AMAZON EC2

What Is Amazon EC2?

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

Features of Amazon EC2:

* Virtual computing environments, known as instances.
* Preconfigured templates for your instances, known as Amazon Machine Images (AMIs), that package the bits you need for your server (including the operating system and additional software).
* Various configurations of CPU, memory, storage, and networking capacity for your instances, known as instance types.
* Secure login information for your instances using key pairs (AWS stores the public key, and you store the private key in a secure place).
* Storage volumes for temporary data that's deleted when you stop or terminate your instance, known as instance store volumes.
* Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as Amazon EBS volumes.
* Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as regions and Availability Zones.
* A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using security groups.
* Static IPv4 addresses for dynamic cloud computing, known as Elastic IP addresses.
* Metadata, known as tags, that you can create and assign to your Amazon EC2 resources.
* Virtual networks you can create that are logically isolated from the rest of the AWS cloud, and that you can optionally connect to your own network, known as virtual private clouds (VPCs)

## How to Get Started with Amazon EC2

First, you need to get set up to use Amazon EC2. After you are set up, you are ready to complete the Getting Started tutorial for Amazon EC2. Whenever you need more information about an Amazon EC2 feature, you can read the technical documentation.

**Get Up and Running**

* [Setting Up with Amazon EC2](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/get-set-up-for-amazon-ec2.html)
* [Getting Started with Amazon EC2 Linux Instances](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html)

**Basics**

* [Instances and AMIs](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instances-and-amis.html)
* [Regions and Availability Zones](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html)
* [Instance Types](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-types.html)
* [Tags](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Using_Tags.html)

**Networking and Security**

* [Amazon EC2 Key Pairs](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html)
* [Security Groups](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-network-security.html)
* [Elastic IP Addresses](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html)
* [Amazon EC2 and Amazon VPC](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-vpc.html)

**Storage**

* [Amazon EBS](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AmazonEBS.html)
* [Instance Store](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/InstanceStorage.html)

**Working with Linux Instances**

* [Remote Management (Run Command)](https://docs.aws.amazon.com/systems-manager/latest/userguide/execute-remote-commands.html)
* [Tutorial: Install a LAMP Web Server on Amazon Linux 2](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-lamp-amazon-linux-2.html)
* [Tutorial: Configure Apache Web Server on Amazon Linux 2 to Use SSL/TLS](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/SSL-on-an-instance.html)
* [Getting Started with AWS: Hosting a Web App for Linux](https://docs.aws.amazon.com/gettingstarted/latest/wah-linux/)

## Pricing for Amazon EC2

When you sign up for AWS, you can get started with Amazon EC2 for free using the [AWS Free Tier](https://aws.amazon.com/free/).

**Amazon EC2 provides the following purchasing options for instances:**

**On-Demand Instances**

* Pay for the instances that you use by the second, with no long-term commitments or upfront payments.

**Reserved Instances**

* Make a low, one-time, up-front payment for an instance, reserve it for a one- or three-year term, and pay a significantly lower hourly rate for these instances.

**Spot Instances**

* Request unused EC2 instances, which can lower your costs significantly.
* For a complete list of charges and specific prices for Amazon EC2, see [Amazon EC2 Pricing](https://aws.amazon.com/ec2/pricing).
* To calculate the cost of a sample provisioned environment, see [Cloud Economics Center](https://aws.amazon.com/economics/).
* To see your bill, go to your [AWS Account Activity page](https://aws.amazon.com/account-activity). Your bill contains links to usage reports that provide details about your bill. To learn more about AWS account billing, see [AWS Account Billing](https://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/).
* If you have questions concerning AWS billing, accounts, and events, [contact AWS Support](https://aws.amazon.com/contact-us/).
* For an overview of Trusted Advisor, a service that helps you optimize the costs, security, and performance of your AWS environment, see [AWS Trusted Advisor](https://aws.amazon.com/premiumsupport/trustedadvisor/).

## How to create an Instance

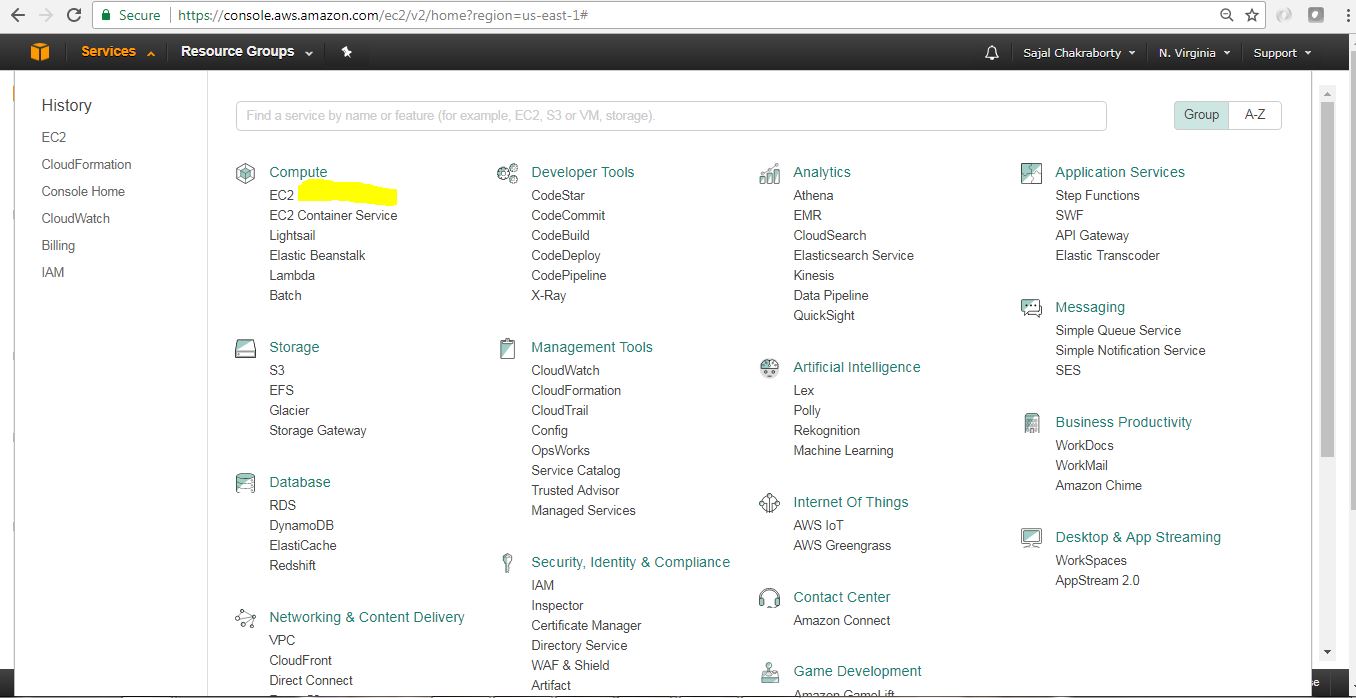
## Prerequisites

Before starting up this exercise we need below setup as prerequisites.

