPCs. Let’s do it.

To create a VPC Peering connection between two VPCs, you need to perform the following steps:

1. Click **VPC Connection** in the left pane and then click **Create VPC Connection**.

2. On the **Create VPC Peering Connection** page, you need to specify the following settings:

**Peering connection name tag**: Name of the VPC peering connection such as VPC1-and-VPC2-Peer

**VPC Requester**: Select the source VPC such as AWS-Test-VPC1 as shown in the following figure.



3. On the **Select another VPC to peer with** section, you need to understand the following options.

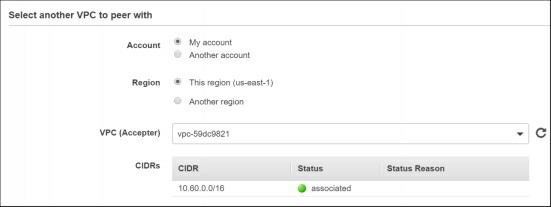
**Account**: It might be either your own account or another account for which you have sufficient permissions.

**Region**: VPC Peering can be within the same region or with a different region with a different account.

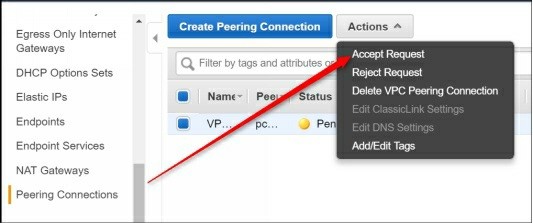
**VPC (Acceptor):** Select the destination VPC name to which you want to establish VPC peering such as AWS-Test-VPC2 in our case.

4. Refer the following figure for our lab exercise and complete the

Peering connection wizard.



5. Once you created a VPC Peering connection between the desired VPCs, it will be listed as pending in the VPC Peering list. You need to approve the VPC Peering manually as shown in the following figure.



6. Now, VPC Peering connection is done. However, instances from AWS-VPC1-Subnet would still not be able to communicate with AWS-VPC-2-Subnet. Think, why?

7. Yes, because you have not updated the routing information yet. Let’s do it.

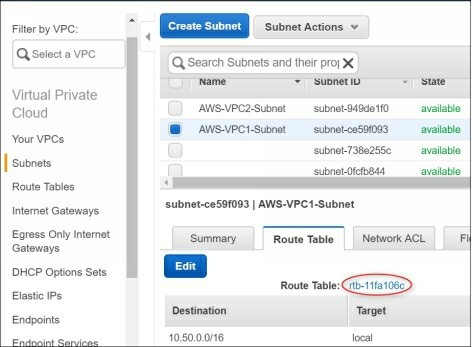
**Updating Routing Tables and Routes**

To update routing table and adding routes, you need to perform the following steps:

1. Select the **Subnets** option in the left pane.

2. Select AWS-VPC1-Subnet and then select the **Route Table** tab.

3. Open the attached routing table with this subnet in a new tab as shown in the following figure.



4. In this routing table, you can see that routes for the only local

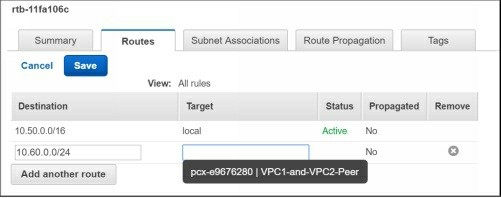
network are allowed. Here, you need to add routes for other subnet(s)

of other VPC which is in our case 10.60.1.0/24.

5. For this, click **Edit** and then click **Add another route**.

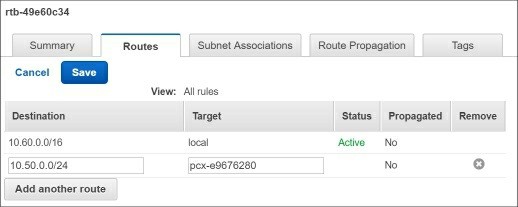
6. In the **Destination** section, type the subnet range of remote subnet.

7. In the **Target** section, select the VPC Peering Connection name that has appropriate requestor and Acceptor VPCs as shown in the following figure.



8. Finally, click **Save** to save the route information.

9. Using the same steps, go to the routing table attached to the AWS- VPC2-subnet, update the route for 10.50.0.0/24 and select the same VPC Peering connection.



Now resources (instances) between AWS-Test-VPC1 and AWS-Test-VPC2 can communicate to each other. However, you still need to use and configure Network Access Control List (ACL) and Security Groups as per the service and/or application requirements. These topics are covered in the separate sections.

9. Working with Amazon Elastic Cloud Compute

(EC2)

EC2 is one of the premier services of AWS cloud. One should have an excellent understanding and hands-on the Amazon EC2 components, features, services, and configuration. Amazon EC2 provides you almost everything that you may need for your Infrastructure as a Service (IaaS).

Managing EC2 is more than just launching or creating virtual machines (called as instances in AWS term). Once you become comfortable with Amazon EC2 services, you are almost on the track of becoming an AWS cloud expert.

Since EC2 is a broad level of topic, so we recommend you to remain focused on the basics of EC2 as a beginner. We will try to cover all the things that are necessary for an AWS Certified Solution Architect –Associate level. Since this book is focused on mainly for the hands-on lab exercises, so we will cover only the basic terms and theories for EC2.

The following are the major terms, components, features, and services you should know along with Amazon EC2.

**EC2 instance**: Amazon EC2 instance is a term used to define a virtual machine hosted on AWS cloud.

**Amazon Machine Image (AMI)**: AMI is a preconfigured virtual machine with all the predefined/preinstalled software and settings. It acts as a template for other instances that you would launch using this template image.

**AWS Region**: Region is a geographical location where AWS hosts its services with the locally hosted data centers. There are various AWS regions and increasing year by years. However, each region is completely independent and have at least two or more availability zones.

**Availability Zone (AZ)**: Each Availability Zone is isolated and can be considered as different datacenters. But the Availability Zones in a region are connected through low-latency links.

**Instance Type**: AWS instance types are covered in the later sections.

**Security Groups**: Security groups act as a virtual firewall for the

EC2 instances. Each VPC has a default security group. Security

groups are covered more in detail in the coming sections.

Now, you are familiar with the most common terms of EC2, let’s create and launch an Amazon Linux instance.

**Creating Amazon EC2 Linux Instance**

The following steps need to be followed to launch an EC2 instance.

1. In the **EC2 dashboard**, click **Launch Instance** button.

2. On the **Choose an Amazon Machine Image (AMI)** page, you can select either of the following options:

**Quick start**: Select this option if you want to launch a fresh instance and want to configure it as per your wish.

**My AMIs**: Here you will see the list of AMIs that have been created earlier

**AWS Market Place**: There are thousands of pre-configured VMs are available in the AWS market. Few of them are free and few of them are chargeable. If the pre-configured AMI fulfills your requirements, you can directly launch it. We will cover this in an upcoming section.

**Community AMIs**: There are few AMIs which are created and shared by the community people. You can search and find the desired pre-configured AMIs for your own use. But, these AMIs may not be trusted, so be careful before to use them.

3. For this demo, make sure that the **Quick Start** option is selected.

Select the platform (e.g. Linux or Windows) and variant (e.g. RHEL, Ubuntu) you wish to use and proceed to next page. For this demo, select the **Ubuntu 16.04** and proceed to the next page.



**Selecting Instance Type**

In the AWS Cloud, instance type defines the configuration and performance of your VMs. There are various kinds of categories and types of EC2 instances. Each type of instance has its own configuration, features, prices, and performance. For example, M4 and C4 instance categories provide greater network performance and better bandwidth than T2 instance categories such as T2 Medium and T2 Large instance. Plan well in advance before to choose instance categories and types for your server.

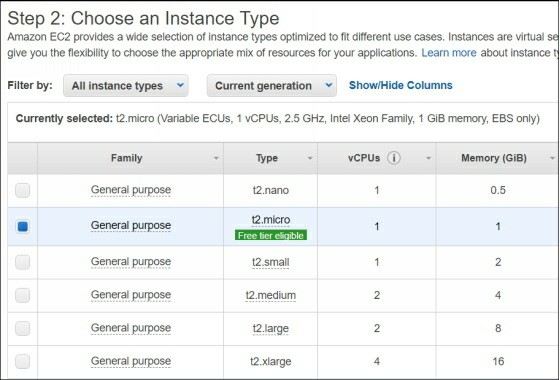
We highly recommend you to visit the following link to know more about the

EC2 instance types. Don’t try to remember everything, just go through once at high-level and get familiar with few terms.

[EC2 Instance Categories and Types](https://aws.amazon.com/ec2/instance-types/)

[EC2 Instance Pricing](https://calculator.s3.amazonaws.com/index.html)

For this demo, we just select the free tier eligible t2 micro and proceed to the next page.



**Specify the Instance Configuration Options**

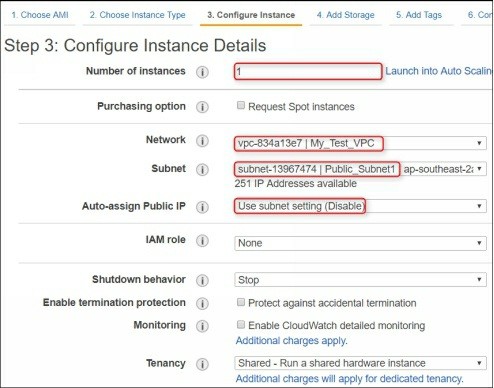
On the **Configure Instance** page, you need to specify various options. The important options are as follows:

**Number of instances**: Specify the no. of instances you want launch like this. For example, 1.

**Network**: Select the VPC in which you wish to launch the instance. For example, My\_Test\_VPC

**Select the subnet** in which you want to launch this instance. For example, Public\_Subnet1

**Auto-assign Public** IP: Keep it disabled as of now.



We have covered the rest of the options of this page in the upcoming

sections.

**Adding Storage Volumes**

On the **Add Storage** page, you can specify the size of Root volume or you can also add additional volumes to this instance as per your requirement.

*Note: We will cover the AWS Storage Types in an upcoming section.*

Proceed to the next page and specify the tag name such as **My\_Test\_VM1**.

**Selecting Security Group**

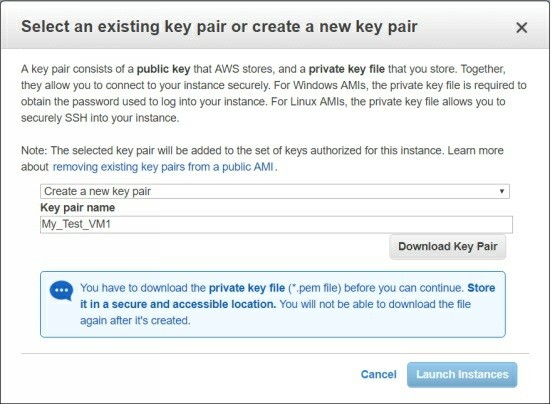
Security Group is one of most important feature that you should focus. Although, it has been covered in the later section. As of now, just create a descriptive security group that allows SSH to connect EC2 instance.



On the **Review** page, just review all the settings and options you have chosen. Go back and correct the settings or option you may wish. Finally, click Launch to launch the instance.

**Creating/Selecting Key Pair**

On the **Key Pair** page, you can either select an existing key pair if you have already created or you can create a new key pair. In AWS Cloud, Linux instances do not have a username and password authentication enabled for SSH access. By default, you can only access SSH of an EC2 Linux instance using the key pair.



Finally, click **Launch Instances** to complete the task. The instance creation process will start and your instance will be available in the EC2 instance list within few minutes.

**Practice lab exercise**: Using the mentioned steps in this hands-on lab, create one more EC2 with the following settings:

Instance type: T2 Micro/Ubuntu 16.04

VPC name: My\_Test\_VPC Subnet name: Private\_Subnet1

Auto-assign Public IP: Disable

Storage: Default

Instance name: My\_Test\_VM2

Security group name: My\_Test\_SG2

Key pair name: My\_Test\_VM2

*Note: We assume that you created the above-mentioned instance, as this instance has been used in few of the upcoming hands-on lab exercises.*

10. Creating and Configuring Security Groups Security groups act like a virtual firewall for EC2 instances. A Security Group controls the incoming and outgoing traffic for the EC2 instances. You can attach multiple security groups, having different rules, to a single EC2 instance. You can modify the rules for a security group at any time as per your requirements.

Please visit the following link to know more about the AWS Security Groups.

[Getting Started with AWS Security Groups.](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-network-security.html)

**Creating a Security Group**

To create a Security Group, you need to perform the following steps:

1. Select the **Security Groups** option in the EC2 dashboard.

2. Click the **Create Security Group** button.

3. On the **Create Security Group** window, specify the name, description, and VPC for which you want to create Security Group.

4. Select the **Inbound** tab and click **Add Rule** to add the desired rules.

**Security Group Rules**

A Security Group rule consists of the following options:

**Type**: Select the protocol or service for which you want to create a rule. There are multiple major services such as HTTP, SSH, ICMP are already available to use for you. However, you can also select Custom Protocol option if the service is not listed that you may wish to add.

**Protocol**: Typically this either will be TCP or UDP

**Port range**: This option specifies the specific port or port range for the selected protocol or service.

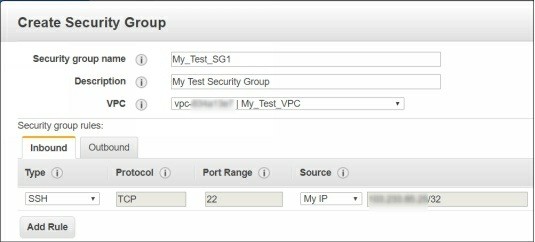
**Source**: There are three options in this drop-down:

**Custom**: Source can be either a specific IP address, specific another security group in your AWS cloud, or I can also be CIDR range.

**Anywhere**: This allows to access the specified protocol or service worldwide.

**My IP**: The protocol or service will only be allowed for

your connect ISP’s public IP. For SSH access, this is very handy and use full if you don’t have further security (such as OpenVPN and Linux bastion for SSH access) in your AWS cloud.



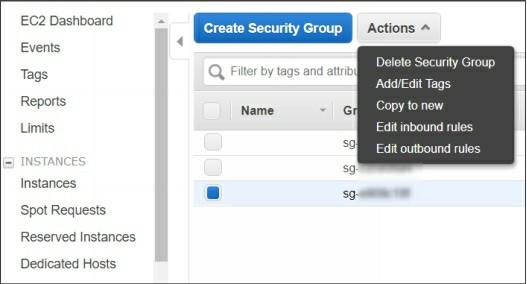
5. Similarly, you can create, add, and customize the outbound rules. By default, all the outgoing traffic is allowed for anywhere.

6. For this demo, just create an SSH rule for your own Public IP and click **Create** to complete the task. Now, this rule can be attached to any instance available in the selected VPC.

*Note: A security group can be referenced and linked with other security groups. However, if the two security groups belong to different VPCs, you will not be able to link those until you configure proper VPC peering.*

**Modifying Security Group**

To modify a security group, just select the security group and click the **Actions** button to perform the available functions. Please refer the following figure for the available options.



11. Managing Elastic IP Addresses

Elastic IPs are the manually assigned and fixed public IPs that you can assign to any AWS Instance. Instances running in the public subnet must have a

valid public IP in order to access the internet. Depending on the public subnet “Auto–Assign Public IP” setting, the instance may or may not get public IP assigned automatically. Or you may need to fix a specific public IP for a specific instance – This can be done using the Elastic IPs.

*Note: If you have assigned an Elastic IP to an instance and you stopped that instance, you will be charged for not using the associated Elastic IPs. As a free tier account, we recommend sticking with dynamically provided IPs rather than associating EIPs manually.*

Please visit the following links for more details about EIPs.

[AWS Elastic IPs Basics.](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html#eip-basics)

*Note: By default, all AWS accounts are limited to 5 Elastic IP*

*addresses per region.*

**Elastic IP Pricing**

Pricing for EIPs varies depending on the region you have selected. The following are the pricing for the Mumbai region. However, pricing for any AWS resource may slightly change at any time depending on the AWS policies and regulations.

$0.00 for one Elastic IP address associated with a running instance

$0.005 per additional Elastic IP address associated with a running instance per hour on a pro rata basis

$0.005 per Elastic IP address not associated with a running instance per hour on a pro rata basis

$0.00 per Elastic IP address remap for the first 100 remaps per month

$0.10 per Elastic IP address remap for additional remaps over 100

per month

To know the pricing of other regions, visit the [EC2 official pricing page](https://aws.amazon.com/ec2/pricing/on-demand/#Elastic_IP_Addresses) and select the appropriate region.

**Assigning Elastic IPs to EC2 Instances**

To assign an EIP, perform the following steps:

1. In the EC2 Dashboard, click **Elastic IPs** in the left pane.

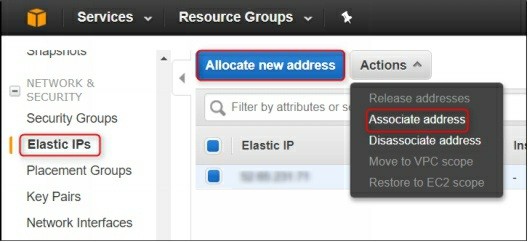
2. In the right pane, click **Allocate new address** option and then click

**Allocate**.

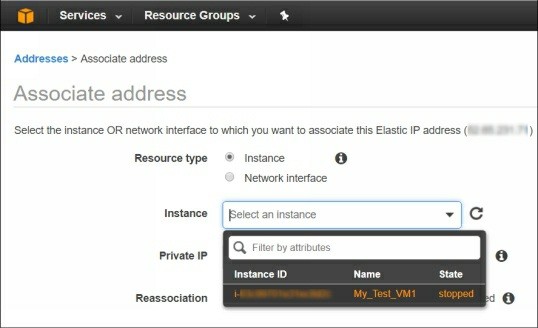
3. Select and right-click the allocated EIP and then click the **Actions**

menu.

4. Select the **Associate address** option as shown in the following figure.



5. On the **Associate address** page, select the instance to which you want to allocate this EIP as shown in the following figure.



6. Finally, click the **Allocate** button to finish the task.

12. Connecting EC2 Linux Instance Using PuTTy, Gitbash, and Console

In the previous sections, we have created an EC2 instance in the Public

subnet and allowed SSH protocol for My IP. However, we need to attach an Elastic IP to the public instance in order to access it from outside network. There are various options and tools to access a Linux EC2 instance using SSH protocol. The most popular such tools are PuTTy and Gitbash.

Here, we will just explain how to access EC2 Linux instances belonging to the Public subnet directly. In the later sections, we will also explain how to access EC2 Linux instances more securely using OpenVPN and Linux Bastion servers.

Once you have launched EC2 instances and assigned the Elastic IPs (if required), now you can connect your EC2 instance and perform the actions such as installing and configuring packages and services.

There are various methods that can be used to connect instance and Windows instances. Please visit the following link for more details if you are interested.

[Connecting EC2 instances.](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstances.html)

You can connect an EC2 Linux instance using the following method:

1. Connect EC2 Linux Instance Using Web Browser

2. Connect EC2 Instance Using [PuTTy](http://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html)

3. Connect EC2 Linux Instance Using GitBash

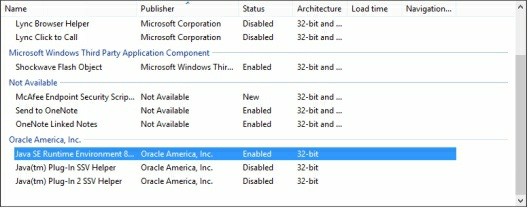
**Connecting EC2 Instance Using Web Browser**

In this method, you can use your Web Browser to connect EC2 instance. However, you need to install the compatible version Java for your browser. Few of the Web Browsers such as Chrome may not work properly for the EC2 connection. We recommend using Internet Explorer with a compatible version of Java installed. To connect your EC2 instance using Web Browser, you need to perform the following steps:

1. First, download the compatible version of **Java** and install it properly. The process is pretty simple and should not be a challenge for a technical guy.

2. Once the Java is installed, make sure you have also turned on the

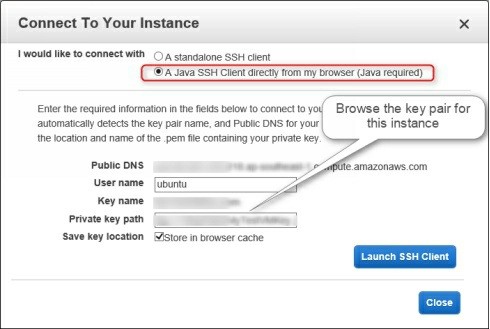
**Java Adds-on** in Internet Explorer as shown in the following figure.



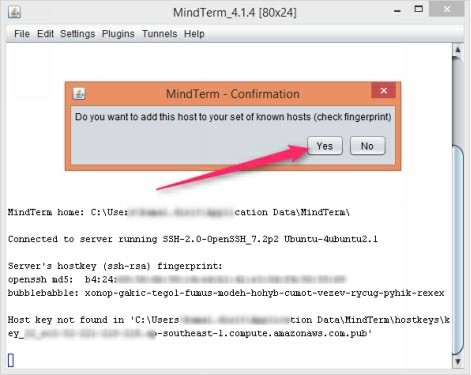
3. Now, select and right-click your EC2 instance and then select **Connect**.

4. On the **Connect To Your Instance** window, select **A Java SSH Client Directly** option.

5. Browse and select the key pair associated with this instance, and then click **Launch SSH Client** as shown in the following figure.