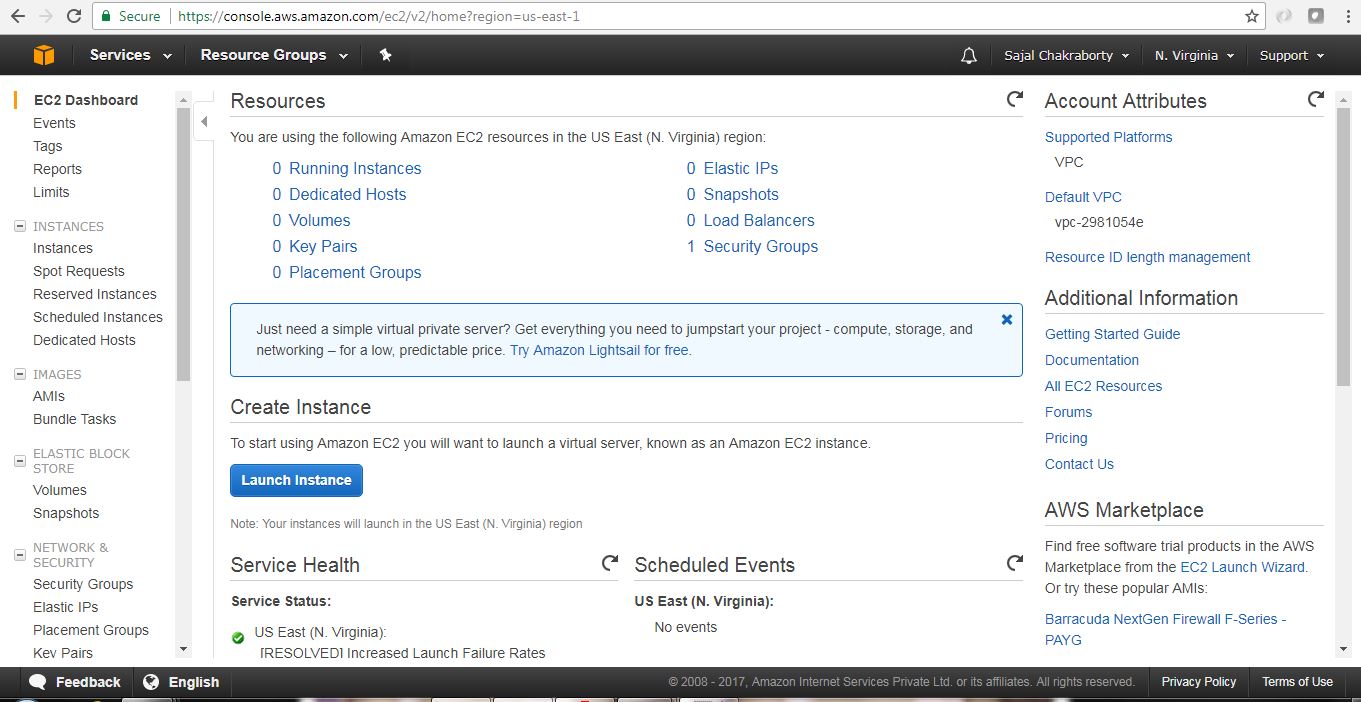
* **AWS account** – Must have to open one AWS account to do this exercise.
* **SSL Client – PuTTY** – This should be installed in Local M/C including PuTTYgen, PUTTY, Pageantapplications. Full set of applications related to PutTTY can be downloaded from [this link](https://the.earth.li/~sgtatham/putty/latest/w64/putty.zip). Just unzip to a convenient location.

So, let’s get started by creating an AWS EC2 instance.

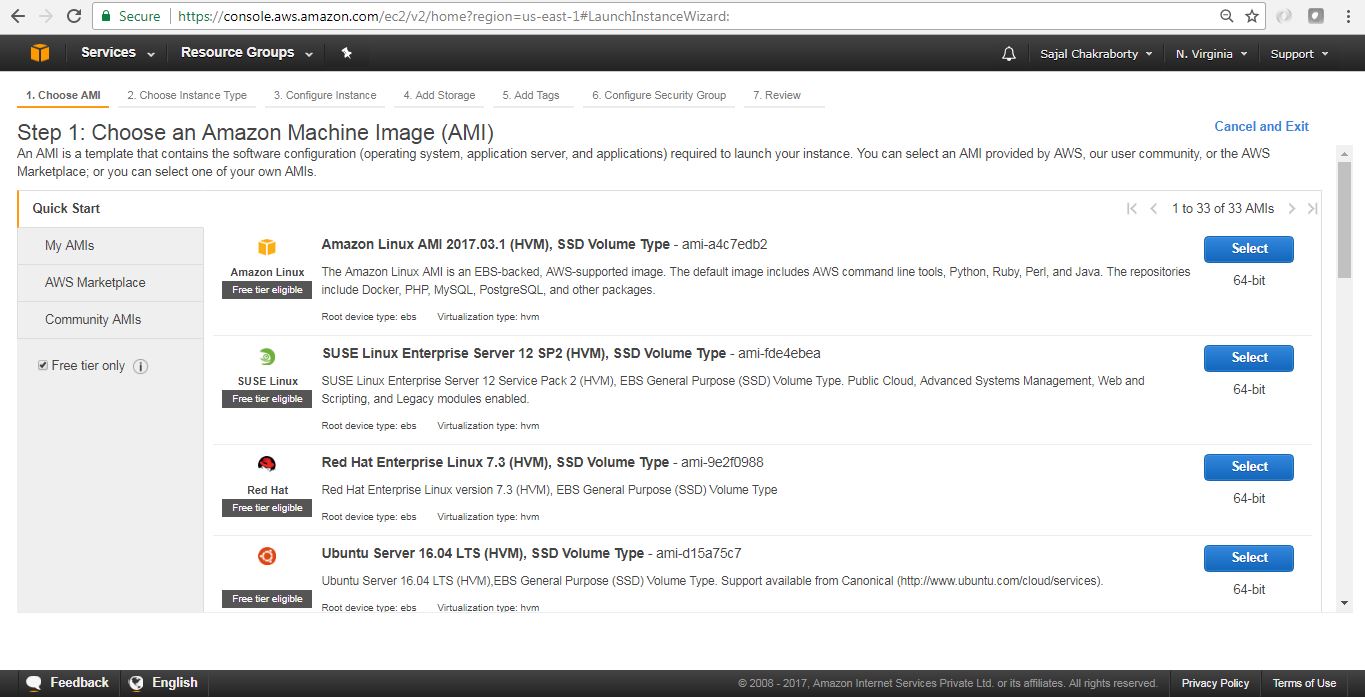
## Steps to Create EC2 Instance

1. Login to [AWS console](https://console.aws.amazon.com/console) and open EC2 home screen – Once log in and click Services menu in the top left corner of the home screen, we will need to click the EC2 Link under compute section. The EC2 landing page will look like :[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/EC2_0_landingScreen.jpg)Figure 1.0 : Services Menu with Ec2 Link

This is how EC2 Home page looks like. This is the place we land when we click EC2 link in the previous step (Figure 1.0).