6. A new terminal console will be displayed and you will get the EC2 terminal console, where you can perform the desired tasks for your EC2 instance as shown in the following figure.



**Connecting EC2 Linux Instance Using PuTTY**

PuTTY is the most popular tool to access SSH and other remote connections such as Telnet. Most of the AWS Administrators use PuTTY to connect EC2

Linux instances. In this method, we are going to explain how to use PuTTY

to connect EC2 instances.

First, you need to download and install PuTTY and PuTTYgen tools on your local system. It’s pretty straightforward process and you should be able to do it very easily.

[Download PuTTY and PuTTYgen Tools.](http://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html)

*PuTTYgen allows you to convert .pem key format into .ppk key format. By default, the EC2 SSH key pair extension is .pem but PuTTY require .ppk key pair extension. So you need to convert*

*.pem key pair to .ppk format before to use with PuTTY.*

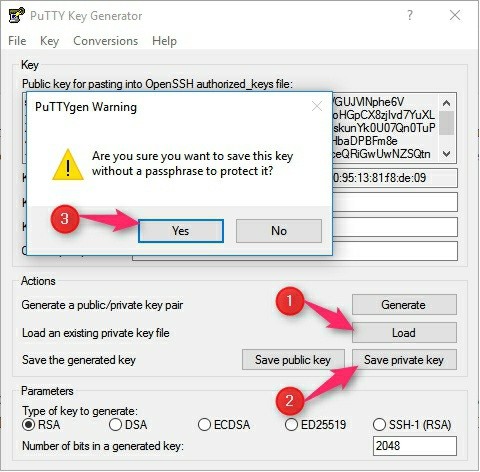
1. Once you downloaded and installed the above-mentioned tools, open the PuTTYgen console.

2. Click **Load** and browse the key pair, in the browse window select the

**File Types** as **All** and then select the **.pem** file.

3. Click **Load** to save the key pair file from .pem to .ppk. Optionally, you may also set the passphrase for this converting key for more

security.



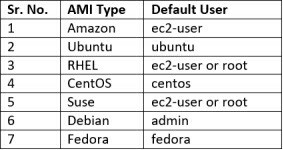
4. Now, open the PuTTY tool and type the Public IP or DNS name of your EC2 instance. The format should be followed as **user@ec2- public-ip-address** or **user@ec2-domain-name**. For

example: [ubuntu@54.55.56.57](mailto:ubuntu@54.55.56.57)

Default EC2 Instance Users

The default username for the EC2 instance may vary depending on the AMI

type and OS platform you have selected.



5. Continue from the previous step and scroll-down in the left pane, expand the **SSH** section and select **Auth** option.

6. Browse and select converted key pair (extension .ppk) and then click **Open** to open the SSH console.



A **Security Alert** will be displayed, better to read it and click **Yes** to proceed. Now, you should be able to access terminal of EC2 instance.

**Connecting EC2 Instance Using GitBash**

Apart from the above-mentioned methods to connect EC2 instances, there are few more tools such as GitBash that you can use to connect EC2 Linux instances. GitBash is my favorite tool for connecting EC2 Linux instances.

[Download Gitbash for Windows/MAC/Linux](https://git-scm.com/downloads)

You just need to download and install it, which is pretty simple. You can install Gitbash for Windows as well for Mac OS.

1. For the Windows version of GitBash, double-click the setup file and just follow the on-screen installation instructions.



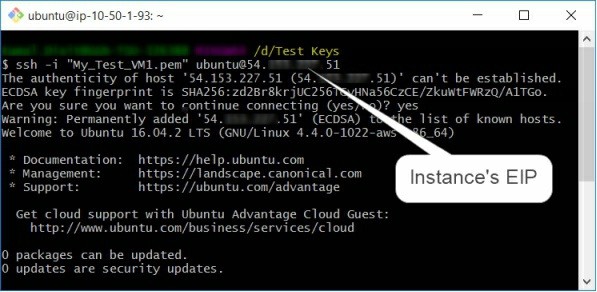
2. Once the GitBash tool is installed, open it and go the directory where you have saved the key pair file.

3. On the GitBash terminal, paste the EC2 instance’s public IP address or domain name in the following format:

***$ ssh -i “ec2-key-pair-name.pem” user@ec2-public-ip-address***

***$ ssh -i “ec2-key-pair-name.pem” user@ec2-domain-name***

4. Here, **ssh** is the command, **-i** means importing key, **ec2-key-pair- name.pem** is key pair name. For example, to connect an Ubuntu EC2 instance using its public IP, let’s have look at the following figure.



13. Connecting Private Instance using SSH Agent

Forwarding

This is an optional exercise, you should only proceed if you have created a second test VM named as My\_Test\_VM2 as mentioned in a previous lab exercise.

We have two instances: My\_Test\_VM1 belong to public subnet and has public IP and My\_Test\_VM2 belong to private subnet and does not have public IP. There are various methods to connect your private EC2 instance from your local machine such as using OpenVPN, Bastion and SSH Agent Forwarding tool.

Here, we will only explain how to access private instance using SSH agent forwarding service. Rest have been covered in the separate articles. SSH agent forward service allows you to carry the key pair file of an instance to another private or public instance.

**Allow SSH connection from one security group to another security group**

You can allow one security group to allow or deny specific protocol or port to another security group in your VPC. If the two security groups belong to different VPCs then you need to first establish the VPC peering. In our scenario, My\_Test\_VM1 is already allowed SSH incoming using public IP. But, we don’t have public IP for My\_Test\_VM2 instance, so we need to

allow SSH incoming connection for My\_Test\_VM2 instance’s security group (My\_Test\_SG2) for public instance’s (My\_Test\_VM1) security group (My\_Test\_SG1). For this, perform the following steps:

1. Select the **Security Groups** in the EC2 dashboard.

2. Copy the SG ID of the **My\_Test\_SG1** security group.

3. Select the **My\_Test\_SG2** security group.

4. Select the **Inbound** tab and click **Edit** to edit the rule.

5. In the **Source** drop-down list, select **Custom** and paste the group id of the **My\_Test\_SG1** security group.



6. Finally, click the **Save** button.

**Access SSH using SSH Agent Forwarding**

Now, open the GitBash tool from the location where you have saved your key pairs.

1. Execute the following command to start the SSH forward agent service

***$eval (ssh-agent)***

2. Add the key pairs you want to use with SSH forward agent

***$ssh-add –k My\_Test\_VM1***

***$ssh-add –k My\_Test\_VM1***

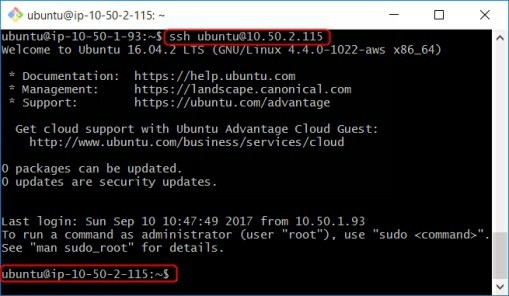
3. Connect the public instance with agent forward option

***$ssh –A ubuntu@public-instance-ip-address***

4. You will be connected to your public instance that is My\_Test\_VM1. From this instance, just type the following command to connect your internal private instance.

***$ ssh ubuntu@internal-instance-private-ip***

5. You should be able to connect your private instance as shown in the following figure.



*Note: We highly recommend to always use OpenVPN and Bastion server as a secure channel to connect your production servers. OpenVPN server implementation is covered in the upcoming section.*

14. Accessing EC2 Linux Instance Using RDP with

GUI Interface

By default EC2 Linux instance have only the CLI interface to work with the instance. However, sometimes, you may need to work with GUI interface for few reasons. But, before to access EC2 Linux instance using graphical interface, you need to install the graphical packages and the enable GUI remote access tool such as RDP to access it. Although, you can use various Graphical tools to work with EC2 Linux instances. These tools include MATE, TigerVNC and many more.

Here, we will explain to you how to install graphical packages in an Ubuntu

EC2 Linux instance and access remotely it using RDP.

My favorite GUI Ubuntu Desktop interface is MATE. So we will discuss here how to access EC2 Linux instance using MATE graphical interface.

**Installing MATE Desktop Environment in EC2 Ubuntu**

**Instance**

The following steps need to be followed to install MATE Desktop in Ubuntu system:

1. Execute the following command to add the MATE repository

***$ sudo apt-add-repository ppa:ubuntu-mate-dev/xenial-mate***

2. Run the following commands to update the repository and install the MATE packages

***$ sudo apt-get update***

***$ sudo apt-get install mate***

3. Run the following command to apply any of the available upgraded packages on your system.

***$ sudo apt-get dist-upgrade***

4. It will take around 10-12 minutes to complete and upgrade the package installation process.

5. If you are using local Ubuntu system, you can reboot the system, choose the MATE desktop as your environment and start using it. Since, we are focusing on EC2 instance, we need to enable a Remote Desktop method such as VNC or RDP.

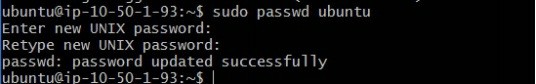
6. Once the MATE desktop environment is ready, install the

XRDP packages to enable RDP for Ubuntu instance. For this,

just execute the following command.

***$ sudo apt-get install xrdp -y***

7. Since by default EC2 instance does not have any user password set. Here, you need to set the password for your instance user. For Ubuntu VM, the default username is Ubuntu. Execute the commands mentioned in the following figure to set the password for ubuntu user.



8. Now your EC2 instance is ready to be accessed over the network using the GUI interface. However, you need to allow the incoming protocol and port for the method you wish to use, in this case, RDP (port-3389).

**Allow RDP Rule in the Security Group**

We will not discuss security group in detail here as we have already discussed it in a previous section. To modify the security group, perform the following steps:

1. Open the security group that is attached to your Ubuntu instance.

2. Select the Inbound tab and then click **Edit** to edit the rules.

3. In the **Edit inbound rules** window, set the values as shown in the following figure.

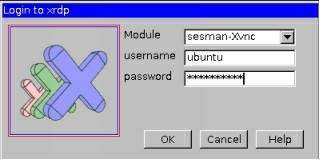


**Connecting EC2 Ubuntu using MSTSC (RDP)**

1. Once the security group is configured, open the **MSTSC** dialog box on a Window machine.

2. Type the instance public IP or public DNS name and click Connect.

3. On the **Login to xrdp dialog** box as shown in the following figure, type the username and password to connect it.



4. You should be able to access EC2 Ubuntu Linux instance with GUI

graphical interface using RDP.

15. Recovering and connecting EC2 instance if the

SSH key is lost

We know that EC2 Linux instances are accessible through the private keys (.pem keys) by default. However, SSH is allowed but you cannot use SSH password authentication to access Linux instance as it is disabled by default. So, what would happen if you lose the private key of your EC2 Linux instance? Here are few things that you should know before to proceed to this topic:

You cannot recover the private key for Linux instance if you have chosen **Root** Device Type as **Instance Store**.

You can connect and access your Linux instance in case of private key lost if you have chosen **Root Device Type** as EBS Store.

Keeping the above guidelines in the mind, let’s begin the whole process “How can we c[onnect EC2 Linux Instance if](https://protechgurus.com/connect-ec2-linux-instance-using-putty-gitbash-web-browser/) we lost the private key or .pem key file?

You need to perform the following steps in order to connect EC2 Linux instance if the private key is lost:

1. Stop the EC2 Linux Instance

2. Detach the Root Volume

3. Launching a new Temporary Instance

4. Attach the Root Volume to New Instance

5. Modify the authorized\_keys File

6. Reattach the Root Volume to the Original Instance

7. Start and Connect the Original Instance with New Private Key

Before starting this exercise, note down the following key information of your instance:

Instance ID, AMI ID, and Availability Zone of original Instance

Name of the Root Device volume such as /dev/sda1

Volume ID of Root Volume

**Stopping Original EC2 Linux Instance**

Stopping an EC2 instance is pretty simple, you just need to perform the

following steps:

1. Login to AWS console, navigate to the EC2 dashboard, and select the instance.

2. Right-click instance and select **Instance State** and then select **Stop** to stop it.

**Launching New Temporary Instance**

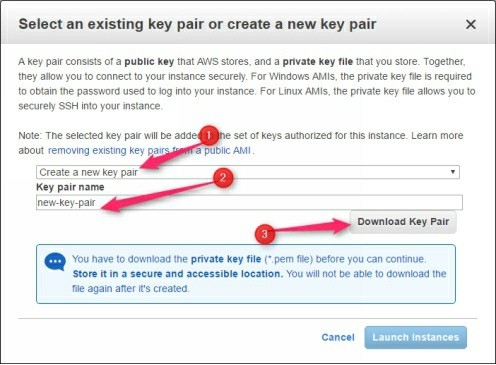
In this task, we need to launch a new EC2 instance with exact same settings and **in the same availability zone**. Refer the following information which you need to provide while creating a new instance.

**Instance Name**: Give a Temporary Name

**AMI**: Select the same as selected for the original instance

**Security Group**: Select the same Security Group that is attached to the original instance

**Key pair**: Create a new key pair named it as new-key-pair.pem as shown in the following figure, and store it in the safe location.



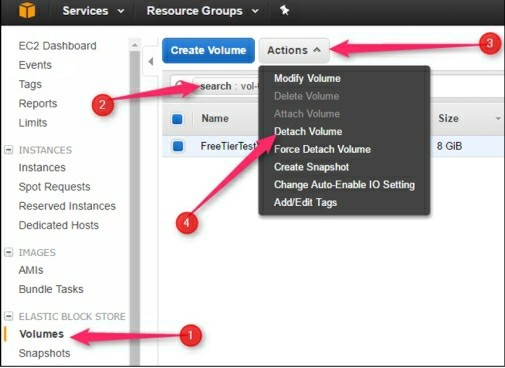
**Detaching Root Volume from Original Instance**

To detach the root volume from the instance of which you lost the .pem key, you need to perform the following steps:

1. Select the **Volumes** section in the left pane, type the volume ID of root volume of the original instance in the search box.

2. Select the **Root Volume**, click **Actions** and then select **Detach**

**Volume** to detach it as shown in the following figure.



3. On the **Warning** message box, click **Yes Detach**.

**Attaching Root Volume to Temporary Instance**

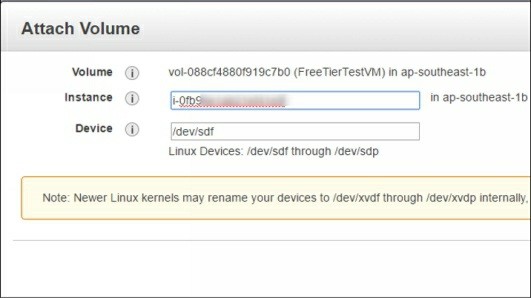
We assume that the Root Volume is still selected that you have detached in the previous steps. To attach the Root Volume to the (newly launched) Temporary instance, you need to perform the following steps:

1. Click the **Actions** menu and then select **Attach Volume** to attach a volume.

2. In the **Attach Volume** dialog box, type new instance name “**Temporary**” in the **Instance name** box. Alternatively, you can also type instance ID (of the temporary launched instance) if you remember or noted-down it somewhere.

3. Note down the new volume device name (such as /dev/sdf) and then click **Attach** to proceed.

*Note: Make sure the new instance and attaching volume both are in the same availability zone.*



**Mounting Attached Volume**

To mount the attached volume, you need to perform the following steps:

1. Select and right-click the new instance (Temporary) and open its console. We assume that the volume name was /dev/sdf.

2. Use the **lsblk** command to view the partitions and note down the new attached volume name such as /dev/xvdf1.

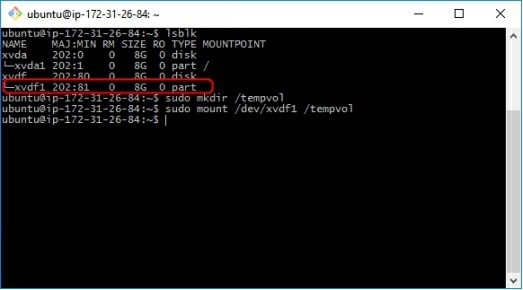
3. Use the following commands to create a mount point named as

/tempvol and mount the attached volume under it.

***$lsblk***

***$sudo mkdir /tempvol***

***$sudo mount /dev/xvdf1 /tempvol***



*Note: The volume may appear with the different name*

*depending on the Linux variant you use. For this demo, it shows as /dev/xvdf1.*

**Modifying the authorized\_keys File and Updating the New**

**Private Key**

Here, we will copy the SSH key file of the connected instance (temporary) to the mount volume’s user’s SSH authorized\_keys file. For this, you need to perform the following steps:

1. Use the following command to update the new key pair and to access the original instance:

***$cp .ssh/authorized\_keys***

***/tempvol/home/ubuntu/.ssh/authorized\_keys***

2. If the above command failed to execute, you may need to change the permission of /home/user/.ssh file with write permission. For this, execute the following command:

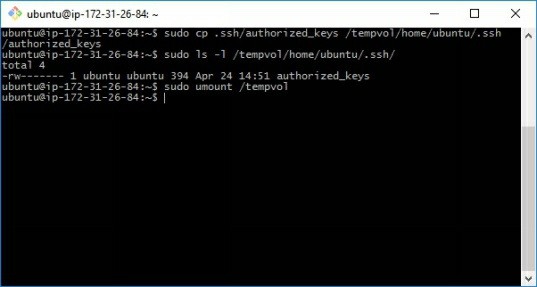
***$sudo chmod 757 /tempvol/home/ubuntu/.ssh/authorized\_keys***

3. Once the file is copied, you may revert back the permission as 755.

*Note: The username may vary depending on your EC2 instance variant. For example, ubuntu for Ubuntu Linux and ec2-user for Amazon Linux.*

4. Next, unmount the attached volume using the following command as shown in the below figure.

***$sudo umount /tempvol***



**Detaching Volume from Temporary Instance and Re- attaching With the Original Instance**

Now, you are almost done the required changes. Just, detach the old root volume form the temporary instance and –re-attached it with the original instance. For this, perform the following steps:

1. Go to the **Volumes** section, select the root volume (of the original instance), click **Actions** and select **Detach Volume** to detach volume.

2. Once the volume is detached, click again **Actions**, and select **Attach Volume** to attach it.

3. In the **Attach Volume** window, type the original instance name or ID, change the volume name as **/dev/sda1** and then click **Attach** as shown in the following figure.



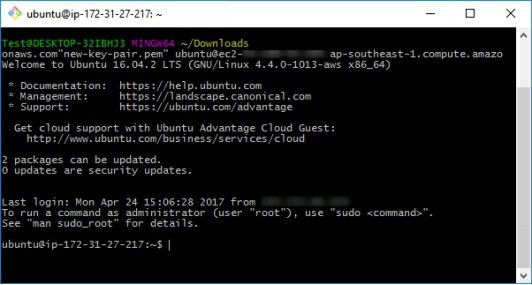
**Connect EC2 Linux Instance (Original Instance) With New**

**Private Key**

Now, you have done all the necessary tasks to recover your lost key pair. You

can connect the original instance with the newly created private key. For this, start the original instance and connect it with the key pair you created for Temporary instance that is in our case: new-key-pair.pem.

You should be able to access and connect EC2 Linux instance as shown in the following figure.



That’s all you need to do to connect EC2 Linux instance if the private key is lost.

16. Changing Instance type, security groups, volumes

& other settings

It is always recommended to plan well before to launch an instance. You should plan about the following things before to launch an instance:

Instance type that suits your application

VPC name in which the instance will be launched

Subnet name and type (private or public) Security group name and rules

SSH key you want to use

However, the good thing in the cloud is that you have the flexibility to upgrade and downgrade servers as per your requirement. But, you may need to stop your instance to change few settings which will finally cause the downtime for the production servers. Here, we will explain the major settings that you may wish to change for an EC2 instance.

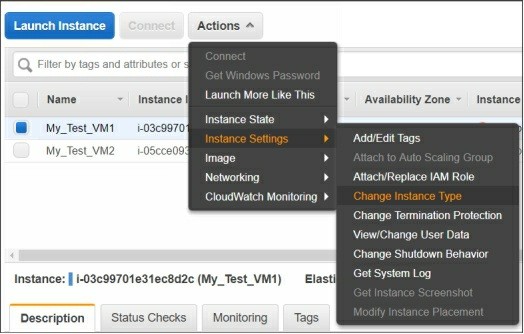
**Changing instance type**

Sometimes you may wish/need to upgrade/downgrade the hardware resources (RAM and vCPU) for an instance. For example, suppose you have chosen T2.Large instance type for your xyz application. But later you realize that

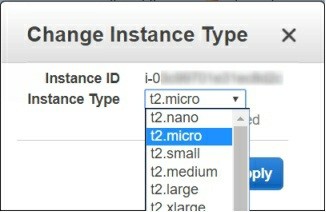
your application is not using so much resources and you are paying more than you need. Here, you can downgrade your instance form T2.Large to T2.Small instance type. For this, first make sure that your instance is stopped then perform the following steps:

1. Select your instance, click **Actions** and navigate to the **Change**

**Instance Type** as shown in the following figure.



2. On the **Change Instance Type** window, as shown in the following figure, select the instance type you may wish to use, and then click **Apply**.



**Accidental Termination Protection**

A mistake can be made by anyone anytime, we should always be ready with a mitigation plan to reduce the risk as much as possible. EC2 termination

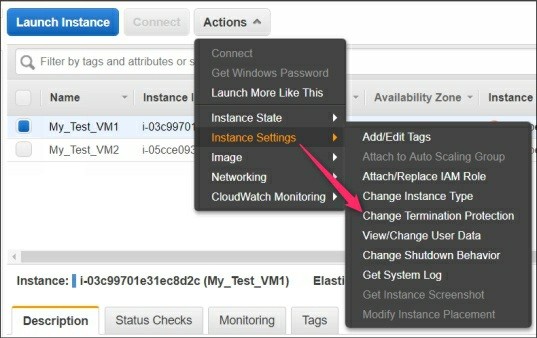
means simply destroying your EC2 instance permanently. Once you click the terminate button and confirm the Yes, you will no longer be able to recover that instance and its data. Even, AWS will not be able to help you in recovering the terminated instance. We recommend you to enable Termination Protection for all your critical instances.

For this you need to perform the following steps:

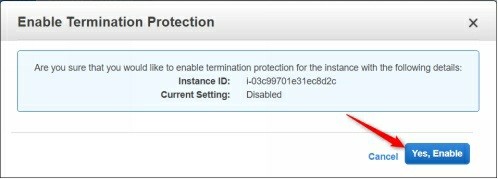
1. Select the instance for which you want to enable Termination

Protection and then click **Actions**.

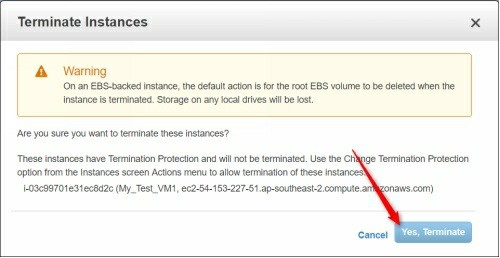
2. Navigate to the **Change Termination Protection** option as shown in the following figure.



3. On the **Enable Termination Protection** window, click the **Yes, Enable** as shown in the following figure.



4. Now the termination protection for your instance is enabled. If someone will try to terminate the protected instance, the **Yes, Terminate** button will be greyed out as shown in the following figure.



Now, first, you would need to disable the Terminate Protection feature manually before to terminate this instance.

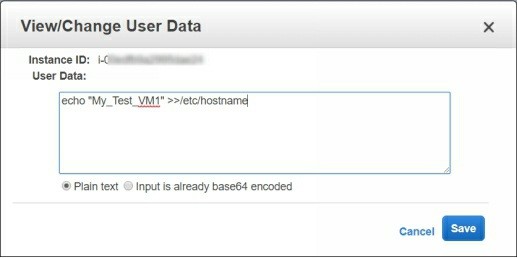
**Specifying Auto Start Script with EC2 Instance**

Sometimes, you may need to run a series of commands (or script) that should execute when the instance starts. This works in the similar way as we set the login scripts or startup scripts in Windows machines. This can be done by using a setting called View/Change User Data.

1. Just navigate to the **View/Change User Data** option and type the commands or script you want to execute when the instance starts.

2. For example, the following command will set the instance’s

hostname automatically as My\_Test\_VM1, as shown in the following figure, when the instance starts.



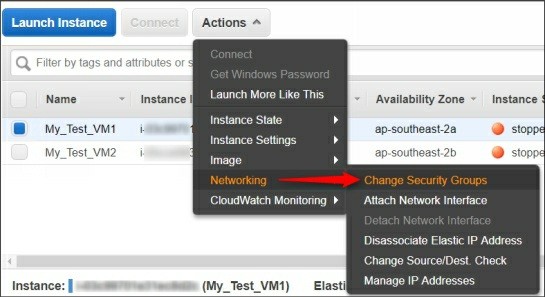
The View/Change user data option is very handy and useful in Auto Scaling that we will cover in the later sections.

**Changing/Adding Security Groups to EC2 Instance**

As discussed earlier, security groups control the incoming and outgoing traffic for the EC2 instances. You can always add or remove the security groups to and from your EC2 instance without rebooting.

1. For this, just select the instance, click **Actions**, and navigate to

**Change Security Groups** option as shown in the following figure



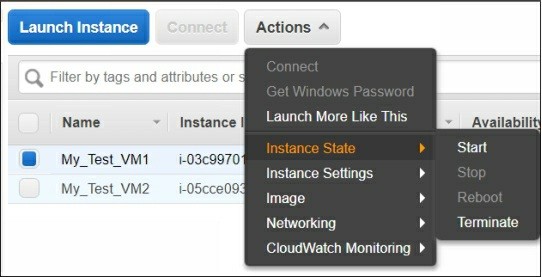
2. In the **Change Security Groups** window, select or remove the checkbox in front of the security group that you want to add or remove and then click **Assign Security Groups** to apply the changes.

17. Start, Stop, Reboot, and Terminate EC2 Instance

Starting, stopping, rebooting, and terminating an EC2 instance is pretty

simple and straightforward. Just select an instance that you want to terminate, click **Actions** and navigate to the **Instance state** option as shown in the following figure.

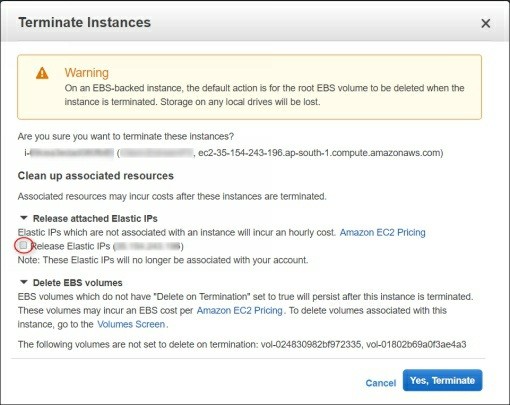
Here you can see the four options. Start, stop, and reboot options are self- explanatory and do not require to be explored further.



Terminating an instance means removing or deleting the instance along with its entire data. However, the attached volume will also be removed or not, it depends on the setting you configured on the volume properties.

By default, root volume is marked as delete on termination but the additional volumes are not marked delete on termination by default. So if you want to keep the attached volume, just remove the Delete on terminate option before to terminate the instance.

If there is an EIP is also attached to that instance, you have the option to keep this EIP or release this EIP while terminating the instance.



As discussed earlier, if the **Termination protection** is enabled then first you

need to disable it before you could terminate an instance. Once the termination process is started, it may take around half and an hour to flush from the instance list, but you will have no option to do anything with the terminated instance. So be very careful before to terminate any instance.

18. Creating and configuring Elastic Load Balancer A Load Balancer is a service that distributes and balance the load into different servers that are grouped together to serve a specific service. For example, you can configure a load balancer for a web server. You can then have multiple servers behind this ELB with the same settings.

Please visit the following link to know more about the AWS Load Balancers type and features.

[Getting Started with AWS Load Balancers](http://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/load-balancer-getting-started.html)

[AWS Load Balancing Pricing](https://aws.amazon.com/elasticloadbalancing/pricing/)

**Load Balancer Modes**

The ELB can act in either of the following modes:

1. Active-Active: In this mode, multiple servers can share a load of users’ traffic simultaneously.

2. Active-Passive: In this mode, one server waits until the primary server fails to respond for a specific time duration.

**AWS Load Balancer Types**

If we talk about AWs load balancer concept, there are three types of load balancers:

1. **Application load balancer**: Choose an Application Load Balancer when you need a flexible feature set for your web applications with HTTP and HTTPS traffic.

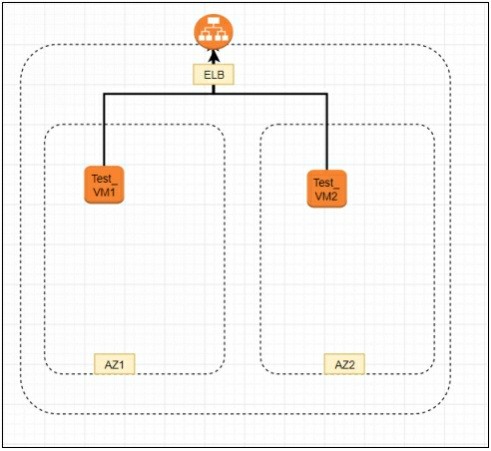
2. **Classic load balancer**: Choose this Load Balancer when you have an existing application running on the EC2-Classic network

3. **Network load balancer**: Choose a Network Load Balancer when you need ultra-high performance and static IP addresses for your application.

[You can explore more about the load balancer here.](https://www.sumologic.com/aws/elb/aws-elastic-load-balancers-classic-vs-application/) [Comparison between AWS Load Balancers](https://aws.amazon.com/elasticloadbalancing/details/#compare)

Here, we are going to show you how to configure and use an Elastic Load

Balancer using the below network design.



Before to proceed with ELB creation, make sure that you meet the following prerequisites:

1. Make sure you two subnets and two availability zones, at least one subnet in each availability zone.

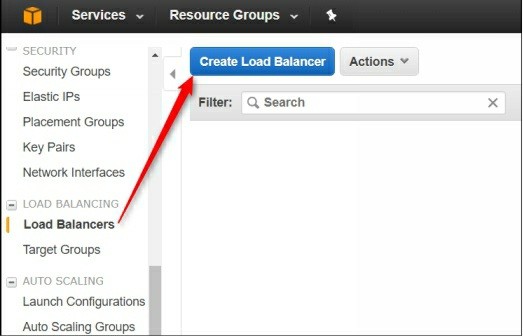
2. Make sure you have an application running on at least a single instance of any availability zone. For the demo purpose, having multiple instances is not mandatory.

Once you are ready, let’s begin with the hands-on practice.

**Create an Application Load Balancer**

To create and configure an ELB, you need to perform the following steps:

1. In the **EC2 dashboard**, click **Load Balancers** in the left pane, and then click **Create Load Balancer** as shown in the following figure



2. On the **Select load balancer types** page, select the type of load balancer that suits most with your application, for example, application load balancer and proceed to the next page.

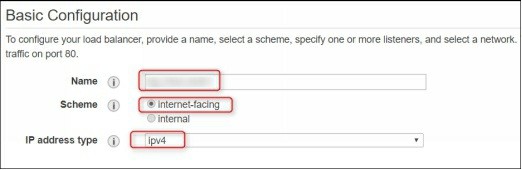
3. In the Basic configuration section, you need to set the following values:

**Name**: Name of the ELB

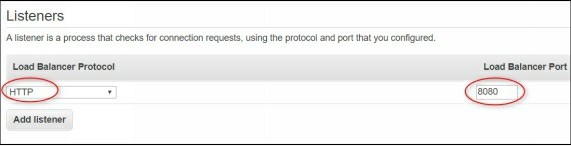
**Scheme**: Internet-facing or internal facing

**IP address type**: IPv4 or dual stack

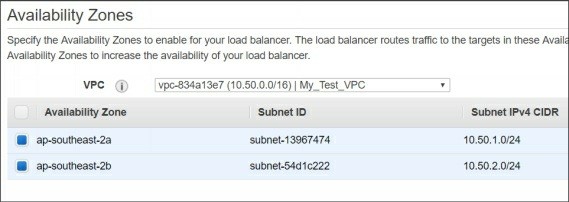
4. For the demo purpose, select the setting as given in the following figure.



5. In ELB, you also need to specify the listener settings. A listener checks the health status of your instances that hosting the application. Since application load balancer only supports the HTTP or HTTPS protocol, so you can host only web services with Application load balancer. For example, if your web server is running on 8080 port, you have to set the listener as given in the following.



6. In the **Availability Zone** section, select the VPC in which your application is hosted. Select at least one subnet per availability zone and proceed to the **Configure Security Groups** page.

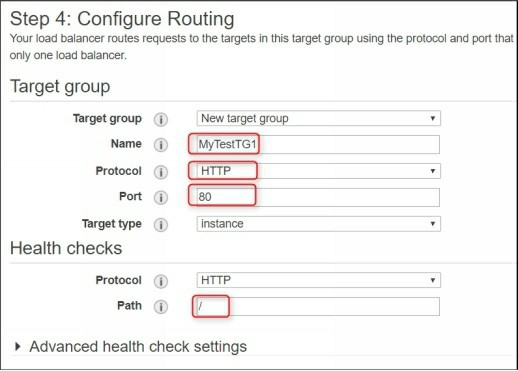


7. Select **Create a new security group** option and specify the name and description

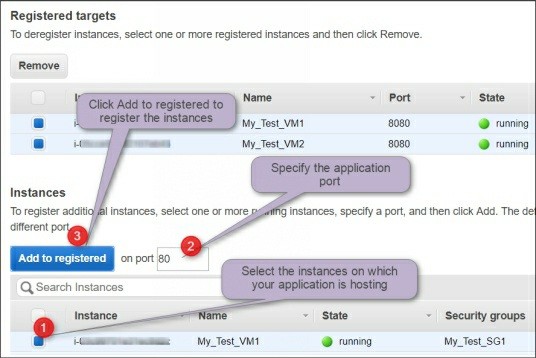
8. In the **Rule** section, enable the port on which the application is running, in our case TCP/8080 as shown in the following figure.



9. On the next page, specify the **Target name**, **Protocol**, **Port**, and **Health check path** as per your application settings as shown in the following figure.



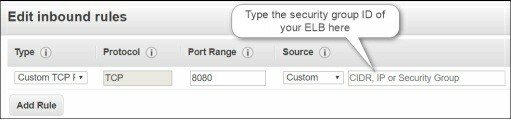
10. On the register targets page, select the instance, application port and add them to the registered target group as shown in the following figure.



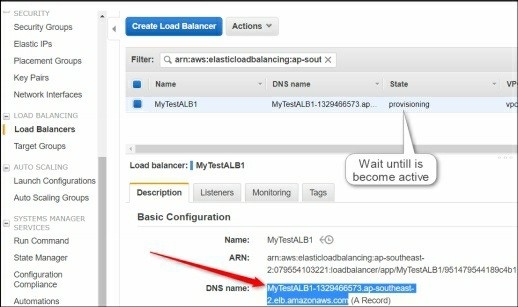
11. Proceed to the next page, review the settings you have selected and finally click **Create** to finish the wizard.

12. After few minutes, your Application load balancer will be active.

Now, allow your ELB’s security group to communicate with the application hosting instances. For this, you need to modify the instances’ security groups as shown in the following figure.



13. Navigate to the **Load balancers** page in the EC2 Dashboard and copy the load balancers **A record** as shown in the following figure.



**Domain Registration for ELB**

If you also own the DNS registration account, you can register this ELB’s A record with respective DNS record such as www.myapp.com or you can use the this A record directly in the browser to check the application functionalities with Application Load Balancer.

Similarly, you can also create and configure Network load balancer and Classic load balancer. However, few steps and options will different obviously.

19. Scheduling Auto Snapshot of Volumes

Snapshots are good and cheaper solution for the EC2 instance’s backup. You can take a snapshot of any attached volume either manually or schedule it for a specific recurring time interval. For this, you need to create a rule that will specify the following values:

1. Event source: The source can be either an event pattern or a fixed schedule

2. Rule name and description: Name of the rule and description.

3. Rule permission: An IAM role that grants the permission to take a snapshot.

4. Targets: Target can be a lambda function, API call, SNS alert etc.

**Creating Rule**

For now, you can just follow the CloudWatch rule steps and not to go deeper inside it here. We will cover CloudWatch in detail in the upcoming sections.

To create a CloudWatch rule for auto snapshot backup, open the CloudWatch console and select Rules in the left pane.

1. On the **Rule** page, click **Create rule** to create a new rule as shown in the following figure.



**Defining Event Source**

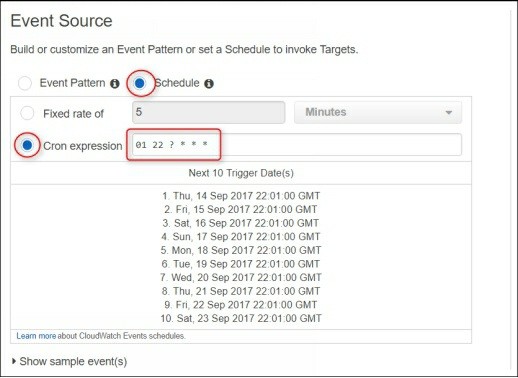
An event source can be either a pattern or a schedule. Schedule further can be configured with fix rate of time or in the format of the cron job. Here, we will select the Schedule with cron expression. We assume that you are already familiar with the cron schedules. AWS cron scheduler is almost same with

slight differences in the syntax as what we use for Linux cron scheduler.

We recommend you to please have a look at the following cron schedule details and examples:

[AWS Schedule Expressions for Rules](http://docs.aws.amazon.com/AmazonCloudWatch/latest/events/ScheduledEvents.html)

For example, in the below figure, we have configured a cron schedule that will take a snapshot of a volume on daily basis at 22:01 PM (GMT).



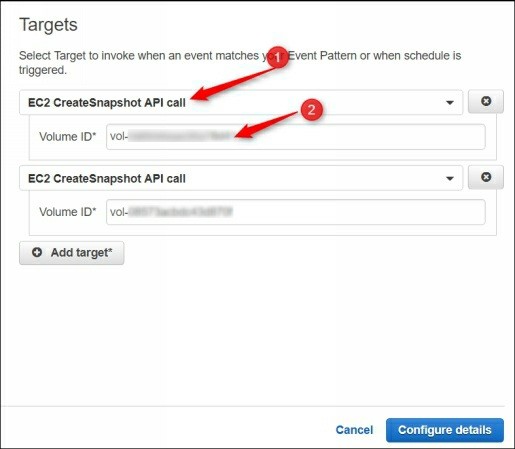
**Cron Expressions Syntax and Examples**

The following figure shows few of the most common cron schedule examples. Hope, this will help you to understand Cron Expressions Rules.



**Specifying Target**

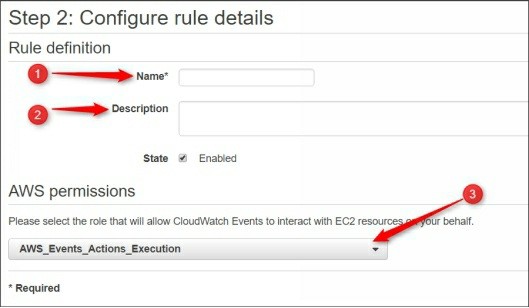
The Target can be various types, but for this exercise, we just need to select the **EC2 CreateSnapshot API**. You also need to specify the volume ID of the volume of which you want to schedule auto snapshot backup. You can add multiple targets to a single CloudWatch rule.



**Configure Rule and Details**

1. On the **Configure rule** details page, specify the name and description of the rule.

2. You also need to select the IAM role that has permission to create snapshots. By default, there is a preconfigured role that you can use.



Finish the wizard once you are satisfied with your settings and values. That’s all you need to do to automatically schedule snapshot for an EC2 instance’s volumes.

When the scheduled time will come, your snapshots for the specified volumes would be created and available in the snapshot list in the EC2 dashboard.

20. Creating AMI and Recovering EC2 Instance using AMI

Amazon Machine Image (AMI) is a complete backup image of a running or

stopped instance. When you create an AMI for an instance, all the system configuration along with all the attached volumes and data are copied in the form of a single file called AMI. This AMI image or cloned VM is then stored in the S3 storage.

Please visit the following link to know more about the AMI creation process.

[Amazon EBS-Backed Linux AMI](http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html)

The AMI can be used to launch a new instance with similar settings and configuration of the instance using which you had created this AMI. This is the best option while migrating a server from one VPC, availability zone, or region to another VPC, availability zone, or region.

Here we will create the AMI of My\_Test\_VM1 instance and then will launch in another availability zone and subnet. Before this, we will create a file and a user on source VM so it could be verified that the settings and files are available in the destination VM. However, IP address of new instance will, of course, be different from the source instance.

**Creating some users and files in the source Instance**

Login to My\_Test\_VM1 and create a file named My\_File1 and a user named testuser1 as shown in the following figure.



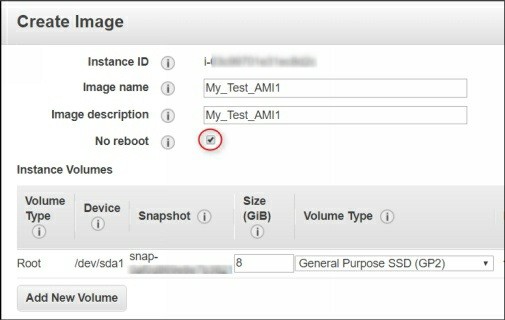
**Creating Image of an EC2 instance**

Now, select and right-click the **My\_Test\_VM1** and navigate to the **Create**

**Image** option as shown in the following figure.

**Specifying AMI settings**

In the **Create Image** window, type the name and description of the AMI. Select the **No reboot** check box if you don’t want to reboot the running instance while creating AMI. However, if possible, it is recommended to leave this check box unchecked.



If required, you can modify the volume size or you can also add additional volumes as per your requirement. Finally, click the **Create Image** button.

The AMI create process will start and after few minutes, the AMI will be created and available in the AMI list as shown in the following figure.

**Launching an EC2 Instance using AMI**

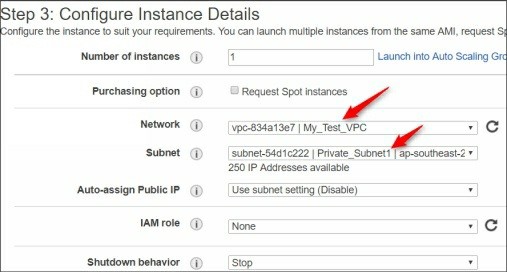
1. Select the AMI and click **Launch** to launch a new instance using this AMI. The launching AMI process is almost same as launching a new instance.

2. On the **Choose instance type** page, select the instance type such as

T2.Micro and proceed to the next page.

**Configuring Instance Details**

1. On the **Configure Instance Details** page, select the destination VPC, subnet (in which the instance will be launched) and other settings as per your need as shown in the following figure.