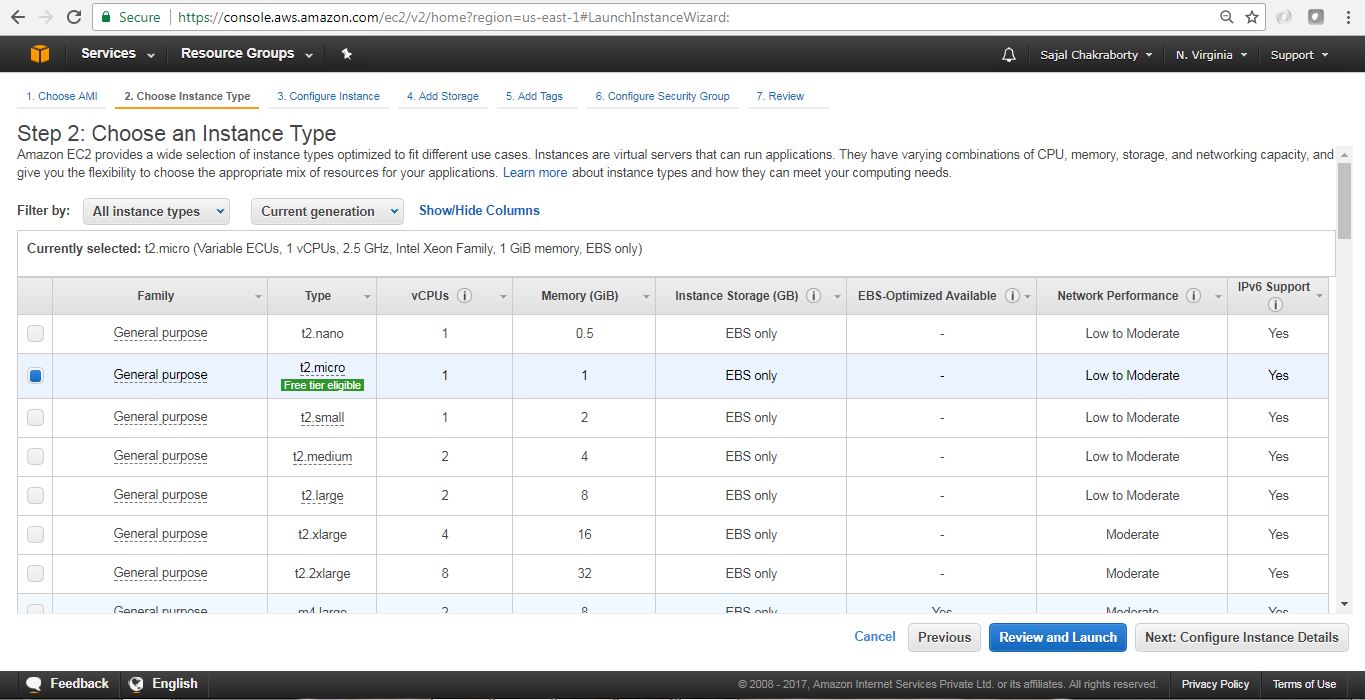
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_1.jpg)Figure 1.1 : EC2 Home Screen

1. **Launch a new virtual server** Now we need to click the Launch Instance button from the EC2 Home page.This will start the instance creation wizard which will guide us through the following steps mainly:
   * Selecting an OS
   * Choosing the size of your virtual server
   * Choosing the Extra attached storage we need for this instance
   * Configuring details
   * Reviewing your input and selecting a key pair for SSH
2. **Select AMI** – AMI stands for **Amazon Machine Images**, which is kind of a blue print of the instance that we will create, it tells about the Operating System of the Instance as well as the basic softwares that will be pre-installed. So in this step, while choosing AMI, we are choosing the underlying Operating System(OS) and the preinstalled software bundles that would be available in the instance upfront.We need to choose AMI based on our need. We will choose only those AMIs which falls under Free Tire Eligible type. An AMI is the basis your virtual server starts from. AMIs are offered by AWS, by thirdparty providers, and by the community. We can also create custom AMI based on our need. Some AMIs are chargeable. Here for this exercise we will choose Ubuntu Server 16.04 LTS(HVM) AMI.

[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_2.jpg)Figure 3.0 : Select AMI

Click on the Select button beside the AMI to proceed with the sect step.

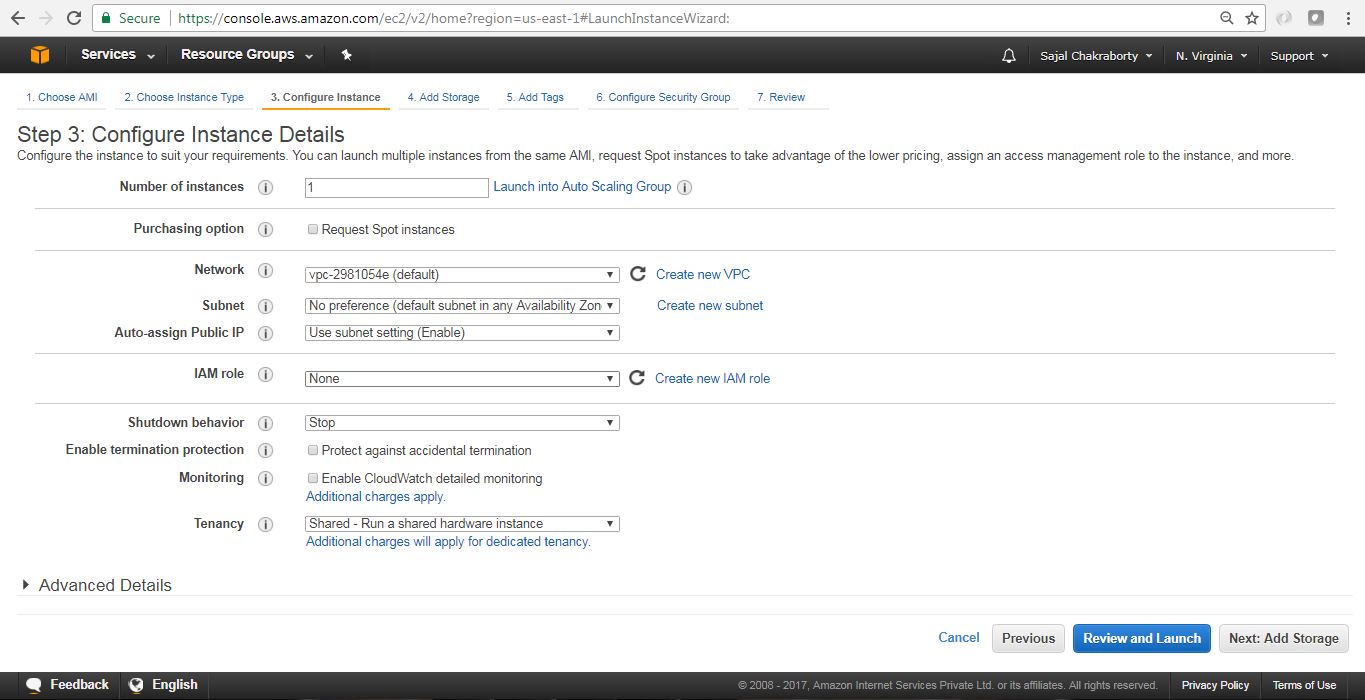
1. **Select Instance Type** – It’s now time to choose the computing power needed for your virtual server. Instance Type indicates the computing power of the instance by allocating the amount of virtual hardware dedicated to the instance. On AWS, computing power is classified into instance types. An instance type primarily describes the number of vCPUs and the amount of memory an instance will be allocated to once created. We will choose only the Free Tire Eligible Instance Type (t2.micro) which provides 1 vCPU and 1 GB of Memory. AWS has already defined many [instance types](https://aws.amazon.com/ec2/instance-types/), we will now choose the t2.micro instance type for this exercise. Once instance type is selected, click on button Next: Configure Instance Details in the bottom right corner of the page to proceed with the next steps.

[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_3.jpg)Figure 4.0 : Instance Type Selection

1. **Instance Details page** This will look like below. Here we will take all the default values and proceed with clicking Next: Add Storage button at the bottom right corner of the page. Before proceeding to the next section please note few important things that we can modify here:
   * Number of Instances we are going to create in this wizard.
   * We can select the Spot Instance option, Spot Instances are idle compute capacity that AWS makes available based on bid prices from customers.
   * Networking/Subnet/Public IP setting – This is the VPC under which the instance would be created, We can choose existing VPC, or create a new VPC here. VPC is itself a large topic which is out of scope of this article. Please follow official AWS [documentation](https://aws.amazon.com/documentation/vpc/)for the same.
   * Also we can configure, Shutdown Behavior, Termination Protection Flag(To avoid accidental Termination), Tenancy(Common vs dedicated H/W), Cloud Watch Detailed Monitoring. Here some of them are chargeable option.