2. On the **Storage** page, modify the storage volume size or add new volume, if required.

3. On the **Add tags** page, specify the instance name such as

My\_Test\_VM3.

**Configuring Security Group**

On the **Configure Security Group** page, select the existing security group that you want to attach with this instance or create a new security group with appropriate rules.

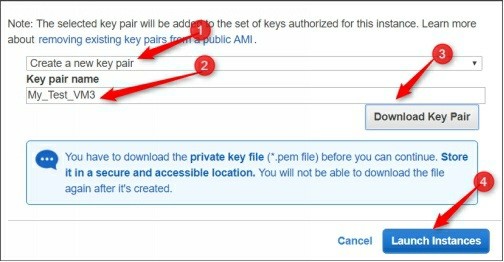
**Reviewing and launching Instance**

1. On the **Review** page, review the settings you selected and click

**Launch**.

2. On the **Key pair** page, select an existing key pair or create a new key pair for this instance.

3. Finally, click **Launch Instance** as shown in the following figure. The instance will be ready within few minutes.



**Login in to the New Instance**

Now login to the launched instance and verify whether the data from the source instance is available or not.

1. On the GitBash or Linux terminal, execute the following command.

***$ eval (ssh-agent)***

***$ shh-add –k My\_Test\_VM1.pem***

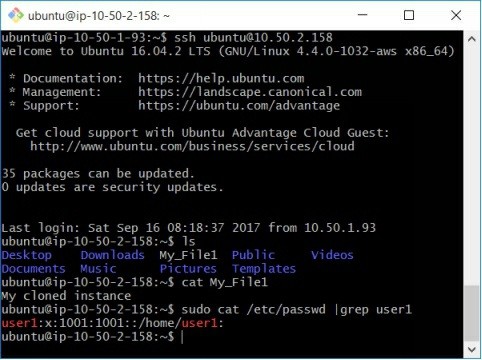
***$ ssh-add –k My\_Test\_VM3.pem***

***$ ssh –A ubuntu@<my-test-vm1-public-ip>***

2. After connecting the first VM, execute the following command inside the connected instance to connect private internal instance.

***$ ssh ubuntu@<my-test-vm3-private-ip>***

3. As you can see in the below figure, the file and user is present that you have created in the source instance.



That’s all you need to do to create, launch, and use Amazon Machine Image

(AMI).

21. Configuring CloudWatch Monitoring

Monitoring is one of the most important factors of managing IT infra in a proper way and having the pro-active solutions before anything goes wrong with your infrastructure. CloudWatch helps you to monitors operational and performance metrics for your AWS cloud resources and applications. There are hundreds of metrics that can be used with CloudWatch.

The most common metrics that every administrator would like to use are CPU, RAM, Disk, and Network consumption and usage report. However, basic CloudWatch monitoring does not support all of the metrics and also have the longer three sold value than Advanced CloudWatch monitoring which is more expensive than the basic CloudWatch monitoring.

[Basics of Amazon CloudWatch](https://aws.amazon.com/cloudwatch/)

**Amazon CloudWatch Pricing**

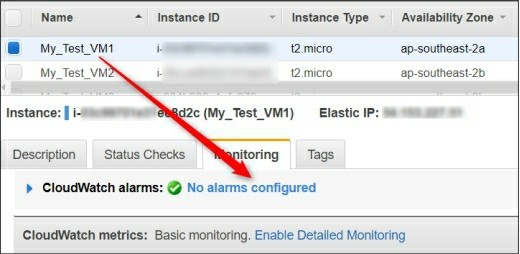
You should be aware of your free tier limitations for any AWS services else you might be charged by AWS if you have used any resources beyond the free tier limitations. For AWS CloudWatch, the following link explains the free tier limitations and the pricing details for the detailed monitoring if you wish to configure.

[AWS CloudWatch Free Tier Limitations and Pricing](https://aws.amazon.com/cloudwatch/pricing/)

Here we will show you how to use CloudWatch to monitor CPU, Disk, and Network resource usage and get alerts when the resource utilization exceeds the limit beyond the specified values.

**Selecting an Instance**

In the AWS console, select the instance for which you want to enable CloudWatch monitoring for example My\_Test\_VM1. You can see, by default, there is no alarms are configured.



**Creating a CloudWatch Alarm**

To create a CloudWatch alarm, you need to perform the following steps:

1. Click **Create Alarm** to proceed. On the **Create Alarm** window, make sure that the **Send a notification to** check box is selected.

2. Select the SNS topic if you have already created or click to create now.

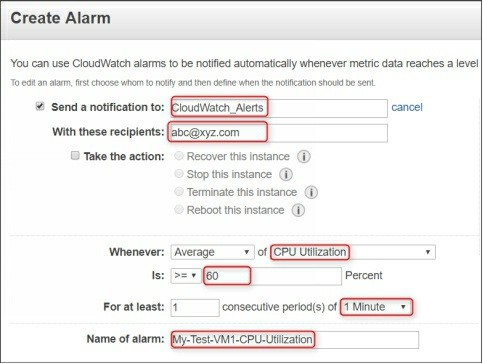
Type the **topic name** and **email ID** where the email alerts will be sent.

Select a metric such as CPU utilization, Disk reads, or Network in/out etc. that you want to enable.

In front of the **Is** box, select the formula and alert value such as

60% utilization.

In the **Consecutive period** box select the number of minutes as three sold value till then CloudWatch wait before to send any alert. In the **Name of the alarm** box, specify the name of alarm. Refer the following figure for a demonstration.



Now CloudWatch will monitor the CPU utilization of your instance and will

send an alert when the three sold value will exceed.

*Note: Before you can get alerts from AWS CloudWatch service, you need to request and confirm your subscription using the Simple Notification Service (SNS) console. The next section covers SNS in details.*

22. Configuring Amazon Simple Notification

Service (SNS)

Amazon SNS is a web service used to coordinate and manage the delivery and sending messages to its subscribers. If you want to explore more about the Amazon SNS, please have a quick look in to the following links:

[Basics of Amazon SNS](http://docs.aws.amazon.com/sns/latest/dg/welcome.html)

[Getting Started with Amazon SNS](http://docs.aws.amazon.com/sns/latest/dg/GettingStarted.html)

**Amazon SNS Free Tier vs Paid Pricing**

The following table explains the limitations of the free tier and the pricing details for using Amazon SNS.

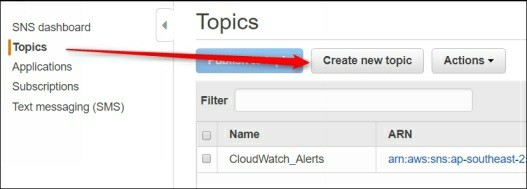


In order to use Amazon SNS with other Amazon services such as Lambda, CloudWatch etc. you need to perform the following tasks.

**Creating a Topic**

To create an SNS topic, you need to perform the following steps:

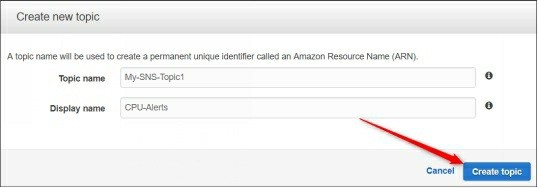
1. Open the **Amazon SNS console** and click **Create new topic** under the **Topics** section.



2. In the **Create new topic** dialog box, type the **Topic name** and

**Display name** and then click **Create topic** as shown in the following

figure.

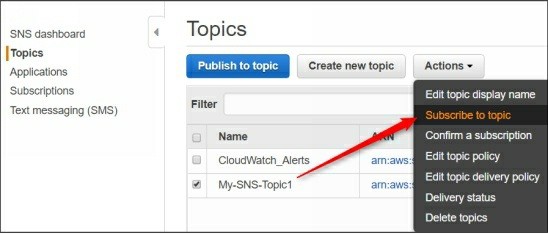


**Subscribing to the topic**

1. Once the topic is created, it will be listed in the topic list.

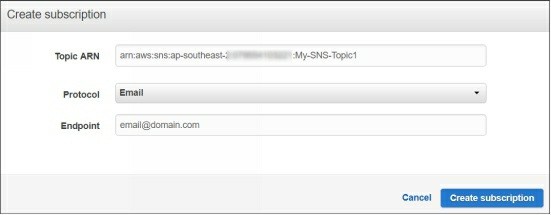
2. Select the created topic, click **Actions** and select **Subscribe to topic**

as shown in the following figure.



3. On the **Create subscription** dialog box, select the protocol that you want to use with SNS. We have selected **Email** but you can also select HTTP, HTTPS, AWS Lambda, or SMS depending on your choice and need.

4. In the **Endpoint** text box, type the **Email ID** where the alerts will be sent and then click **Create subscription**.



5. Once you receive the email in your inbox, click **confirm subscription** link to complete the subscription.

Now, your Amazon SNS topic is ready to be used for sending alerts and integration with other AWS services.

23. Configuring Centralized Log Management

Using CloudWatch Log Group

Logs are very handy and useful when it comes to troubleshooting any system and application issue. In a cloud environment, it may be very difficult to

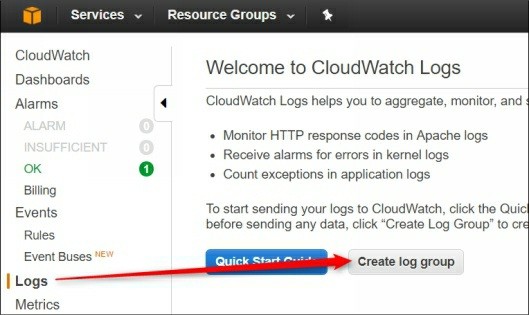
login and check logs for individuals systems. The best way to centralize all the EC2 instance systems’ logs is CloudWatch Log group using CloudWatch Log Agent.

You can use the CloudWatch Logs agent installer for an existing EC2 instance to install and configure the CloudWatch Logs agent. Once the installation is completed, you need to specify the log files and CloudWatch log group name where the log will be stored. Using CloudWatch log agent, the logs are automatically sent from the instance to the log stream you have created while installing the agent.

**Creating a CloudWatch Log Group**

Before you can use CloudWatch Log Agent to centralize the logs, you need to create a CloudWatch Group. For this, you need to perform the following steps:

1. In the left pane, click **Logs** and then click **Create log** group as shown in the following figure.



2. On the **Create Log Group** name dialog box, type the name of log group and click **Create log group**.



3. Open the **CloudWatch console** and select the region where you want to create the log group.

**Granting Permission to EC2 Instance to Use CloudWatch**

**Logs**

Here, you have two options for assigning permission to an EC2 instance to use CloudWatch Group.

1. Using an IAM role and using AWS CLI credentials using access keys. If you have a huge number of EC2 instances to use CloudWatch logs, create and use an IAM role.

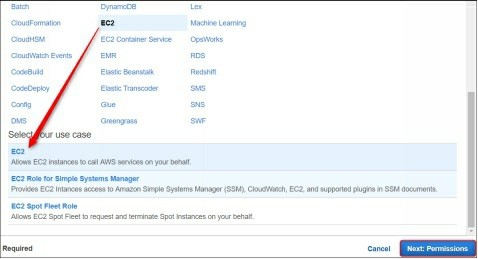
2. If you have limited no. of EC2 instances, you can configure access keys in CloudWatch agent configuration file.

3. To create an IAM role for CloudWatch, open the **IAM console** and click **Roles** in the left pane.



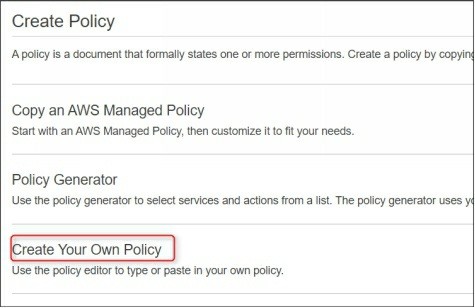
4. On the **Select role type** page, click **AWS service** and then select

**EC2** as shown in the following figure.



5. On the **Permission**s page, click **Create policy**. A new tab “Create policy” tab will be opened.

6. Click **Select** in next to **Create Your Custom Policy** option.



7. On the **Review policy** page, specify the name and description of the policy.

8. In the Policy document t text box, type the below policy carefully and click **Validate policy** to validate it.

*{*

*"Version": "2012-10-17", "Statement": [*

*{*

*"Effect": "Allow", "Action": [*

*"logs:CreateLogGroup", "logs:CreateLogStream", "logs:PutLogEvents", "logs:DescribeLogStreams"*

*],*

*"Resource": [ "arn:aws:logs:\*:\*:\*"*

*]*

*}*

*]*

*}*

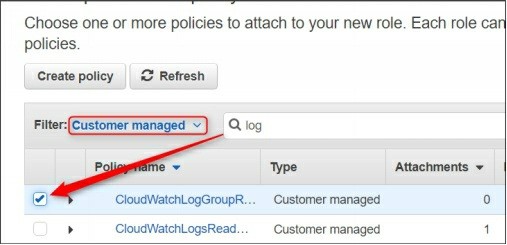
9. Finally, click **Create Policy** to finish the wizard.

**Attaching policy to IAM Role**

Once the policy is created, you need to attach this policy to the desired role. For this, perform the following steps:

1. Switch back to the previous tab (IAM role page).

2. Select **Filter** as **Customer managed**, search the created policy and select it as shown in the following figure.



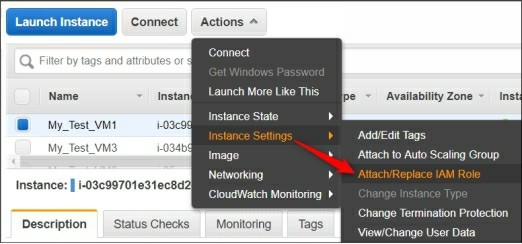
3. On the next page, specify the role name and description and finish the wizard.

**Assigning Role to an EC2 Instance**

To assign IAM role to an EC2 instance, perform the following steps:

1. Select the instance, click **Actions** and navigate to **Attach/Replace**

**IAM Role** as shown in the following figure.



2. On the **Attach/Replace IAM Role** page, select the role you have created earlier and click **Apply** to complete the wizard.



**Installing AWS CloudWatch Log Agent**

Now connect to your EC2 instance and install the AWS CloudWatch Log

Agent. The process of CloudWatch Log agent installation differs based on

the Linux variant and platform. The below process assume the Ubuntu Linux instance.

1. On the terminal, execute the following command to install the

CloudWatch log agent installation script on Ubuntu machine.

***$curl https://s3.amazonaws.com//aws- cloudwatch/downloads/latest/awslogs-agent-setup.py -O***

2. Execute the following command to configure the AWS CloudWatch log agent.

***$sudo python3 ./awslogs-agent-setup.py --region <region-name>***



3. Here, you will we asked to specify various settings. Let’s have a look each at the following:

*AWS Access Key ID [None]:* ***<Leave it blank>***

*AWS Secret Access Key [None]:* ***<Leave it blank>***

*Default region name [ap-southeast-2]:* ***<Your region where you have created Log Group>***

*Default output format [None]:* ***<Leave it None>***

*Path of log file to upload [/var/log/syslog]:* ***<Log’s path that will flow to***

***CloudWatch>***

*Destination Log Group name [/var/log/syslog]:* ***<You log group name>***

*Choose Log Stream name:* ***<Name of log stream inside log group name>***

*Choose Log Stream name:* ***<Refer the following figure to understand this>***

*Choose Log Event timestamp format****: <Refer the following figure to understand this>***

*Choose initial position of upload:* ***<Specify whether you want include***

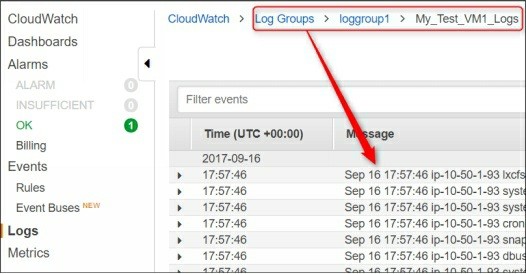
***logs since*** *the beginning or from right now>*



**Verifying Logs in CloudWatch Log Group**

Now switch back to the CloudWatch console and navigate the loggroup1

CloudWatch log group. You should be able to see all the logs of EC2 instance’s log file that you have selected while configuring the agent. In our case, it is **/var/log/syslogs**.



In the same way, you can redirect multiple log files in a single Log group using the different log stream names for easy identification.

25. Schedule Auto, Start, Stop, and Reboot EC2

Instances

When an instance is not in use, you can stop it to reduce unnecessary charges. Typically, every organization usage Development and Quality Audit (Dev

and QA) environments for the testing and verification purpose before going to live with any code and changes. These DEV and QA instances remain idle in the nights and weekends. This can help you to save the huge amount if you are running hundreds of Dev, QA, and Productions servers for an organization.

You can either stop your EC2 instances manually or you can set a schedule to

Auto Start Stop EC2 instance. If you have a large number of running instances the manual method cannot be possible or might be too irritating. However, you can schedule auto start-stop EC2 instances at regular intervals using Lambda functions.

In this article, we are going to explain a step by step guide how to start and stop your EC2 instance at a specific time, nights, and/or weekends.

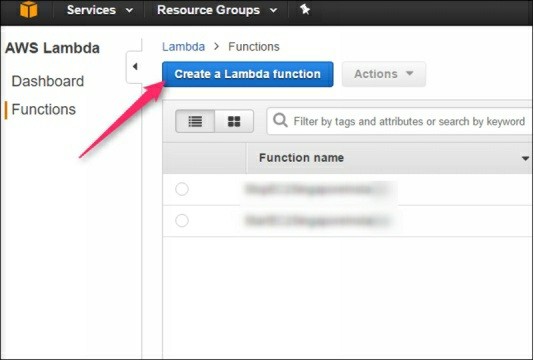
To Schedule Auto Start-Stop EC2 Instance, you need to perform the following tasks.

**Create an Auto Start EC2 Instance Lambda Function**

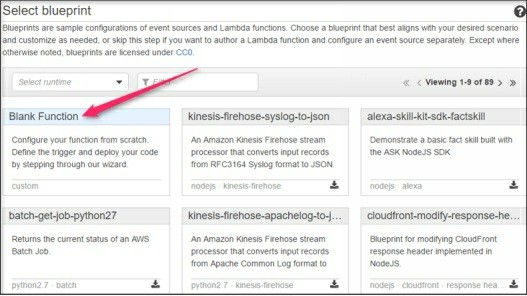
AWS Lambda is a powerful server-less service. However, Lambda is itself a wide technology and we will cover the basics of Lambda in a separate section. As of now, please follow the steps given below.

To create a Lambda function you need to perform the following steps:

1. Open the **AWS Lambda Console** and click **Create a Lambda Function** as shown in the following figure to create a Lambda function.



1. On the **Select Blueprint** page, click on **Blank Function** to choose it as shown in the following figure.



2. On the **Configure Triggers** page, click **Next** to proceed.

3. On the **Configure Function** page, set the following values: Name: **AutoStartEC2Insatnce**

Description: **Auto Start EC2 Instance**

Runtime: **Python 2.7**.

4. On the C**ode entry type** area, type the following script carefully.

***import boto3***

***# Specify the region where your insatnces are running. For example***

***'ap-southeast-1'***

***region = 'ap-southeast-1'***

***# Specify the insatnce IDs that you want to start at specific time. For example, ['i-abcd01234567', 'i-efgh01234567']***

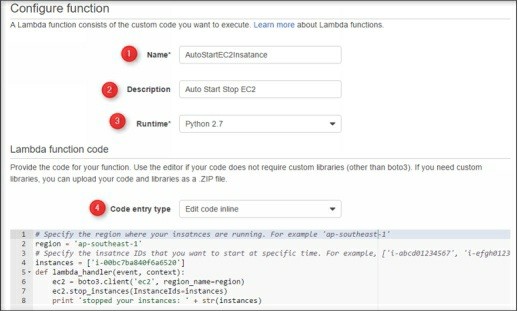
***instances = ['i-00bc7ba840f6a6520']***

***def lambda\_handler(event, context):***

***ec2 = boto3.client('ec2', region\_name=region) ec2.start\_instances(InstanceIds=instances) print 'started your instances: ' + str(instances)***

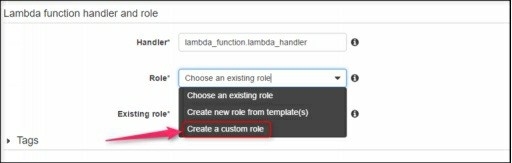
*Note: Replace your instance ID and region in the above python code (script) appropriately.*

5. The **Configure function** page is shown in the following figure.



6. On the same page, scroll down to

the Lambda function and handler section and select **Create a custom role** as shown in the following figure.



7. The IAM management console will be opened in a new tab, type a

Role name such **AutoStartStopEC2Role**.

*Note: The role should have permissions to create logs in AWS Cloud Watch and Start and Stop the EC2 instance. For this, you need to set the appropriate permissions. For this, keep following the below steps*

8. Click **View policy document** and then click **Edit** to edit it.

9. Remove the existing script text and type the following script as-is in the edit policy box.

***{***

***"Version": "2012-10-17", "Statement": [***

***{***

***"Effect": "Allow", "Action": [ "logs:CreateLogGroup", "logs:CreateLogStream", "logs:PutLogEvents"***

***],***

***"Resource": "arn:aws:logs:\*:\*:\*"***

***},***

***{***

***"Effect": "Allow", "Action": [ "ec2:Start\*", "ec2:Stop\*"***

***],***

***"Resource": "\*"***

***}***

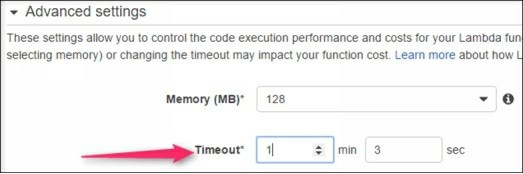
***]***

***}***

10. Click **Save** to save the changes. Click **Allow** and return to the

**Lambda Function** console.

11. On the same page, scroll down to the **Advanced Settings** section and set the **Timeout value** more than 1 minutes.



12. Finally, click **Next** to proceed. On the **Review** page, click **Create**

**Function** to complete the wizard.

13. To test that your function works properly, make sure that the instance you mentioned while creating the Lambda function is stopped.

14. Click on the **Test** tab and then click **Save and Test**, if everything goes fine, you will see the script execution result as succeeded.

*Note: If you get any error, please check the Lambda Function script code.*

15. To verify that instance is started, go to **EC2 running instance list**

and check the status of the instance you mentioned in the script.

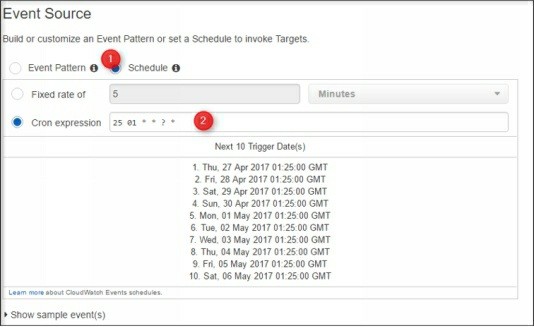
**Creating an Auto Start-Stop Event Schedule Rule**

Till now, you have created the Lambda function and IAM role, however, you have still not defined the time when this instance should start. For this, you need to create an Event scheduler and perform the following steps.

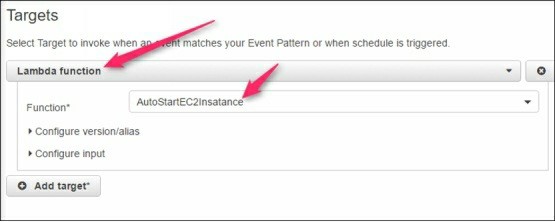
1. Open the **CloudWatch** console, click **Rules** in the left pane.

2. On the **Create Rule** page, select the **Schedule** button.

3. In the **Cron expression** box, set the desired time when you want to start your instances. In our case, we have set it to start at 01:25 GMT at daily.



4. In the right pane, select the **Lambda Function as Targets** and then select the Lambda function name you have created previously.



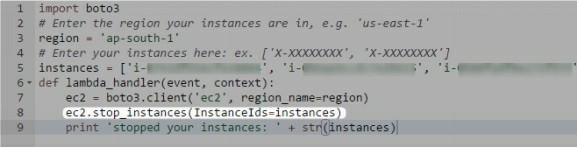
5. Click **Configure details** to proceed. On the **Configure details**

page, specify the rule name, description, and complete wizard.

**Test and Validate Auto Start-Stop EC2 Schedule**

Now you have done all the steps. Just wait for the time you mentioned in the schedule expression and verify that your instance starts automatically.

Using the similar process you can also schedule auto stop event for specific instances at a specific time. The only difference is that you need to use the following Lambda Function script as shown in the following figure. Just change the highlighted part, region, and EC2 instance ID as shown in the following figure.



In addition, you also do not need to create the custom IAM role as it is a one- time activity.

26. Creating and Recovering EC2 Instance Using

Snapshots

Snapshots are really helpful in the case of system or service failure and even in the case of data loss. Snapshots are actually an entire point-in-time copy of a volume. When you create a snapshot of a volume, all the settings, data of that point in that volume is stored in the S3 storage. Later you can create a volume using this snapshot.

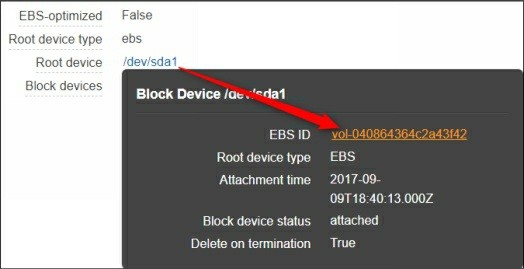
After creating volume from the snapshot you can attach this volume to an EC2 instance and you will get all the data, settings as it was at the time of taking the snapshot.

Here we will show you how to take a snapshot of an EC2 instance, create a volume using snapshot, and recover instance and its data using that snapshot.

**Creating Snapshot of Attached Volume**

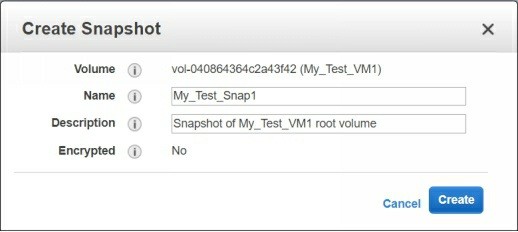
Perform the following steps to create a snapshot of a volume.

1. Select the instance of which you want to take the snapshot. Scroll- down the description tab and click the volume link such as /dev/sda1 of which you want to take the snapshot.

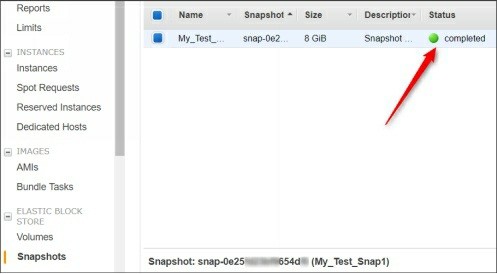


2. In the **Volume** options, click **Actions** and select **Create Snapshot**.

3. On the **Create Snapshot** window, specify the name and description and then click **Create**.

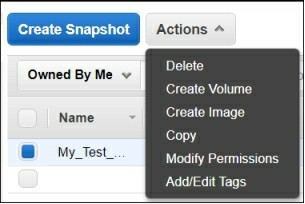


4. After a few minutes, the snapshot will be created and available in the snapshots list as shown in the following figure.



5. While the snapshot is selected, you can perform the following actions as shown in the following figure.

Let’s have a brief look at each of them.



**Delete**: Allows you to delete the selected snapshot. **Create Volume**: Allows you to create an EBS volume. **Create Image**: Allows you to create an AMI image.

**Copy**: Allows you to copy the selected snapshot from one region to another region.

**Modify Permissions**: Allows you to change the snapshot’s

privacy from private to public and vice versa. Additionally, you can share this snapshot with another AWS account holder. **Add/Edit Tags**: Allows you to add or edit the tags.

**Creating Volume of EC2 Snapshot**

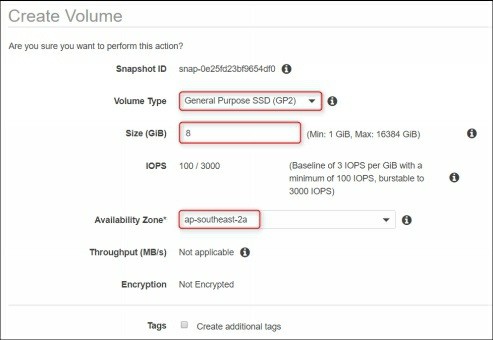
Now, you can create a volume using this snapshot. Later you can attach this volume to another instance and can fix the issues or copy/move its data.

Since we want to recover the instance’s data in this demo, so we will Create

Volume using this snapshot. For this, perform the following steps:

1. Select **Create Volume** option from the **Actions** menu.

2. On the **Create Volume** page, select the volume type, volume size, and the availability zone as shown in the following figure.

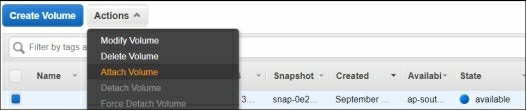


3. Finally, click **Create** button to finish the wizard. The volume will be created after a few minutes.

**Attaching Volume to an Instance**

To attach a volume to an instance, you need to perform the following steps:

1. Select the volume, click **Actions** and select **Attach Volume** to attach it to another instance.



2. In the **Attach Volume** window, type the instance name to which you want to attach this volume.

3. In the **device** box, notice the device name such as /dev/sdf.

*Note: If you want to attach this volume as the root volume to the selected instance, detach the current attached root volume from that instance and set the device name as /dev/sda1.*



4. Finally, click **Attach** to attach the volume.

*Note: The instance must be in the same availability zone in which the volume has been created.*

**Mounting the Attached Volume**

To mount and use the attached volume, you need to perform the following steps:

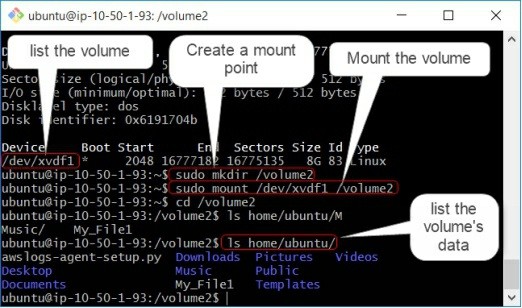
1. Start the instance to which you have attached this volume and connect it.

2. Execute the **sudo fdisk -l** command to view the device later.

3. Create a mount point and mount the above-listed volume.

4. Now all the data of the mounted volume will be available for you.

Perform the actions whatever you want to do. Refer the following figure.



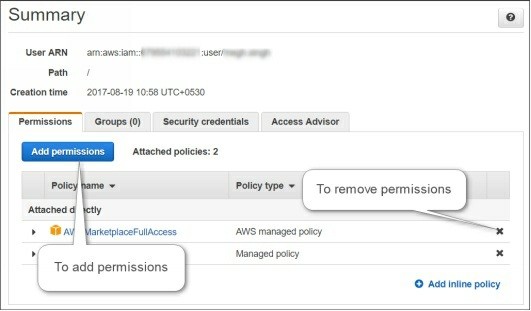
That’s all you need to create a snapshot, create a volume using snapshot, and recover the data using the EC2 snapshot.

27. Working with IAM User Properties

We have already discussed the basics of IAM users and its usage and role. Here, we will explain some of the major properties of an IAM users that will help you to control AWS resources more accurately and appropriately. There are various IAM user properties options that you should know. The following four tabs are available for an IAM user:

**Permissions**

This tab shows what permissions have been granted to the selected IAM user. You can add or remove the permissions using this tab as shown in the following figure.



**Groups**

This tab allows you to add the selected user to a specific group for better control and management.

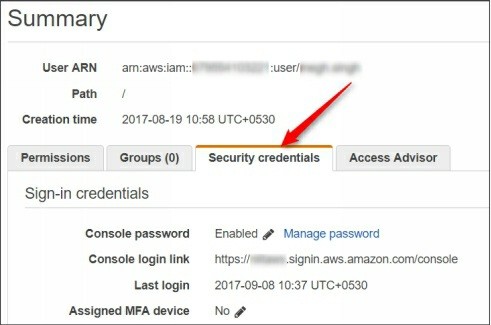
**Security Credentials**

This is a most important tab that holds the key features for an IAM user. Here you can do the following actions:

Enable or disable console login for the selected user. Change the console password for the selected user. Enable or disable Multi-factor authentication

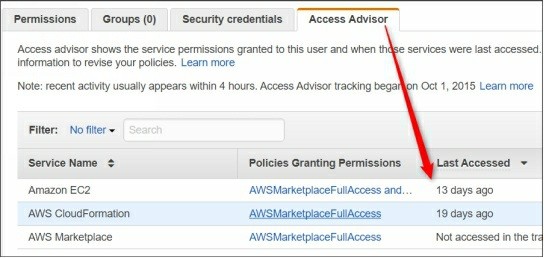
Create and remove CLI-based access keys

Manage CodeCommit credentials etc.



**Access Advisor**

This tab allows you to track the activities performed by this user. However, if you want more details and tracking for an IAM user, you should use the CloudTrail that is covered in later sections.



28. Creating and Using an IAM Role

You can use IAM roles to delegate access and permissions to your AWS resources. The usage of IAM roles in AWS is very wide. There are various scenarios and requirements where IAM roles can be used. Even you can

create a role that allows an IAM user to manage other AWS account managed by your organization. Various CloudForm template and services required

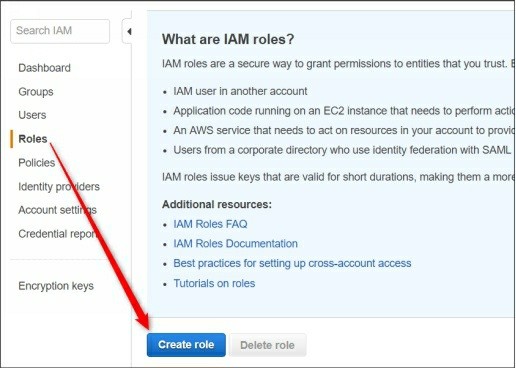
roles to be created in order to use them.

Here, we will show you how to create and use an IAM role to manage AWS

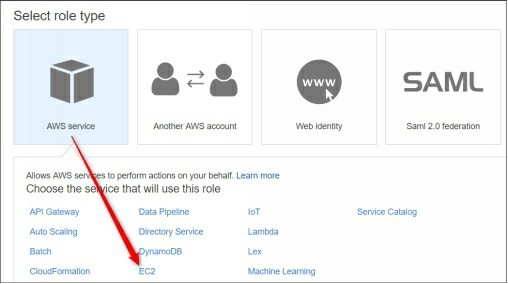
resources.

The following steps need to be followed in order to create an IAM role:

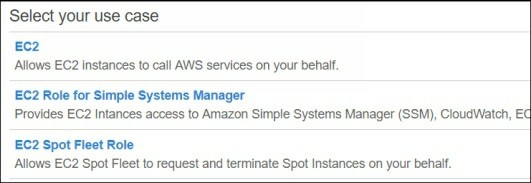
1. In the **IAM console**, click **Roles** and then click **Create role** as shown in the following figure.



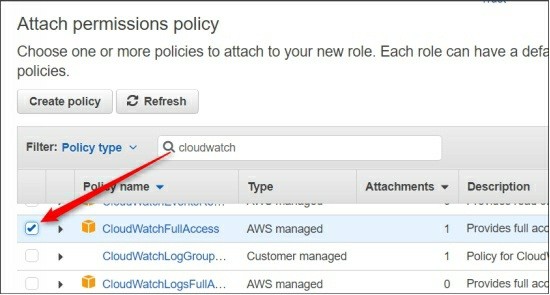
2. On the **Select role type** page, select the type of role, for example, EC2 under the AWS service as shown in the following figure.



3. Depending on the selected role type, you may be further asked to select your use case as shown in the following figure. For this demo case, we will select EC2.



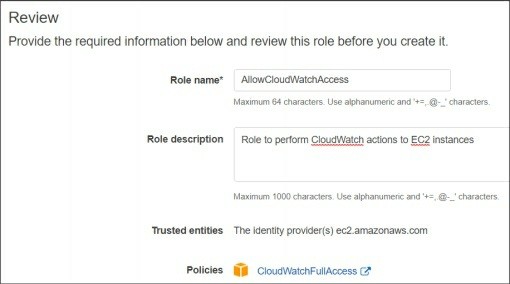
4. On the **Attach permission policy** page search and select the policy you want to attach to this role. For example, if you want that this role should be allowed to perform CloudWatch related jobs, select the CloudWatchFullAccess permission as shown in the following figure.



*Note: There are hundreds of pre-configured policies for the various AWS resource usage and control. However, you may not*

*find the appropriate policy as per your exact custom need. Here comes the custom policy as a handy feature, but you have to be familiar with the policy document and how to create and configure new custom policy.*

5. In the **Review** page, specify the **Role and Description** and finish the wizard.

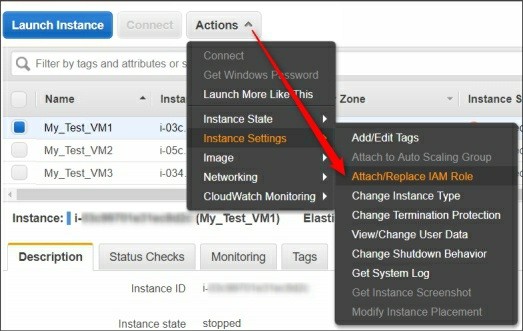


**Attaching IAM role to an EC instance**

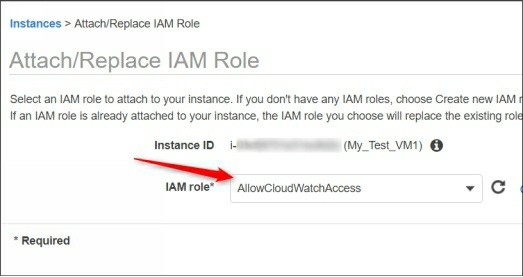
Once the EC2 Role is created, you can attach it to the desired EC2 instance. To do this, perform the following steps:

1. Select an **EC2 instance** and navigate to **Attach/Replace IAM Role**

as shown in the following figure.



2. In the **Attach/Replace IAM Role** page, select the role you have created earlier and click **Apply**.



3. In the **Description** tab, you will see that the role is applied to the selected instance. This instance now can perform the CloudWatch related function without the need of AWS CLI-based access keys at all.

29. Configuring Password Policies for IAM Users Security is always a major area where you should always focus seriously. Keeping strong password for IAM users will only make malicious users to crack the password harder and harder. As per the AWS official document, you should perform the following actions for a strong password policy

1. The minimum password length should be set to 8 or greater.

2. The password should require specific character types, including uppercase letters, lowercase letters, numbers, and non-alphanumeric characters.

3. All IAM users should be able to change their own passwords.

4. All IAM users should change their password after a specified period of time.

5. All IAM users should be prevented from reusing previous passwords.

**Changing IAM Password Policy Settings**

Changing IAM password policy is very simple, just navigate to account settings in the IAM console. The following figure shows the sample of a decent IAM password policy.



After setting the appropriate values for your IAM password policy just click

Apply make it effective.

30. Installing and configuring AWS CLI

The AWS CLI is an open source tool built on top of the AWS SDK for Python (Boto) that provides commands for interacting with AWS services. You can perform almost all the AWS tasks using the AWS CLI that you performed using AWS console. AWS CLI tool is available for both Linux as well for Windows systems. Here we will explain how to manage AWS Resources using AWS CLI.

You can download and install AWS CLI for Windows XP or later systems. Once you installed it on Windows machine, you need to setup credentials method to authorize your Windows machine to perform AWS actions.

**Download AWS CLI for Windows System**

Depending on the architecture you are using for your Windows machine, click the appropriate download link and download the AWS CLI MSI installer for your Windows system.

[Download AWS CLI Installer for Windows 32-bit systems](https://s3.amazonaws.com/aws-cli/AWSCLI32.msi)

[Download AWS CLI Installer for Windows 64-bit systems](https://s3.amazonaws.com/aws-cli/AWSCLI64.msi)

**Installing AWS CLI on Windows System**

The installation process of AWS CLI on Windows is pretty simple and straightforward, just click the executable MSI installer file and follow the on-screen instructions to proceed.

Just finish the installation using the default selections.



**Installing AWS CLI on Linux system**

Depending on the Linux variants you are using the installation commands for installing AWS CLI in Linux system may vary. But if you are familiar with any of the Linux variant such as Ubuntu, you should not face any major issue to install it on other variants such as Red Hat Linux.

For example, to install AWS CLI on an Ubuntu machine execute the following command:

***$sudo apt-get install awscli -y***

This is the simplest way to install the AWS CLI tool. But for any reason, if the above method does not work for you, visit the following link and follow the steps to install it with pip method.

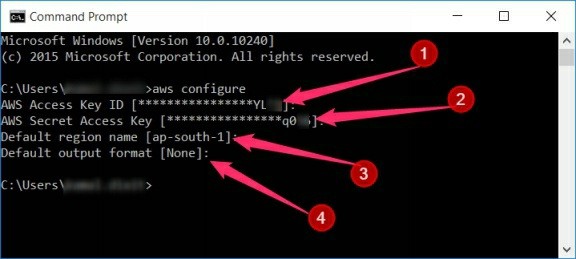
[Installing AWS CLI on Linux with Pip](http://docs.aws.amazon.com/cli/latest/userguide/awscli-install-linux.html)

**Setup Credentials for AWS CLI**

Before you could use the AWS CLI interface to manage AWS resources, you must set up the proper credentials for AWS CLI interface. If you are using your local system to perform AWS actions, you must have AWS Access Key ID and AWS Secret Keys with appropriate permissions on the resources you wish to manage.

For example, if you want to manage S3 buckets using AWS CLI from your local system (Windows or Linux), you first need to create an AWS Access Keys under the Security Credentials tab of IAM user. These keys need to be configured on your local machine using the **aws configure** command as shown in the following figure.

Setting up AWS CLI credentials method will remain same for both Linux and



Windows systems.

Once the credentials are set, you can perform the actions for which your IAM

user is allowed.

31. Configuring OpenVPN Server to Securely

Access Instances

We assume that you are already familiar with the usage and importance of VPN server and its features. In order to access EC2 instance from your local system, you have to enable SSH port for EC2 instance publically.

*Note: This topic is not a part of AWS Certified Solution Architect –Associate Exam. This is just for the best practices while using AWS EC2 instance in the real world.*

Allowing SSH or RDP over the public network might be a little bit risky action in term of security. One more issue you may face that if you have public subnet and private subnet based VPC network, you cannot access SSH or RDP of your private subnet’s EC2 instances directly from your local machine. Here comes OpenVPN as a highly secure and handy mechanism for connecting and managing EC2 instances securely.

The most popular and easy to use VPN solution for AWS cloud is OpenVPN

server solution.