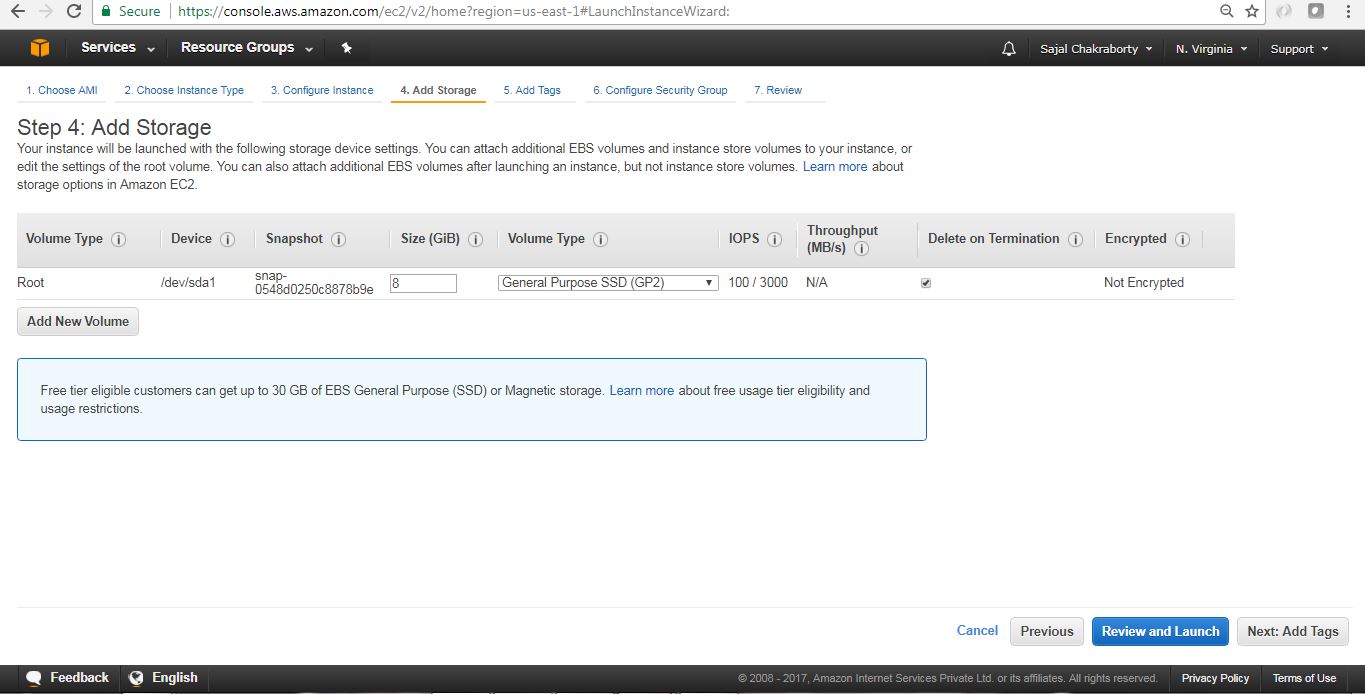
We can Change those later also once the Instance has been created.

Now we will proceed to add EBS (Elastic Block Store) volume details associated with this instance. Configure Instance Details screen will look like

[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_4.jpg)Figure 5.0 : Configure Instance Details

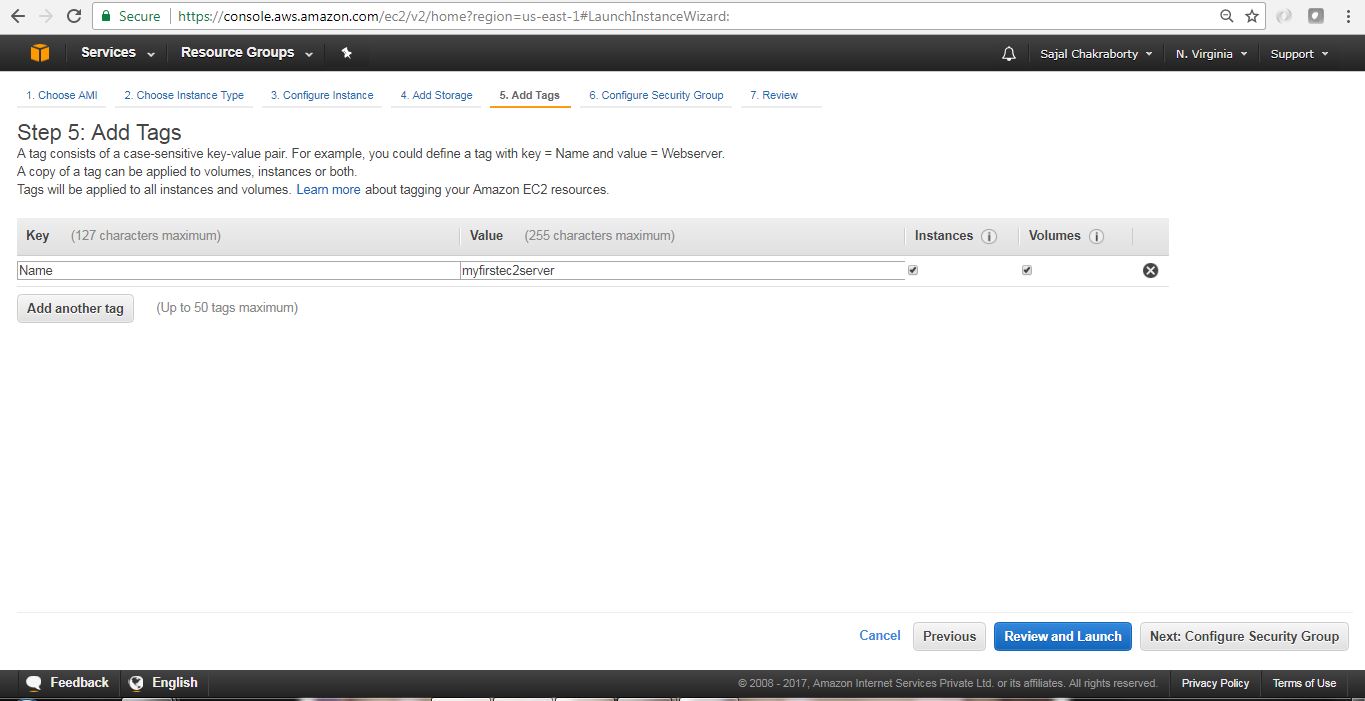
1. **Add EBS Storage** – EBS stands for Elastic block storage. It is basically network-attached storage attached your virtual server. This page will help us to configure Storage that will be associated with the EC2 instance that we are going to launch. We will choose default values to avoid extra cost. Make sure you choose the capacity which falls under Free Tire Eligibility to avoid incurring extra cost.

Review and click Next: Add Tags button at the bottom right corner of the page to proceed. The EBS volume configuration page will look like

[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_5.jpg)Figure 6.0 : EBS volume configuration