We will use the following network for this lab exercise:

VPC: My\_Test\_VPC Subnets:

My\_Public\_Subnet1- Belongs to availability zone1. My\_Private\_Subnet1 – Belongs to availability zone2.

Instances

My\_Test\_VM1 - Belongs to public subnet1 and have elastic IP

attached.

My\_Test\_VM2 – Belongs to private subnet1 and do not have public IP.

Internet Gateway: My\_Test\_IGW1 - Attached to public subnet1

**Launching Pre-configured AMI from AWS Market Place** Here we will use a pre-configured OpenVPN AMI from the AWS Marketplace and will launch it in My\_Public\_Subnet1.

For this, you need to perform the following steps.

1. In the EC2 dashboard, click **Launch Instance** and select **AWS Marketplace** in the left pane.

2. In the search box, type **OpenVPN** and select **OpenVPN Access**

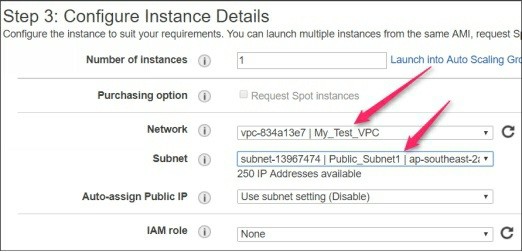
**Server** as shown in the following figure.



3. Review the instance type and pricing fee for this AMI and then click **Continue** to proceed. You will get two VPN users for free to use OpenVPN Access Server.

4. On the **Choose Instance Type** page, select the **T2 Micro** option.

5. On the **Configure Instance Details** page, select the **VPC** and your public subnet as shown in the following figure.



6. On the **Storage** page select the default storage size.

7. On the **Add Tags** page, assign the name as My\_Test\_VPN1 as the name of the instance.

8. On the **Configure Security Group** page, specify the name of description of the security group.

9. Finally, click **Review and** L**aunch** button to launch it.

10. Specify My\_Test\_VPN1 as the key pair name and click Launch. Wait

for few minutes to complete the launch process.

**Assigning Elastic IP**

Depending on your public subnet settings, OpenVPN server may or may not get public IP. To assign an Elastic IP for your OpenVPN server, just generate an Elastic IP and associate it with your VPN server (My\_Test\_VPN1).

**Configuring OpenVPN Initial Settings**

The following steps need to be performed to configure OpenVPN server for the first time:

1. Once the instance is ready and the EIP is attached, connect it using the following command

***$ssh -i <key-name> openvpnas@<public-ip-address>***

You will be prompted to accept the license agreement and you know what you have to do here.

Since this is your primary Access Server node, press **enter** to accept the default setting.

Press **enter** to accept the default setting for network interface option.

Press **enter** to accept default port for VPN Access server. Press **enter** to accept the default Web Admin UI port for your server.

Keep reviewing the prompts asked by the wizard and choose the appropriate option. For the lab exercise, accepting and keeping the default options are enough.

2. One the wizard is finished you will get the URLs that will be used to manager OpenVPN server.



*Note: If you selected any wrong option and want to reconfigure OpenVPN options, you can execute the* ***sudo open-init --ec2*** *to re-launch the OpenVPN initial configuration prompts.*

3. Execute the following command to set the password for your Web UI OpenVPN user.

***$sudo passwd OpenVPN***

4. Now, open the browser and login to your OpenVPN Web Admin UI

with username as openvpn. https://<public-ip-address>/admin

5. You will be asked to accept the license agreement at the first time login.

6. In the Web Admin Console, you can configure various options for

OpenVPN Access server.

The most common and initial configuration options for OpenVPN Access server are:

**Activating License.**

By default you will get two users free license for OpenVPN server, if the number of users are more than two, you can purchase the license. To activate the license, click **License** in the left pane, type license key and click **Add A New License Key** to activate as shown in the following figure.



**Creating OpenVPN user accounts.**

By default one user “OpenVPN” is already created when you configure OpenVPN. This user has admin permission and should be used to perform Web Admin UI action. For the other users who want to connect your AWS infra using OpenVPN server, a separate user should be created.

For this, just click **User Permissions** under **the User Management** section. Type a username, set the desired password, and click **Save Settings** to create VPN user as shown in the following figure.

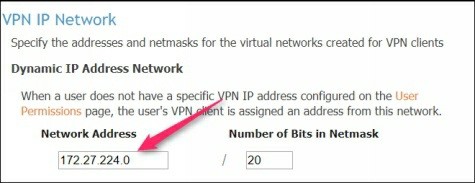


**Configuring VPN settings.**

Now your OpenVPN server is ready to be used. However, you may also want to configure few other OpenVPN server settings such as what IP address should get a client when connected. What private subnets are allowed to connect through OpenVPN server? These settings can be configured in the VPN Settings section.

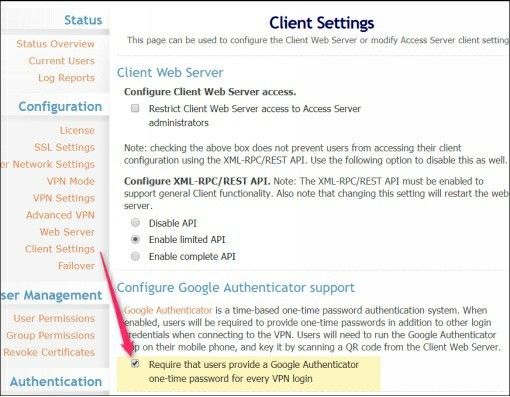
You can customize the dynamic IP address range for a client under the VPN IP Network section as shown in the following figure.

You can specify the private subnet under the Routing section as shown in the following figure.



**Enabling Google Authenticator for VPN Clients**

If you wish to enable two-factor authentication for your VPN users, you can enable **Google Authenticator** option under the **Client Settings** as shown in the following figure.



That’s all you need to setup your OpenVPN server with basic settings. Now

you can connect to this OpenVPN server using the OpenVPN Client tool and can access the private EC2 instances directly.

[Click here more detail about OpenVPN Access Server](https://docs.openvpn.net/getting-started/)

32. Connecting OpenVPN Server

Since we have configured OpenVPN server in the previous lab exercise, here we will explain the process how to connect OpenVPN server using OpenVPN client app.

**Two-Factor Authentication**

If the two-factor authentication is enabled for OpenVPN access server then you should consider the following else you may skip this.

Before you can connect to VPN server, you need to download two steps authentication application such as Google Authenticator on your mobile. Depending on your mobile platform, you can download it for [iPhone store,](https://itunes.apple.com/in/app/google-authenticator/id388497605?mt=8) [Android store,](https://play.google.com/store/apps/details?id=com.google.android.apps.authenticator2&hl=en) and also for [Blackberry mobile.](https://appworld.blackberry.com/webstore/content/29401059/?countrycode=IN&lang=en) If you are using Windows phone, you can use [Authenticator+ app](https://www.microsoft.com/en-in/store/p/authenticator/9nblggh08h54) available in the Windows store. Alternatively, you can also use two factor google chrome extension for Google Chrome browser. But it is not as secure as having mobile based two- factor authentication.

Once you have installed the Two steps Authentication app, you need to perform following steps:

**Setting up Two Factor Configuration**

1. Use the below URL to open the VPN connection

https://<openvpn-public-ip>

2. On the **OpenVPN** window, type your VPN username and password.

3. Select the **Login** and click the **Go** button.



4. On the next window, you will see the link from where you can download the OpenVPN client app.

5. If the two-factor authentication is enabled, scroll-down and scan QR code using **Google Authenticator**. Alternatively, you can also enter secret key manually if you have a problem with scanning QR code.

6. Click **I Scanned the QR code** to proceed.

7. Next, you need to download the OpenVPN client tool depending on the platform Windows/Linux/MAC you are using. For the future reference, you also need to **download the connection profile file** by clicking **yourself (user-locked profile)** link.



Setting Google authentication app, downloading client profile, installing

OpenVPN client connect tool all these are a one-time activity.

**Connecting Using OpenVPN Client Tool**

We assume you have downloaded and installed OpenVPN client on your

machine. Depending on the OS platform, you may need to adjust some settings.

For Linux Platform

For Linux platform (Ubuntu), you need to execute the following commands to connect OpenVPN.

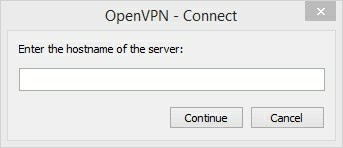
***$sudo apt-get install openvpn***

***$sudo openvpn --config <path of user connection ovpn file>***

For Windows OS

For the Windows-based system, you need to perform the following steps:

1. Open the OpenVPN client application, type the public IP address of your OpenVPN Access Server and then click **Connect**.



2. If your system is behind a proxy server, you need to set the Proxy address before to proceed next. For this, on the **OpenVPN Client** app, click **Options**, select **HTTP Proxy** and then set the appropriate proxy address.

3. On the **Connect** window, type your username and password and proceed to next.



4. If the **Google Authenticator** feature is enabled then you will need to type the authenticator dynamic code before to click **Connect**.

5. Now you should be connected to your OpenVPN Access server and should get a private IP address for your local system.

Now you can connect your internal private EC2 instance from your local system. But before this make sure that your instances’ security groups are configured to allow incoming traffic from the OpenVPN server network.

33. Configuring Linux Bastion Server for Securely

Access SSH of Private Instances

Bastion server is nothing but just a Linux secure acting as jump server and allows you to connect your internal private servers securely. The Bastion server is typically configured in the Private subnet and only accessible through OpenVPN connection.

We will use the following scenario to demonstrate how to use Bastion server to connect internal private EC2 instances.

In the below figure you can see:

There is one VPC with two availability zones.

The VPC consists of two subnets: one is public and another is the private subnet.

The VPC has an OpenVPN server configured and running, which is placed on the public subnet.

The VPC has one instance that belongs to the internal private subnet and does not have any public IP associated.

We will launch the bastion server on the private subnet and will connect the internal private EC2 instance using this bastion.



We assume that you have already configured and running the OpenVPN

access server.

**Launching a New Bastion Server**

Now launch a new Ubuntu Linux instance with following settings:

**Amazon Machine Image**: Ubuntu 16.04

**Instance type**: T2 Micro **VPC**: My\_Test\_VPC **Subnet**: Private Subnet1

**Storage**: Default size (8 GB)

**Instance Name**: My\_Test\_Bastion1

**Security Group**: My\_Test\_Bastion\_SG1

**Key pair Name**: My\_Test\_Bastion1

**Configuring Security Groups**

Once the bastion server is launched, you need to update the security group rules as per the network diagram shown above.

The following points explain the security group requirements in details:

1. My\_Test\_VPN\_SG1 (security group attached to OpenVPN server) The SSH port of this security group should only be allowed for your IP

location. For this, edit the security group inbound rule and select My IP

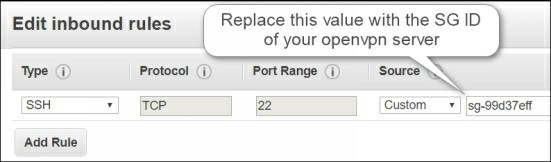
for SSH as shown in the following figure.

2. My\_Test\_Bastion\_SG1 (Security group attached to bastion server) The Bastion server’s SSH should be allowed for OpenVPN network. For



this, select the bastion server’s security group and click Edit inbound rule.

3. In the **Source** drop-down, select **Custom** and specify the security group ID of OpenVPN server’s attached security group as shown in the following figure.



4. My\_Test\_VM\_SG (Security group attached to private instance).

Now, your internal private instances SSH should only be allowed for Bastion server. Anyone who wants to connect your internal instances should come through the bastion server. For this, you need to allow SSH of the private instance for My\_Test\_Bastion\_SG1.

5. Edit the inbound rule and paste the bastion server’s SG ID in the source custom value in front of SSH port as shown in the following figure.

Now, if you want to access SSH of your private instance, you need to perform the following process:



1. Connect to your OpenVPN server using the VPN username, password, and optionally google authenticator code (if enabled).

2. Once you are connected to OpenVPN server, you will get a private IP

address.

3. From your local system’s terminal, you can connect now Bastion server.

4. From bastion server’s terminal, you can connect to your private EC2 instances.

**Connecting SSH using Bastion Server**

From your local system’s terminal, execute the following command to enable

SSH agent forwarding server. This will allow you to add SSH keys of

Bastion and Internal EC2 instance using SSH Agent service. Thus

eliminating the manual copying of SSH access keys on individual servers.

***$eval $(ssh-agent)***

***$ssh-add -k <bastion-server-ssh-key>***

***$ssh-add -k <internal-instance-ssh-key>***

***$ss -A user@<bastion-server-private-ip>***

For example, if your bastion server key is My\_Test\_Bastion1.pem, private instance’s key is My\_Test\_VM2, and the bastion server IP address is

10.50.2.147 then you need to execute the commands. (Assuming your instance’s username is ubuntu)

***$eval $(ssh-agent)***

***$ssh-add –k My\_Test\_Bastion1.pem***

***$ssh-add –k My\_Test\_VM2.pem***

***$ssh –A ubuntu@10.50.2.147***

From the bastion server terminal, you can access all the private instances who’s SSH ports are allowed in the inbound rule for Bastion server’s security group.

For example, if your private instance’s username is Ubuntu and IP address is

10.50.2.115, then execute the following command on bastion server’s terminal.

***$ssh ubuntu@10.50.2.115***

You should be connected even without mentioning the .pem key as you already added that with SSH forwarding agent service.

34. Working with S3 Buckets

Amazon Simple Storage Service (S3) is one of the most popular and premier AWS Cloud service. Amazon S3 is an object storage solution designed for storing and retrieving any amount of data from anywhere. Amazon S3 stores data for millions of applications used by market leaders in every industry. S3 buckets are typically folders used to store your data. You can create your own customized folder structure inside an S3 bucket.

We assume that you are already familiar with basics of Amazon S3 storage and its permissions and policy concepts. However, here we would like to highlight few of the basics features of S3 storage.

You can upload a maximum of a 5 GB of a single file to S3 bucket. You can upload any amount of data to S3 storage.

By default, all the S3 buckets are private until unless you make them publically manually.

You can host a static website using the S3 storage and server-less architecture.

**Recommended links**

[Getting Started With Amazon S3](http://docs.aws.amazon.com/AmazonS3/latest/user-guide/what-is-s3.html)

Here, we will show you how to create, rename, and delete S3 buckets and how to upload and download data in S3 buckets using console-based GUI interface as well using the AWS CLI tool.

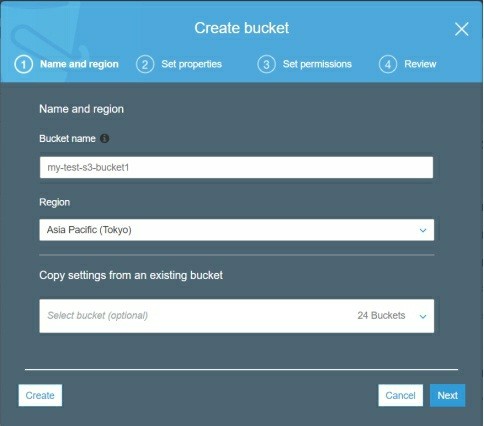
**Creating an S3 Bucket Using the AWS Console**

To create an S3 bucket using AWS console is pretty simple, just follow the following steps:

1. Login to **AWS console** and navigate to the S3 page.

2. Click **Create bucket** to create a new S3 bucket as **Create bucket**

dialog box will be displayed as shown in the following figure.



3. Specify a unique name for your S3 bucket and click **Next**.

4. On the **Set Properties** page, you can define the versioning, tagging, and logging which we will cover in a separate article.

5. On the **Set Permissions** page, you can specify the permissions that you want to assign for this bucket. We will cover permissions in the upcoming sections.

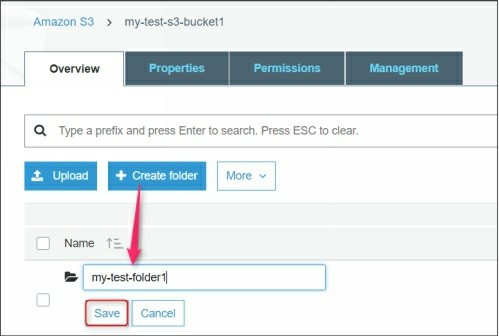
6. Review the options you have selected and finish the wizard to complete the process. Your S3 bucket will be created.

**Creating Folders under S3 Bucket**

Now you can create your folder structure inside the S3 bucket. For this, just perform the following steps:

1. Click the **S3 bucket** and click **Create folder**.

2. Specify the folder name and click **Save** to save it as shown in the following figure.



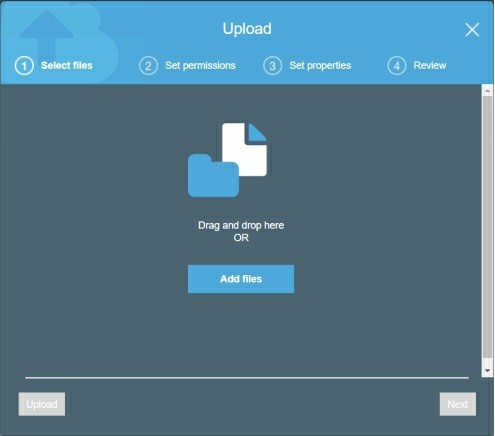
**Upload Content/Files/Data in S3 Bucket**

1. To upload data in a S3 bucket, open the desired S3 bucket and navigate to the folder where you want to upload data.

2. Click **Upload** and drag-n-drop the files you want to upload or click

**Add files** and browse the location.

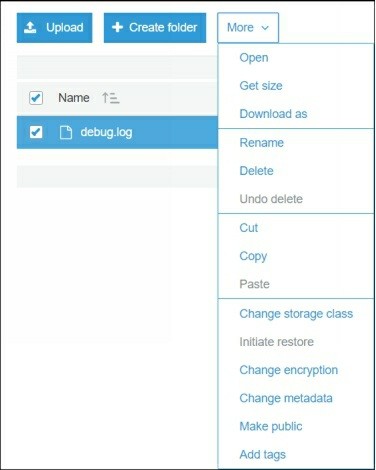
3. Select files you want to upload as shown in the following file.



4. Finally, click **Upload** and your files will we uploaded to the S3 bucket.

5. Select any of the uploaded files and click **More** to view all the available actions that you can perform as shown in the following

figure. Try each one of them to understand their functionalities.



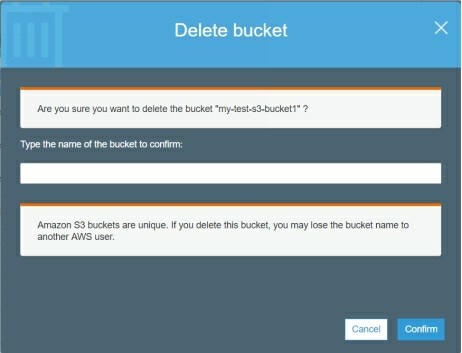
**Deleting S3 Bucket**

Deleting an S3 bucket using the console is pretty simple and straightforward.

1. Just select the S3 bucket you want to delete and click **Delete bucket**

option.

2. On the **Delete bucket** dialog box, type the exact bucket name in the text box and click **Confirm** to delete it as shown in the following figure.



35. Configuring Permissions and policy for S3 buckets

Manage permissions for S3 buckets is a little bit complicated and requires

some study before you could implement proper permissions. But once you become familiar with the various custom-scenario-based S3 bucket policies, you are really on the track of becoming an AWS Cloud expert.

There are various ways to protect your S3 buckets from unauthorized access. Some of the basic security scenarios are:

Restricting specific S3 buckets for specifying IP location. Restricting specific IAM user to access specific S3 buckets. Restricting specific S3 bucket to access from specific domains.

For few another common scenario, please have look at the following article.

[S3 Buckets User Access Policy Examples](http://docs.aws.amazon.com/AmazonS3/latest/dev/example-policies-s3.html)

**Customizing S3 Bucket Policies**

To customize and manage permissions for an S3 bucket, you need to perform the following steps:

1. Navigate the S3 bucket for which you want to customize permissions.

2. Select the Permissions tab to view the assigned permissions for the selected bucket.

3. Permission tab has few other sub tabs such as Access Control List and Bucket policy.

Access Control List Tab

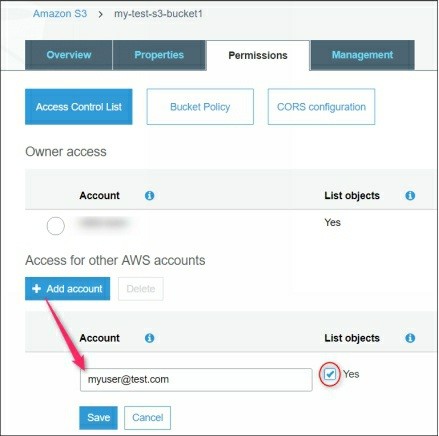
This tab allows you to add or remove IAM user to list and write objects in this bucket.

To add an IAM user to allow list and write objects in this bucket, perform the following steps:

1. Click **Add account** and enter the user’s canonical id or email address.

2. Select the permissions that you want to assign to this user and click

**Save** as shown in the following figure.



Bucket Policy Tab

Here, you can create a custom policy for the selected bucket. The scenarios for the custom policy depends on the organization’s need and may have a wide range of custom requirements. In the next section, we have covered few example scenarios to manage custom S3 bucket policies.

36. Configuring S3 Bucket Policies for Specific

IAM Users

Here we are going to configure the custom S3 bucket policy for the following scenarios:

Allow a specific IAM user such as **my-test-user1** to access a specific

S3 bucket such as my-test-s3-bukcet1.