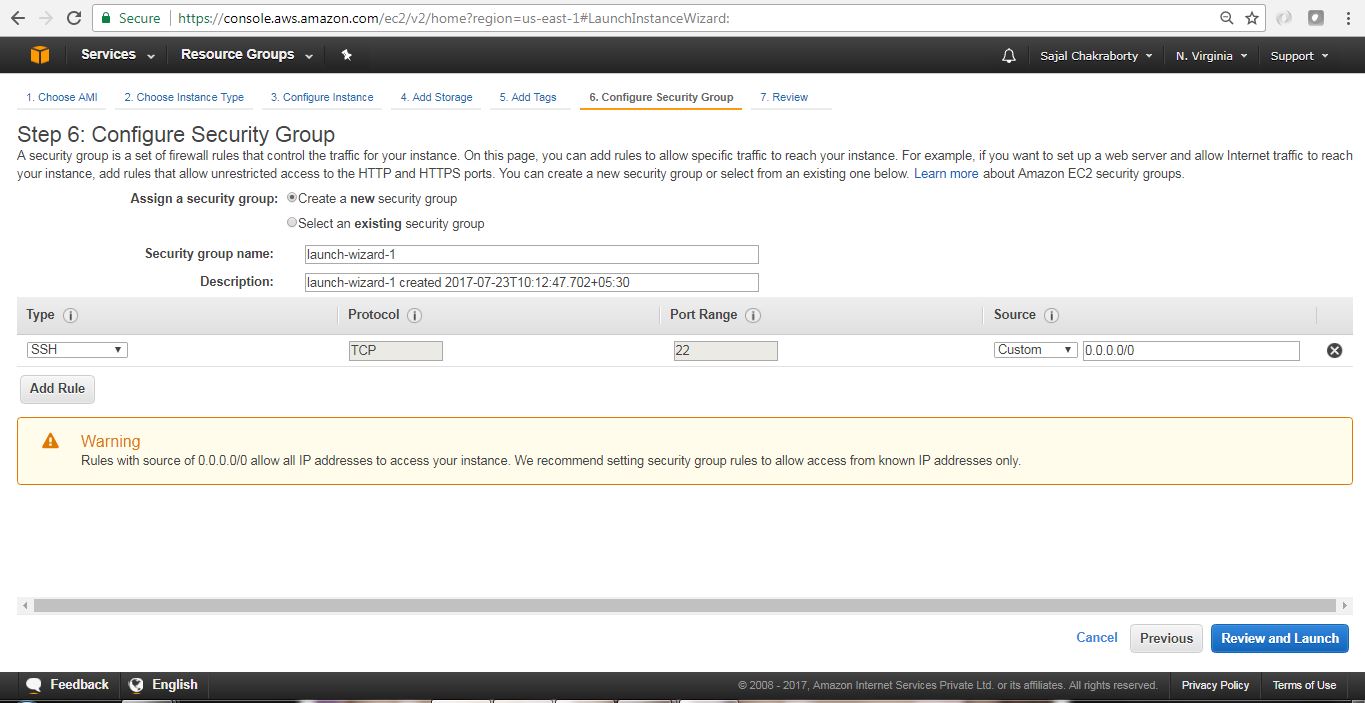
1. **Add Tags** – Tags are kind of an identifier of any AWS resources, with Tags we can easily locate the resource in future and also it helps us to classify the resources. Tags help you to organize resources on AWS. We can add max 50 tags to a particular resources. It is a simple Key value pair associated with the resource.

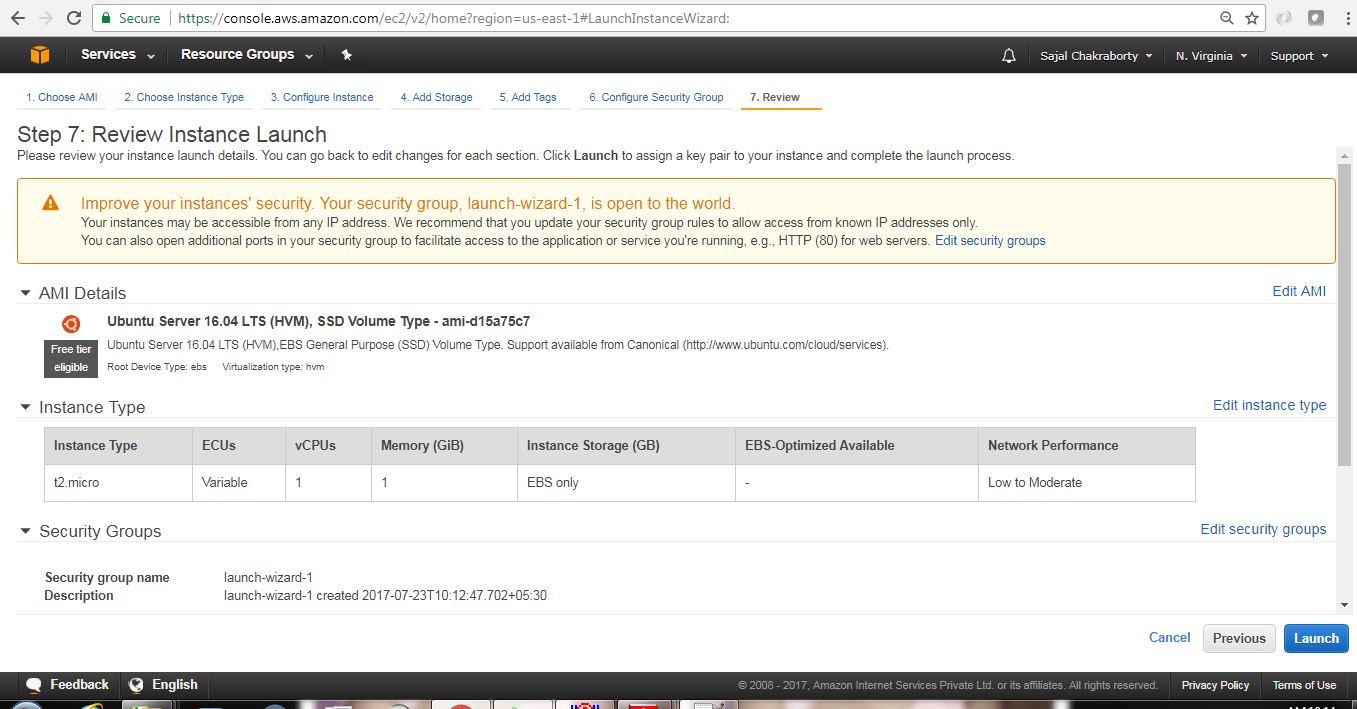
In this step we will add one tag Name with value myfirstec2server. Once tag is created, we will proceed with configuring security group by clicking the “Next: Configure Security Group” button at the bottom right corner of the page. Add Tag screen will look like:

[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_6.jpg)Figure 7.0 : EC2 Add Tag

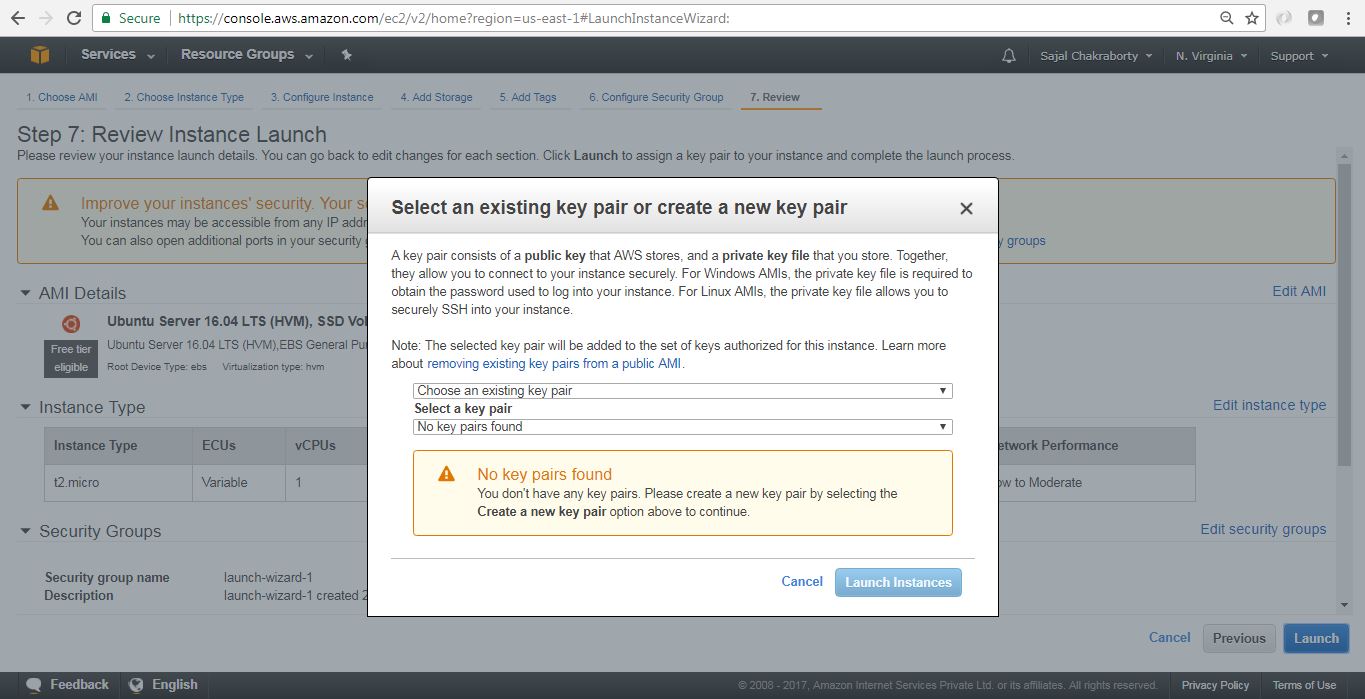
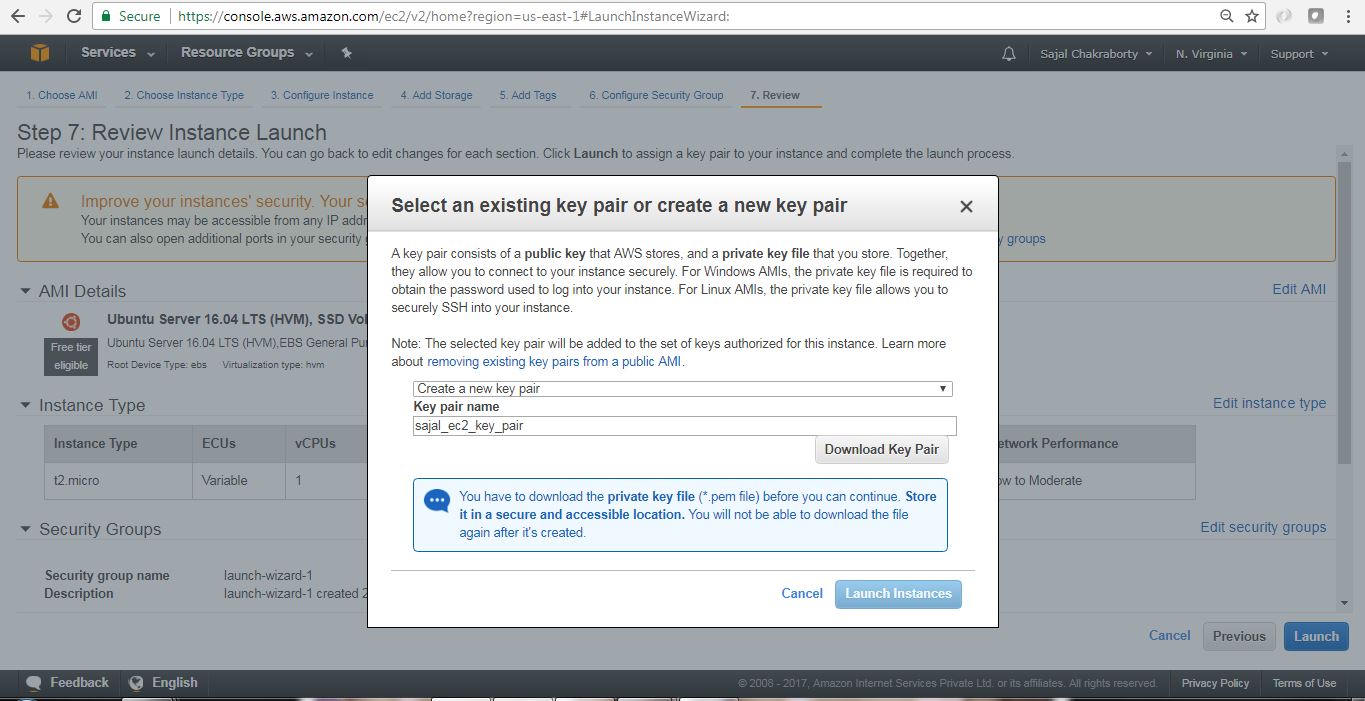
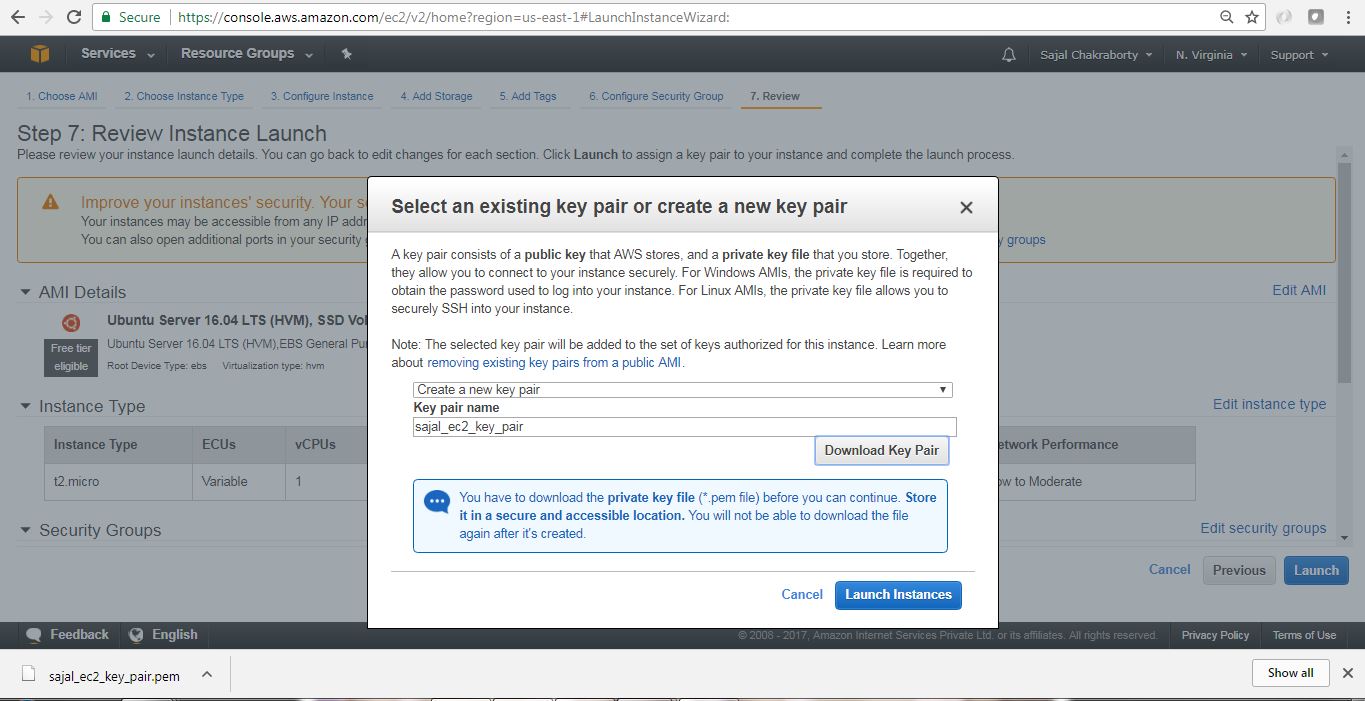
1. **Configure Security Group** – This section is to define a firewall that helps to secure our virtual server. A [security group](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-network-security.html) is a virtual stateful firewall that controls inbound and outbound network traffic to AWS resources and Amazon EC2 instances. All Amazon EC2 instances must be launched into a security group. If a security group is not specified at launch, then the instance will be launched into the default security group for the Amazon VPC. The default security group allows communication between all resources within the security group, allows all outbound traffic, and denies all other traffic.In this screen we can create/reuse security groups based on our need.