Create a role that allows accessing my-test-s3-bucket1 S3 bucket to multiple EC2 instances.

**Creating Custom S3 Policy for a Specific IAM User**

For the first scenario, you need to perform the following steps:

1. Create an IAM user named my-test-user1 using the IAM management console. Since we have already covered the process of creating IAM user so we are going to skip it here.

2. Create a custom policy that allows only my-test-s3-bucket1 for my- test-user1.

3. Click **Policies** and then click **Create Policy** in the IAM console.

4. In the **Create Policy** page, select **Create Your Own Policy**.

5. Specify the Policy name and description as per your wish.

6. In the **Policy Document** text area, paste the following policy code and replace the bucket name, if required.

***{***

***"Version": "2012-10-17", "Statement": [***

***{***

***"Effect": "Allow", "Action": [***

***"s3:ListBucket"***

***],***

***"Resource": [***

***"arn:aws:s3:::my-test-s3-bucket1"***

***]***

***},***

***{***

***"Effect": "Allow", "Action": [***

***"s3:PutObject", "s3:GetObject"***

***],***

***"Resource": [***

***"arn:aws:s3:::my-test-s3-bucket1/\*"***

***]***

***}***

***]***

***}***

7. Finally, click **Create Policy** to create it.

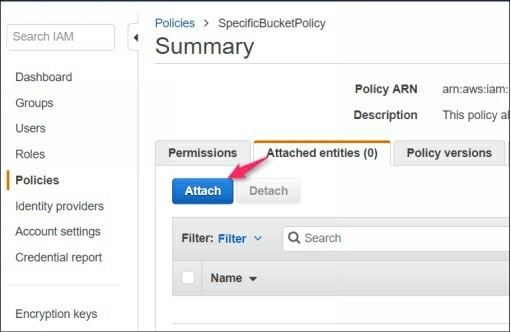
*Note: The above policy will allow (after attaching) to get and put the objects in this S3 bucket.*

**Attaching Policy to IAM Role, Group, or User**

The created policy will not affect anything until unless it is attached to an entity such as IAM user or group. To attach a policy to IAM user, role, or group, you need to perform the following steps:

1. Select the created policy and then go to **Attached entities** tab.

2. Click **Attach** to attach it to an IAM user, role, or group as shown in the following figure.



3. In the **Attach Policy** window, select the IAM user, group, or role to which you want to attach it. For this demo, select my-test-user1 and click Attach Policy.

4. Now you have created a custom S3 bucket policy and attached to the

IAM entity (IAM user).

**Verifying Custom S3 Bucket Policies**

To test and verify whether my-test-user1 is able to view, download, and upload content to my-test-s3-bucket1, login to the AWS console with my- test-user1 and click the below link.

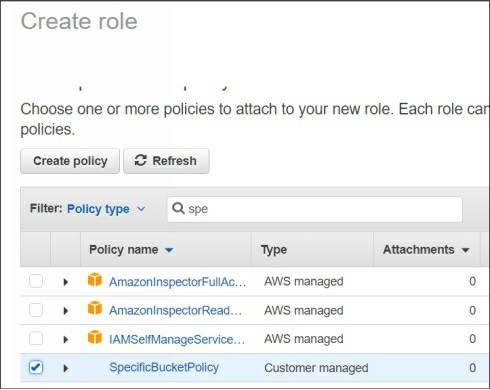
https://s3.console.aws.amazon.com/s3/buckets/<your-bucket-name>

You should be able to view the content of the selected S3 bucket. If you will try to access this bucket with a different IAM user, you will get an Access Denied error.

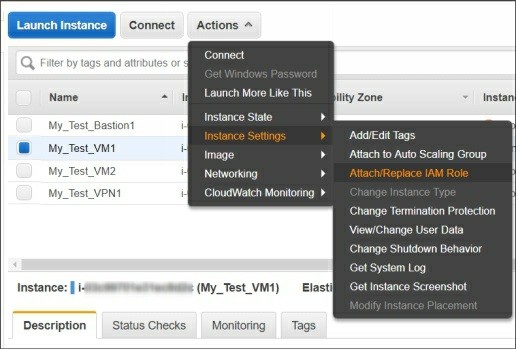
**Verifying S3 Bucket Policy Using AWS CLI Tool**

Here we will test and validate the above-created policy by the following methods:

1. Create an EC2 role and attach the above-created policy (specific- bucket-policy) to this role.

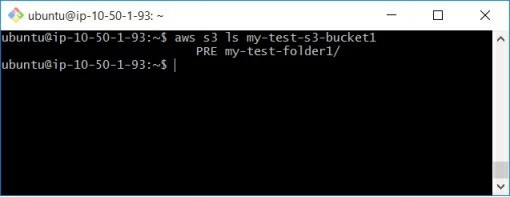


2. Attach this role to one of the test VMs such as My\_Test\_VM1.

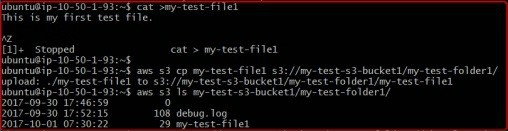


3. Install the AWS CLI tool on the test VM, if not installed already.

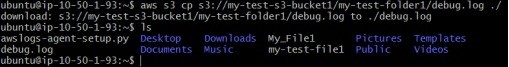
4. List my-test-s3-bucket1 content using the command as shown in the following figure.



5. Create and upload a text file named my-test-file1 to a folder named my-test-folder1 under my-test-s3-bucket1 as shown in the following figure.



6. Download a file named debug.log from s3 bucket location (my-test- folder1 under my-test-s3-bucket1) to the current working directory of local system as shown in the following figure.



Here, we have explained how to allow a specific IAM user, role, and group to access specific one or more S3 buckets using the custom policies.

37. Configuring S3 Bucket Versioning and

Logging.

There are various other configuration options for S3 buckets apart from S3 bucket policies. Here we are going to explain few of the most common options.

**Configuring S3 Bucket Versioning**

If you typically keep changing and modifying files and content stored in the S3 buckets, you can enable S3 versioning to keep multiple versions of modified files.

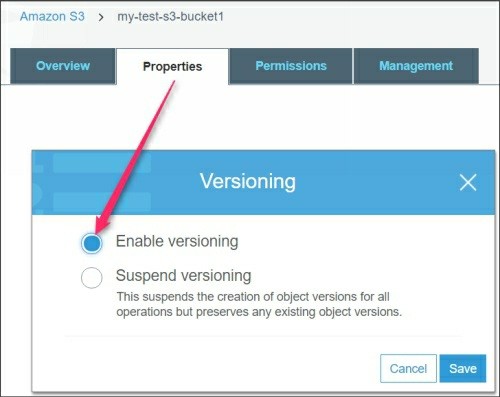
[Click here to know more about the object versioning.](http://docs.aws.amazon.com/AmazonS3/latest/dev/ObjectVersioning.html)

To enable or disable versioning for S3 bucket objects, you need to perform the following steps:

1. Select the desired S3 bucket and select the **Properties** tab.

2. Click the **Versioning feature** box to enable it.

3. Select **Enable versioning** radio button and then click **Save** as shown in the following figure.



**Configuring Logging for S3 Buckets**

Sometimes, it may be very useful to track the activities performed by users for particular S3 buckets. Each access log record provides details about a single access request, such as the requester, bucket name, request time,

request action, response status, and error code, if any. You can track the access activities for S3 buckets if you have enabled the logging feature for S3 buckets.

[S3 Bucket Access Logging](https://docs.aws.amazon.com/AmazonS3/latest/dev/ServerLogs.html)

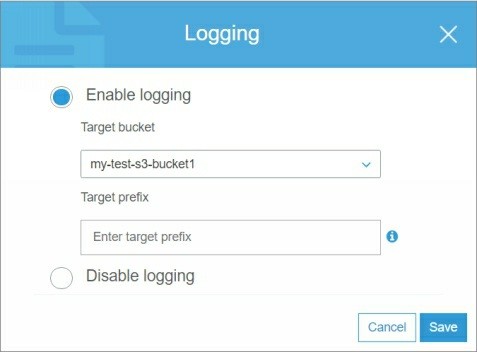
To enable logging for an S3 bucket, you need to perform the following steps:

1. Select the **S3 bucket** and then select **Properties**.

2. Click **Logging** to enable it.

3. On the **Logging** window, select the target bucket and then click **Save**

as shown in the following figure.



38. Configuring S3 Bucket Alerting and

Notifications

In the previous labs, we have done various hands-on labs with the S3 buckets. Here we will explore one more interesting hands-on lab called S3 Bucket Alerting and Notifications.

Sometimes, IT admins may wish to receive alerts whenever there any changes happen with an S3 bucket. These changes may be putting a new object, renaming an existing object, or deleting an object inside an S3 bucket. Before configuring notification for S3 buckets, we assume that you have already created an SNS topic with appropriate mail list of users who will receive the notifications.

[S3 Bucket Notifications](https://docs.aws.amazon.com/AmazonS3/latest/user-guide/enable-event-notifications.html)

To configure notification or events for an S3 bucket, you need to perform the following steps.

1. Select the **Properties** tab of the desired S3 bucket.

2. In the **Advanced Settings** section, click the **Events** box.

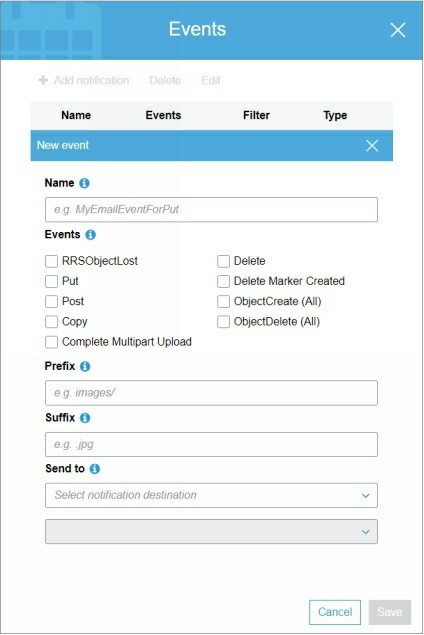
3. Click **Add notification** to configure the notification options. Here you can configure the following options:

**Name**: Name of the notification rule.

**Events**: Type of events for which you want to enable notifications. For example, select **Object Created (All).**

**Prefix**: If the bucket has multiple folders, you can limit the notifications by limiting it to specific prefix as the folder name. **Suffix**: Specify the file type or extensions such as .txt, .exe, .jpg to limit the notifications.

**Send to**: Select the preconfigured SNS topic where the notification will be sent.



4. Finally, click **Save** to save the settings.

Now, whenever changes will happen as per your selected event types such as **Put**, **Copy**, **Delete** etc. a notification will be sent through the selected SNS topic.

39. Configuring S3 Bucket Lifecycle Rule

S3 buckets can store objects and contents as long as you do not delete them manually. But there might be some scenarios when you may want to delete the data of a bucket after specific days. One such example is given as below.

Suppose you are taking the regular backups of your MongoDB database to an S3 bucket on daily basis. After few months there will be a huge number of backup copies and you would require deleting older copies manually which might be a tedious job. Here comes Lifecycle rule as a handy feature.

You can create a Lifecycle rule that defines the lifecycle of the S3 objects. The lifecycle can be defined as:

How many days should the objects remain in an S3 bucket?

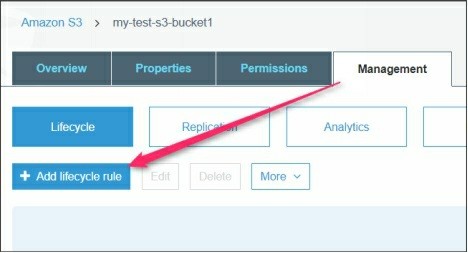
After how many days these objects should go into Glacier storage? After how many days the objects should be deleted permanently from AWS cloud?

Let’s create a Lifecycle rule that will automatically delete the object after 30 days from an S3 bucket.

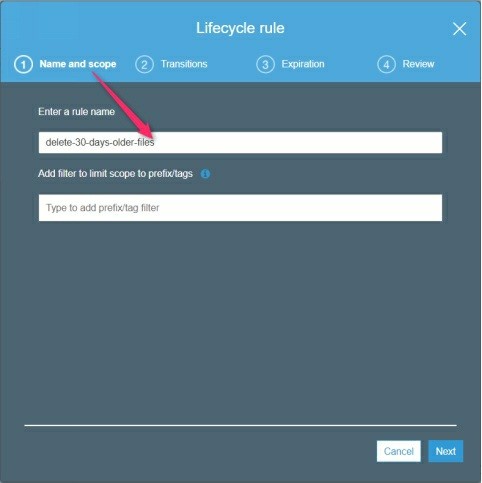
To create and configure S3 lifecycle rule, you need to perform the following steps:

1. Select an S3 bucket for which you want to create lifecycle rule and select the **Management** tab.

2. Click the **Add lifecycle rule** to add a new lifecycle rule.

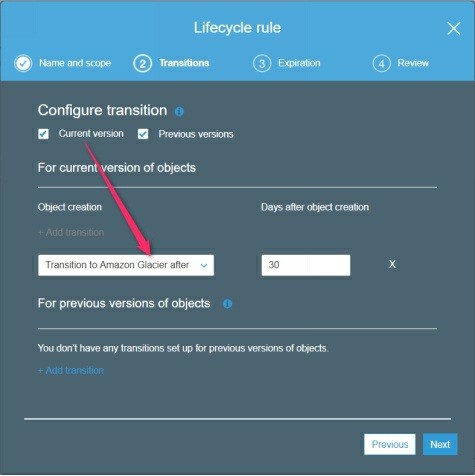


3. On the **Lifecycle rule** dialog box, specify the name and scope as shown in the following figure.

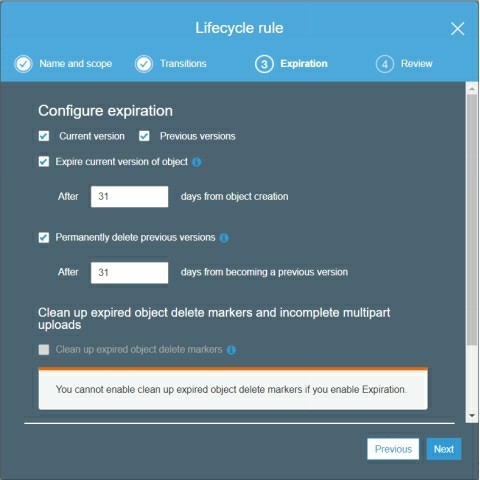


4. On the **Transitions** page, select the current versions or previous versions or both as per your S3 bucket settings and requirements and proceed to next page.

5. Click **Add transition** and select the destination where objects should go after deletion. Also, select the number of days after which objects should be moved to transition storage.



6. On the **Expiration** page, specify the number of days after which object should be deleted permanently from the transition storage which is **Standard –IA** and **Amazon Glacier** also. This setting should be greater than the previously defined value in our case 30 days.



7. On the **Review** page, review all the settings you have selected, click

**Previous** to modify any settings and finally finish the wizard.

8. Now the objects will be moved to transitions storage after the specified days and then will be permanently deleted from the transition storage after the specified days automatically.

40. Implementing Cross-Region S3 Replication Replication is a process of synchronizing and updating content between two objects, places, or services. By default S3 buckets store a single copy of its objects in the region where it is created. But what will happen if that region becomes unavailable due to any reason? Your bucket and its entire data will be inaccessible until the source region comes up again. Here comes cross- region replication as a solution for this. Cross-region replication is a bucket- level feature that enables automatic, asynchronous copying of objects across buckets in different AWS regions.

**Configuring cross-replication of S3 bucket**

In order to configure cross-region S3 replication, let’s consider the following:

Create an S3 bucket in the destination region where you will replicate your source S3 bucket.

Make sure that the versioning is enabled for both source and destination buckets.

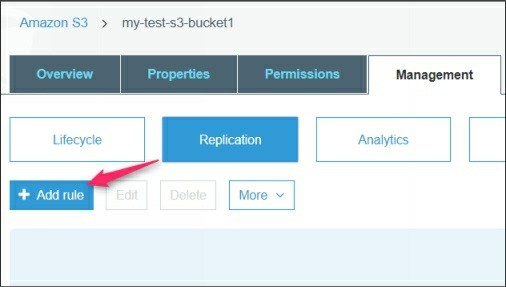
Let’s assume our source bucket is my-test-s3-bucket1 is in Tokyo region and the destination bucket is my-dest-s3-bucket1 is in Sydney region.

**Creating Replication Rule**

To configure cross-region S3 bucket replication, you need to perform the following steps:

1. Select the source s3 bucket and select the **Management** tab.

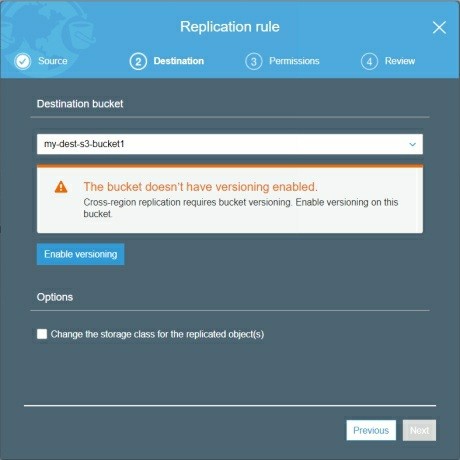
2. Click **Replication** and then click **Add rule** as shown in the following figure.



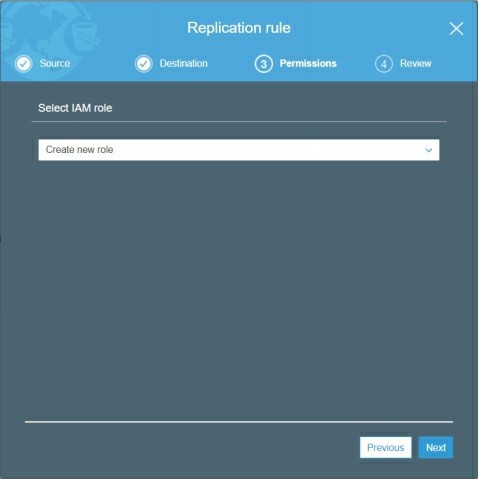
3. On the source page of **Replication rule** page, enable version if not already and then click **Next**.

4. On the **Destination** page, select the destination bucket name in this case my-dest-s3-bucket1.

5. Enable the versioning if not enabled already and then click Next and shown in the following figure.



6. On the **Permissions** page, select **Create new role** or select an existing role if you created already.



7. On the **Review** page, review the settings and click **Save** to finish the wizard. You will see that a new replication rule in the replication rule for source S3 bucket.

**Verifying Cross-region S3 Bucket Replication**

To test whether the cross-region S3 replication is configured and working properly, upload a test file in the source S3 bucket and check the destination S3 bucket objects, the uploaded object in the source S3 bucket should be displayed in the destination S3 bucket.

**Troubleshooting S3 Cross-region Replication**

If you face any issue or replication does not happen, refer the following article for the common issues with cross-region S3 bucket replication.

[Common issues with cross-region S3 bucket replication.](https://docs.aws.amazon.com/AmazonS3/latest/dev/crr-troubleshoot.html)

41. Enabling and configuring AWS CloudTrail Typically, multiple IAM users performed various activities to work with AWS cloud services. As an AWS Administrator, you should have a method that you can use to view what actions have been performed by which IAM users. Here, comes AWS CloudTrail as a possible solution for this.

[Overview of AWS CloudTrail](http://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-user-guide.html)

[CloudTrail Pricing](https://aws.amazon.com/cloudtrail/pricing/)

AWS CloudTrail is an AWS service. It helps you enable control, compliance, and functioning and risk inspecting of your AWS account. AWS API calls or activities performed by a user, role, or an AWS service are recorded as events in CloudTrail. These events include actions taken in the AWS Management Console, AWS CLI tool, and AWS SDKs and APIs.

**Enabling CloudTrail**

CloudTrail is enabled by default for your AWS Account. However, you still need to create and configure CloudTrail settings.

**Creating a CloudTrail**

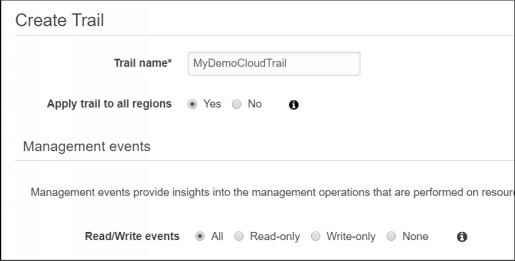
To create a CloudTrail, navigate to the CloudTrail console.

1. Select **Trails** in the left pane and click the **Create trail** button.

2. On the **Create Trail** page, specify the **Cloud Trail** name and select regions for which you want to collect trail logs.

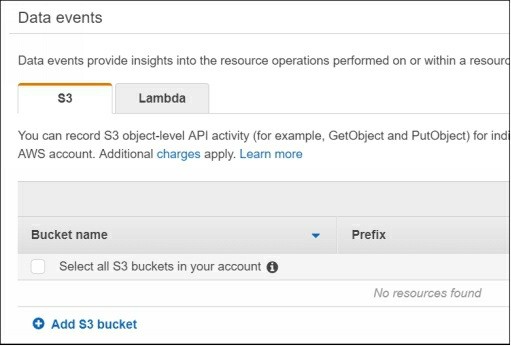
3. On the **Management Event** section, select **All** as **Read/Write**

**Events** radio button as shown in the following figure.



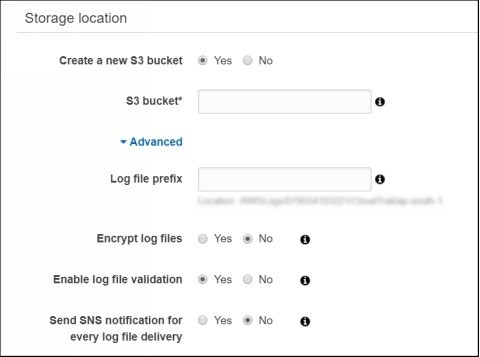
4. On the **Data events** section, select either S3 or Lambda services to record APIs logs. You can also select specific lambda function or S3

bucket depending on your choice.