In this example, assuming that we don’t have any security group created, We will choose the default option and will proceed. To proceed with next page, we will click on the Review and Launch button at the bottom right corner of the page. The Security Group configuration screen will look like:

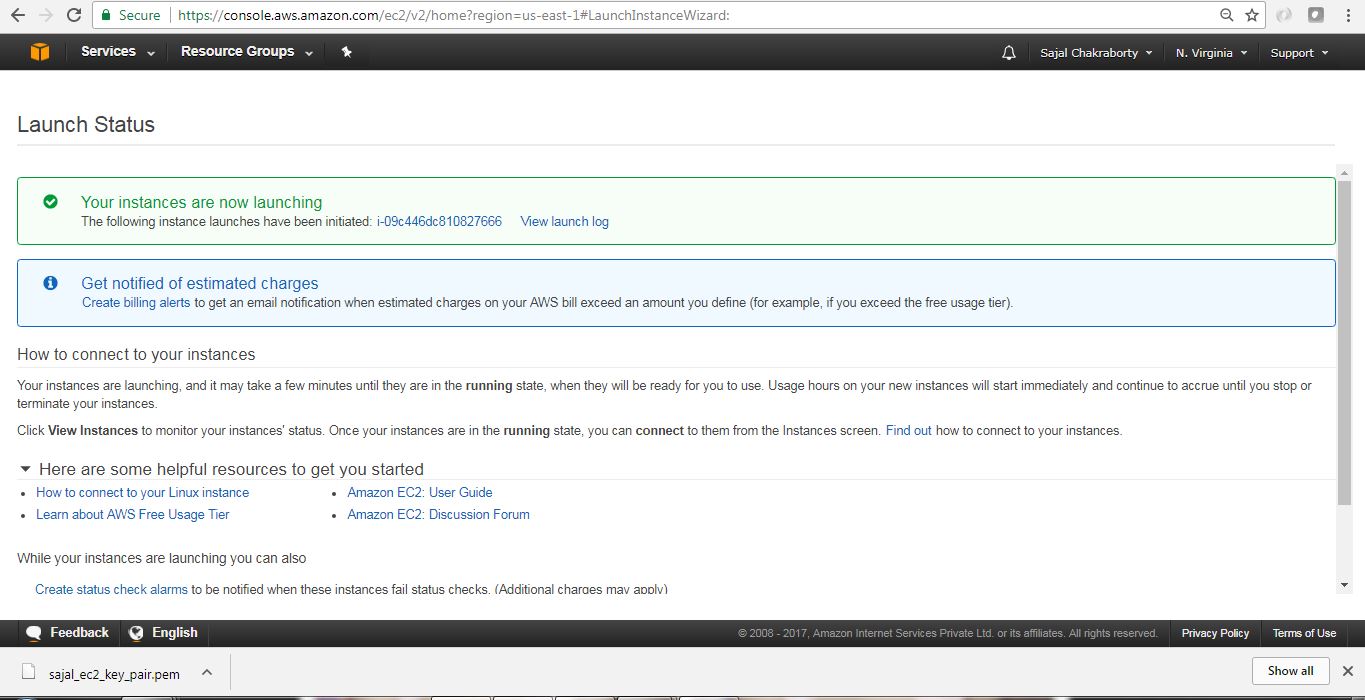
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_7.jpg)Figure 8.0 : Security Group Configuration Screen

1. **Review and associate Key pair** – We will now review all the configurations that we have selected so far in this screen and to proceed further we will click on Launch button at the bottom right corner of the page.[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/EC2_8.jpg)Figure 9.0 : Review Launch configurations

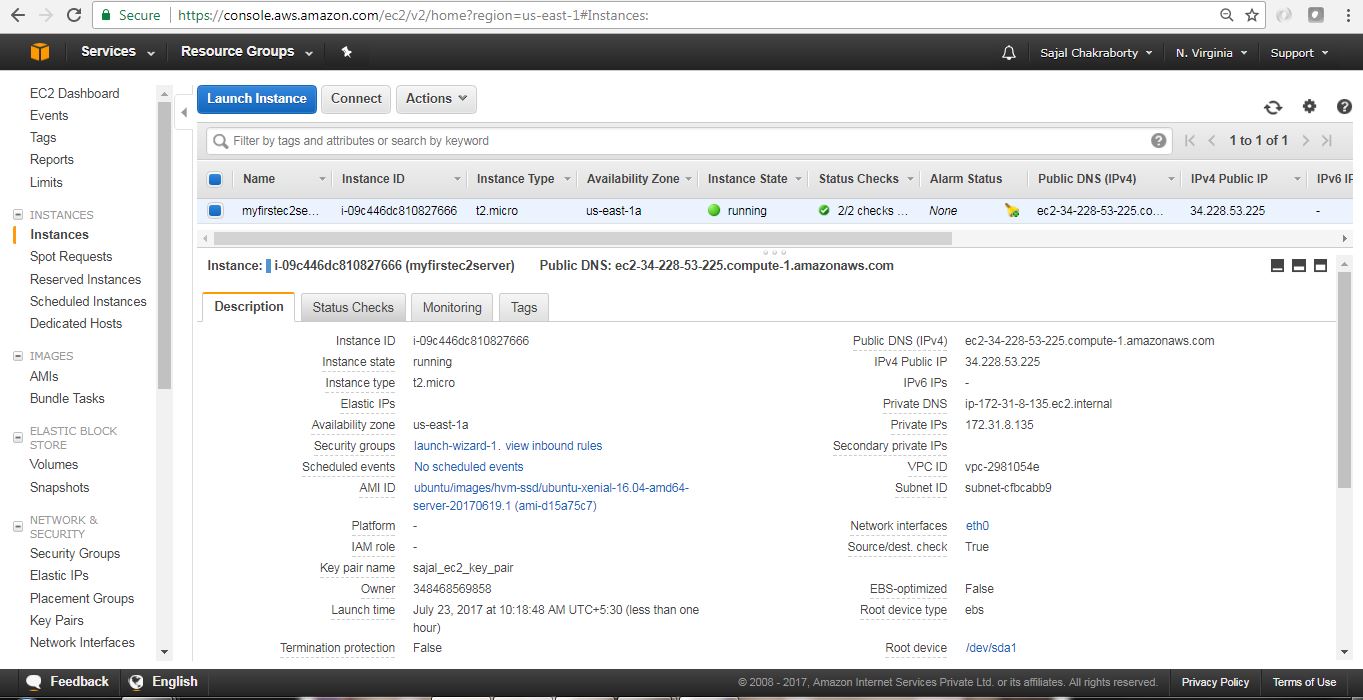
Once we proceed, we will have to select the Key pair which will be used for authentication while connecting with the instance.

1. **Create a new Key Pair** -Logging in to your virtual server requires a key. We use a key instead of a password to authenticate. We will create a new Key pair by choosing appropriate option from drop down and will have to provide a name of the key-pair file and then we need to download that file by clicking the Download Key Pair button in the page. A key is much more secure than a password, and using keys for SSH is mandatory for virtual servers running Linux on AWS. In our case we have chosen Ubuntu Linux, so creating a Key-pair is a must for us here. In this step if we create a new Key-pair, we get one file of type .pem extension. Here are the steps of creating the Key pairs. Instructions are self describing and very easy to follow.[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/EC2_9.jpg)Figure 10.0 : Key Pair Creation Step 1[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_10.jpg)Figure 10.1 : Key Pair Creation Step 2[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_11.jpg)Figure 10.2 : Key Pair Creation Step 3

Once we have created and downloaded the Key pair we need to proceed by clicking Launch Instances button in the **Figure 10.2** page.