5. On the **Storage location** section, select **Yes** to create a new S3 bucket where your CloudTrail logs will be stored.

6. I the **Advanced** section, you can specify a prefix to limit the trail logs, you can encrypt the logs in S3 bucket, or you can also select an SNS topic to receive notifications. For the demo purpose, default options are enough.



Now your logs will be recorded and stored in the specified S3 bucket. The following is the location and format of the CloudTrail Logs.

***bucket\_name/prefix\_name/AWSLogs/Account***

***ID/CloudTrail/region/YYYY/MM/DD/file\_name.json.gz***

However, the recent logs are also visible through the Event history section of

the CloudTrail console.

42. Working with Auto Scaling Group

Auto Scaling Group is one of the highly demandable cloud services in nowadays. Almost every organization want to design and implement a solution that is auto scalable as per the business or application need. This can be done using the AWS Auto Scaling Group and Launch Configuration template.

[Overview of Auto Scaling Group](https://docs.aws.amazon.com/autoscaling/latest/userguide/WhatIsAutoScaling.html)

**Creating a Launch Configuration Group**

A launch configuration specifies the pre-defined settings that will be used by your Auto Scaling group. A launch configuration includes information such

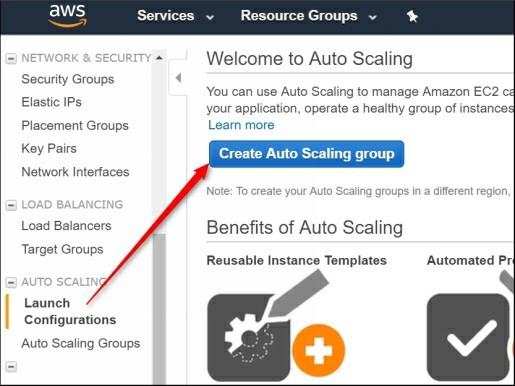
as the AMI ID to use for launching the EC2 instance, instance type, key pairs, security groups, and volumes, and few other configuration settings.

To create a Launch Configuration, you need to perform the following steps:

1. Select **Launch Configuration** in the **EC2 dashboard** under the

**Auto Scaling** section.

2. Click **Create Auto Scaling Group** to proceed as shown in the following figure.



3. On the **Create Auto Scaling Group** page, click **Create Launch**

**Configuration** to proceed next.

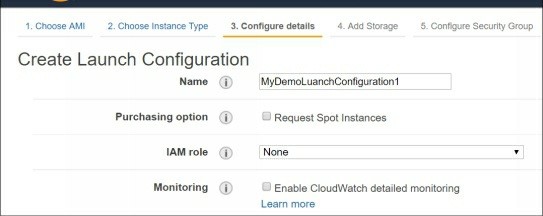
4. On the next page, select the **AMI ID** that will be served as the base

EC2 instance for your Auto Scaling Group. For the demo purpose,

select **Amazon Linux AMI** and proceed to the next page.

5. On the **Choose Instance Type** page, select **T2.Micro** and proceed to the next page.

6. On the **Configure Details** page, specify the launch configuration name, and accept the other settings as default.



7. On the **Add Storage** page, specify the storage size and additional volumes, if required, and proceed to the next page.

8. On the **Configure Security Group** page, select **Create a new security group**, specify the security group name and description, allow SSH port for your public IP, and proceed to the next page.



9. On the **Review** page, click **Create Launch Configuration** and review your settings. Modify the settings if you wish, else click **Review** to proceed to the next page.

10. On the **Select key pair** page, select **Create a new key pair** or select an **existing key pair** that you want to use to connect your Auto Scaling instances later.

11. Finally, click **Create Launch Configuration** to create it. You will be redirected to the Create Auto Scaling Group page.

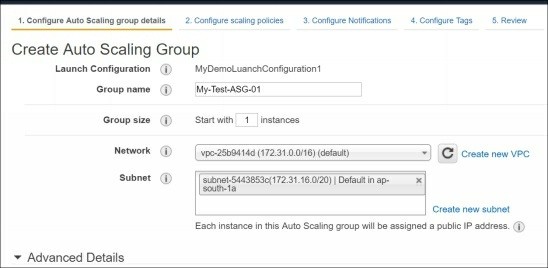
**Creating Auto Scaling Group**

We assume that you are on the **Create Auto Scaling Group** page from the

previous step.

To create an Auto Scaling Group, you need to perform the following steps:

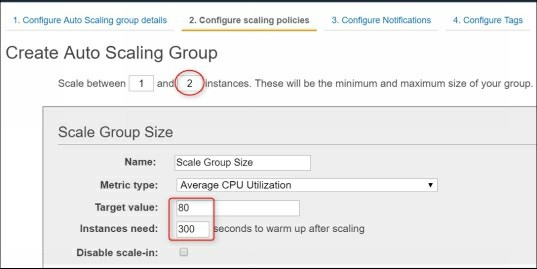
1. On the **Create Auto Scaling Group** page, Specify the Group name, Group size, Network, and subnet as shown in the following figure.



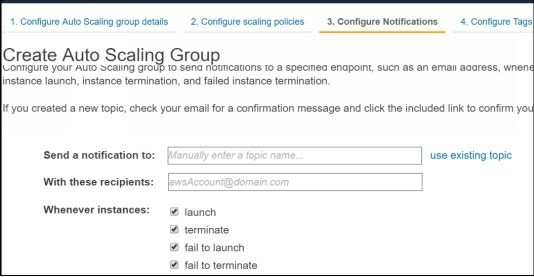
2. Click **Configure Scaling Policies** to configure it.

*Note: Scaling policies define when your Auto Scaling group take the decision to increase and decrease the size of Auto Scaling group or number of instances.*

3. For example, let’s assume: - we want to add one more instance to my auto scaling group (My-Test-ASG-01) if the CPU utilization of the first instance remains more than 80 % for 5 minutes. The maximum number of instances should not increase more than 2 instances. The policy settings will be as follow for the above requirements.



4. On the **Configure Notifications** page, you can specify an SNS topic with your email IDs list where the mail notifications will be sent when there is any change in the Auto Scaling Group such as adding/removing instances (or increasing/decreasing size of your Auto Scaling Group).

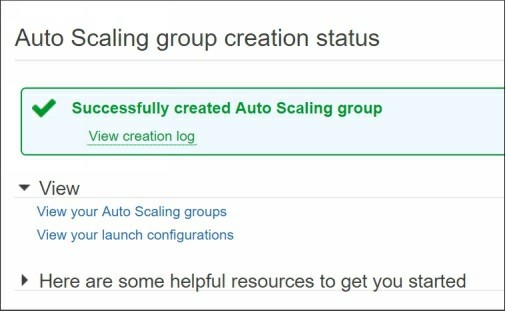


5. On the **Configure Tags** page, specify the name of your ASG such as

**My-Test-ASG-01** and proceed to the **Review** page.

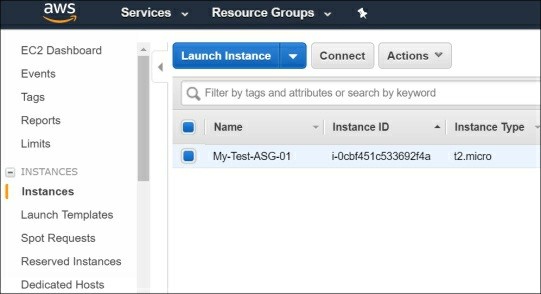
6. Finally, click **Create Auto Scaling Group** to complete the wizard.

One you ASG is created you will get the successful creation message as shown in the following figure.

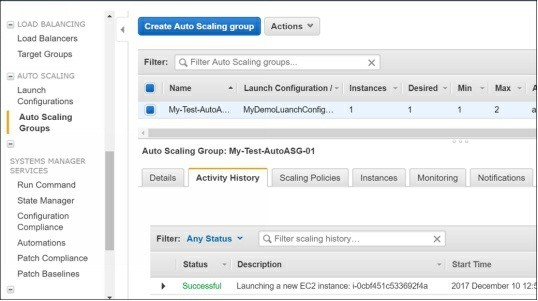


**Verifying Auto Scaling Group Configuration**

1. Now, you have configured your ASG, go the instance list and verify that your instance is running as shown in the following figure.



2. Now navigate to the Auto Scaling Group option and review the settings as shown in the following figure.



3. We recommend you to spend some time to review each of the auto scaling group tabs and try to understand what information these tabs provide you.

**Modifying Auto Scaling Group Settings**

Now, let’s do some fun with our Auto Scaling Group.

1. Click **Actions** and select **Edit** to edit the settings.

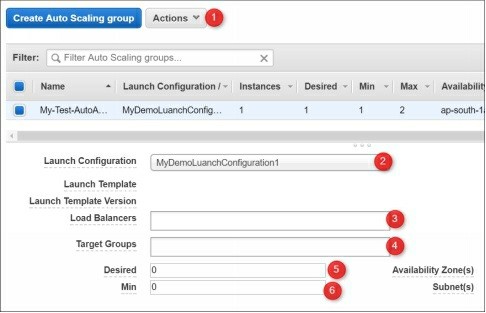
The **Launch Configuration** option allows you to change launch configuration.

The **Load Balancer** and **Target group** options allow you to attach this ASG to your desired load balancer.

The D**esired** and **Minimum** options allow you to set the desired

and minimum number of instances that you want for this ASG.

2. Here, let's change the Desired and Minimum values as 0 as shown in the following figure and save the configuration.



3. Now go back to the instance list and wait for the few minutes. You will see that your instance will be terminated automatically to meet with the configuration of your Auto Scaling Group. Hope, you have understood the power of AWS Auto Scaling Group.

That’s all you need to create and configure the Auto Scaling Group. Optionally you can integrate this Auto Scaling Group with your Load Balancer.

43. Configuring Amazon Route 53

Typically one misconception about the Amazon Route 53 is that most people think Amazon Route 53 as a just a DNS name resolution service. However, Amazon Route 53 is more than this.

[Overview of Amazon Route 53](http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/Welcome.html)

Amazon Route 53 helps you to achieve the following major functions:

Allows you to register and host your domain names such as [www.example.com.](http://www.example.com)

Allows you to route internet traffic to the resources of your domain. Check the health of your resources.

Helps you to perform load balancing.

Although, here we will not go in the deep theory of Route 53. We will just walk through the high-level steps to help you to understand the concept and usage of Amazon Route 53.

*Note: Amazon Route 53 does not allow you to register a domain name as a free tier. You have to purchase the domain name at your own choice. If you don’t want to spend unnecessary charges. Just follow the steps and try to understand what steps are necessary to work with Amazon Route 53.*

**Registering a domain name**

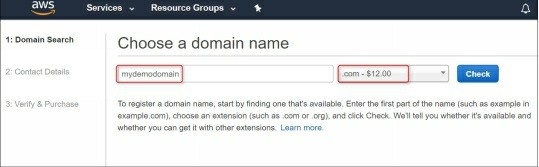
To register a Domain Name using Amazon Route 53, you need to perform the following steps:

1. Visit the **Route 53 console** page using the following link. <https://console.aws.amazon.com/route53/home>

2. Click **Get Started Now** under the **Domain registration** section.

3. On the **Register Domain** page, either you can register a new Domain name or you can transfer an existing domain name if you own it already.

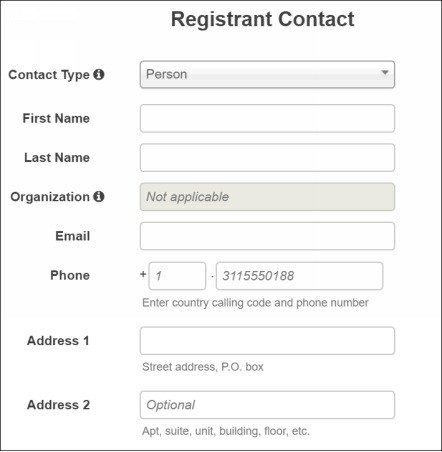
4. For this demo, let’s click **Register new domain**. On the **Choose a domain name** page, select your top level domain such as .com and type your domain name you want to register as shown in the following figure.



5. Click **Check** to check whether this domain name is available to register or not.

6. It may or may not be available, once you find your suitable available domain name, click **Add to card** and click **Continue** proceed to the next page.

7. On the **Contact details** page, fill the accurate contact details for the registering domain name and proceed to the next page.



8. On the **Verify and purchase** page, review your settings and click **Complete Purchase** to purchase it or cancel it to avoid getting charged if you are doing this just for the learning purpose.

9. You will get a verification email that you specified in the contact details. Once you verify the email, your domain registration is almost done. However, sometimes it may take up to 3 days to change the status of your registering domain from pending to confirm in the Route 53 Dashboard.

44.Working with Amazon WorkDocs

Amazon WorkDocs is a fully Amazon managed, highly-secure, enterprise- level storage and sharing service. Unlike S3 based stored files, you can also share your files with other members of your organization for the collaboration or review.

Before to proceed for the Amazon WorkDocs, let’s have look at what

Amazon says about its pricing:

“*With Amazon WorkDocs, there are no upfront fees or commitments. You pay only for active user accounts, and the storage you use. In most regions, WorkDocs costs $5 per user per month and includes 1 TB of storage for each user. WorkDocs provides a* ***30-day free trial*** *with 1 TB of storage per user for up to 50 users. You can invite guest users to log in and view files shared with them at no additional charge.”*

As per the above statement, you have 30 days free trial for up to 1 TB of storage that should be more than enough for the learning purpose.

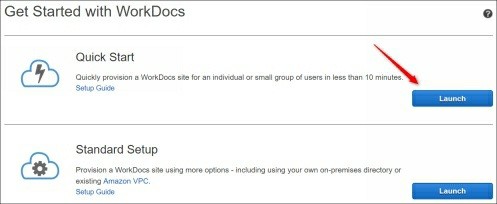
**Creating an AWS WorkDocs Site**

To setup Amazon WorkDocs, you need to perform the following steps:

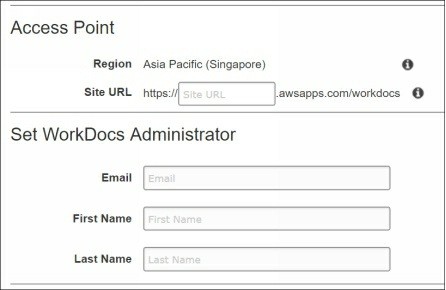
1. Navigate to the **Amazon WorkDocs** home page for the supported region (not every region supports this feature).

2. Click the **Get Started Now** button to proceed to the next page.

3. Here, you will see the **Quick and Standard** setup options as shown in the following figure. For the learning purpose, **Quick Start** setup guide should be enough. So proceed with this.



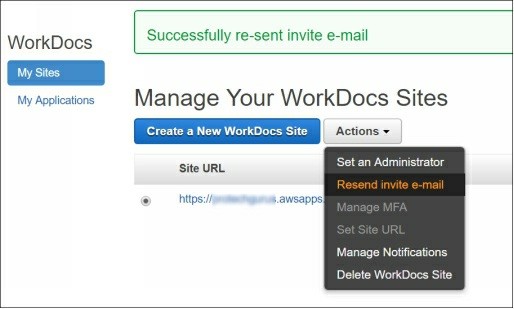
4. On the next page, you need to specify the site URL, email, and name details as shown in the following figure.



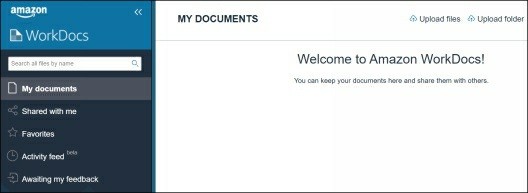
5. Finally, click **Complete Setup** to complete the wizard. Your WorkDocs site will be started to initialize and should be available after some time. In fact, you will get an email once your WorkDocs site is ready.

6. Now, click the invite link you receive in your email box and set the desired password on the next page.

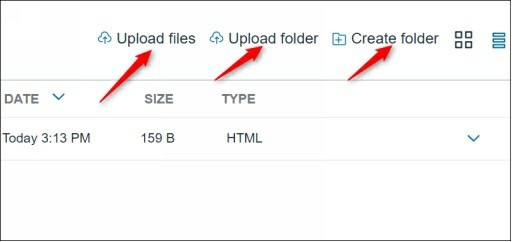
7. If you didn’t get an invite mail yet, select your created WorkDocs site, click **Actions**, and select **Resend invite e-mail** as shown in the following figure.



8. Now, click your **WorkDocs site link**, type your registered email ID and login to WorkDocs console. You will see the WorkDocs console similar to as shown in the following figure.

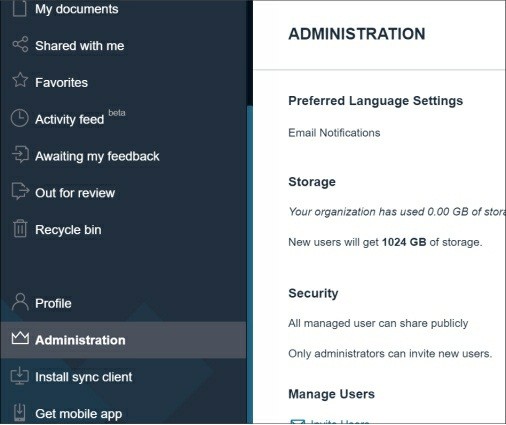


9. In the right pane, you have various options to upload files and folders or create a new folder structure as shown in the following figure.



10. In the left navigation pane, there are various options to work with Amazon WorkDocs as shown in the following figure. We recommend you try to explore each one of them for few minutes to understand

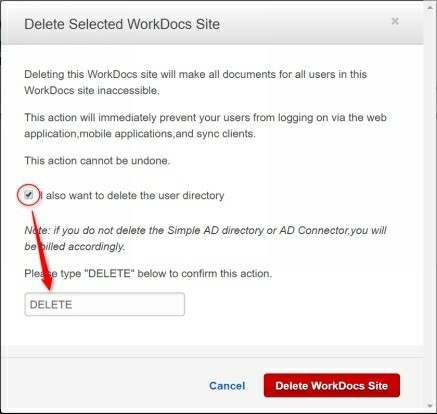
and get familiarized with them.



**Deleting WorkDocs Site**

Once your activity is done, please delete your WorkDocs site to avoid any unnecessary charges. For this, select your WorkDocs site, click **Actions**, and select **Delete WorkDocs sites**.

Follow the on-screen instructions as shown in the following figure and complete the deletion process.



45. Working with AWS Trusted advisor

AWS Trusted Advisor is one of my favorite tool to optimize cost, enhance security, and improve the performance of AWS services. Trusted Advisor provides real-time guidance to help you provision your resources that meet with AWS best practices. However, full features of AWS Trust Advisor are only available for those who have Business Support plan. As a free tier account, you are not eligible to use full features of AWS trust advisor.

[AWS Trusted Advisor Overview](https://aws.amazon.com/premiumsupport/trustedadvisor/)

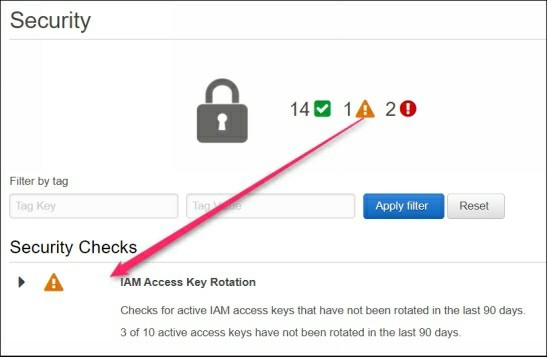
Let’s have a quick example, how Trust Advisor can help you: Let’s assume you have 20 security groups and each security group has 10 inbound rules for different services. You can’t remember or check every time that what ports I have allowed to whom. Suppose you have allowed SSH port to open world as

0.0.0.0/0 for the testing purpose and later forget to fix it. Trust Advisor will tell about that security group risk.

Let’s assume you have created an RDS instance for the learning purpose and forget to delete it. Since it’s in an idle state, when you will visit the Trust Advisor console it will tell you that you can stop this idle instance, RDS, volume etc. to save the cost.

The following is the sample Trust Advisor’s report that tells what’s good and bad about a particular AWS Account.

Select and open a specific dashboard option to know more in detail. For example, the following figure shows the details of Security issues. As you can see, as I click on the warning icon it shows me the issue that is with warning severity.



Sometimes AWS Trust Advisor may show you the idle instances that may be occasionally in use. Or you might have a security group rule open to the world as it might be the requirement of your application. But AWS Trust advisor may show you these requirements as issues. So you need to take the decision very carefully while working on the real world productions infrastructures.

Thank You

Thanks for reading this guide with patience. Hope, this book helped you to gain some really nice and useful hands-on skills in major AWS Cloud Services. We would really like to hear your feedback on the content quality, good and bad things. Your feedback will help us to improve the quality. If you think this book useful for you, please give us an appropriate rating on

this and be connected to the other books as well. We will be adding more and

more lab exercises in the upcoming editions of this book as technology never stops inventing new things.