1. **Finish Creation Process** – Once we have clicked Launch Instances button in the last page where we have created Key pairs, we will start the actual instance creation step as below.[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_13.jpg)Figure 11.0 : Instance Creating Last step

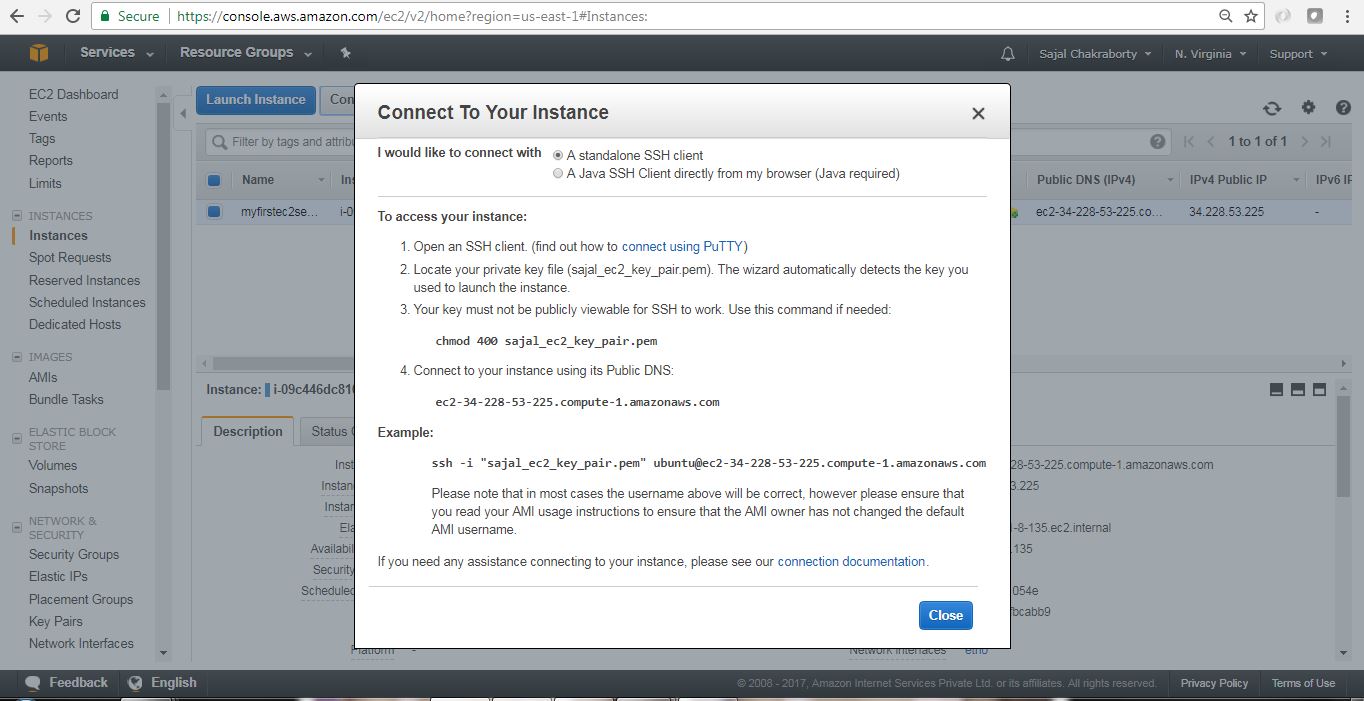
Finally this is the screen where we will see the details of the instance that we have just created.

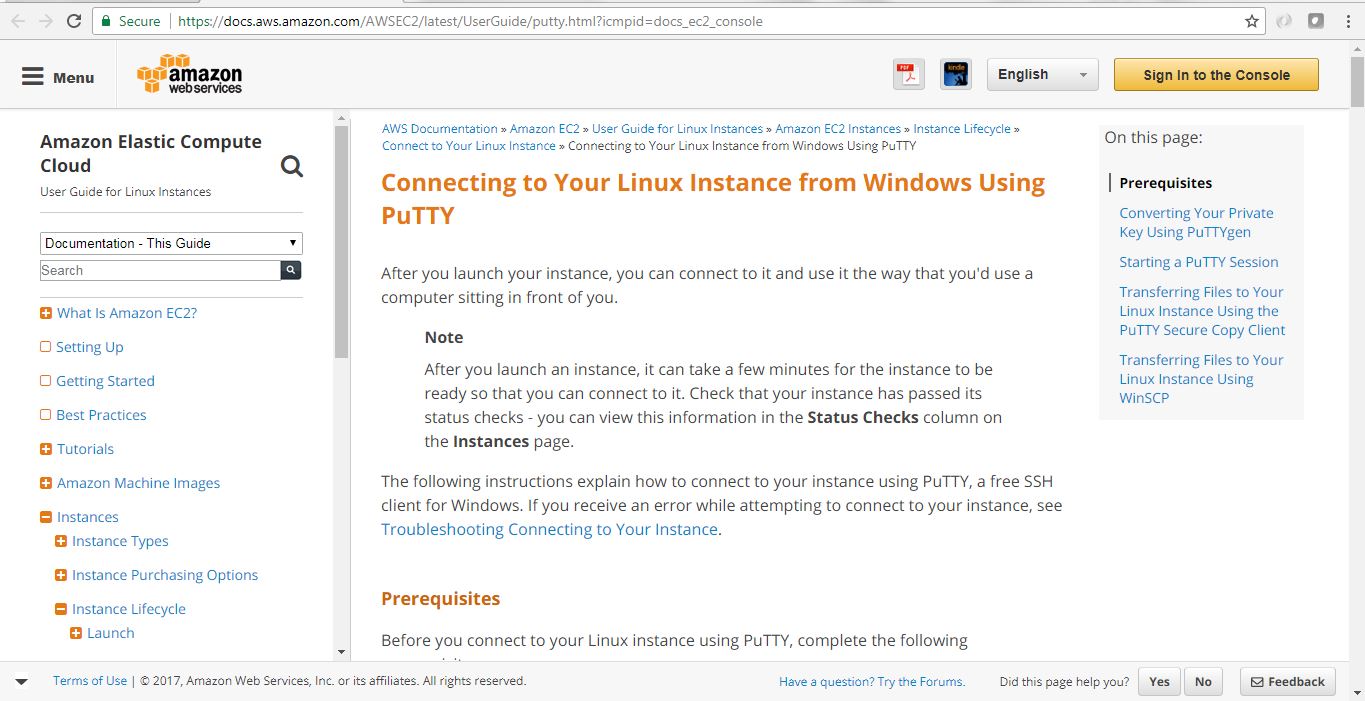
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_14.jpg)Figure 11.1 : Instance Created

Now we have created our first EC2 Instance. Next we will connect to this instance from our local workstation and will install one software (Linkchecker) in EC2 instance and will learn to use that software.

## Connect to EC2 Instance

AWS has provided a very good [documentation](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/putty.html?icmpid=docs_ec2_console) in this regard. Here are the basic steps for configuring and connecting to the instance through putty.

* **AWS provided steps –**In the instance details page, we can click Connect button to view the pop up like this. This will give us the required steps and information regarding connecting to the instance, Also we can go to the actual AWS documentation by going to the link in the pop up – connect using PuTTY link.[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_15.jpg)Connect Pop up Details

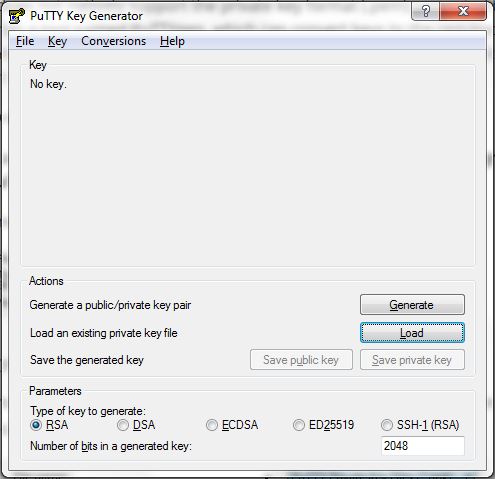
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_16.jpg)Official AWS steps for PuTTY in Windows

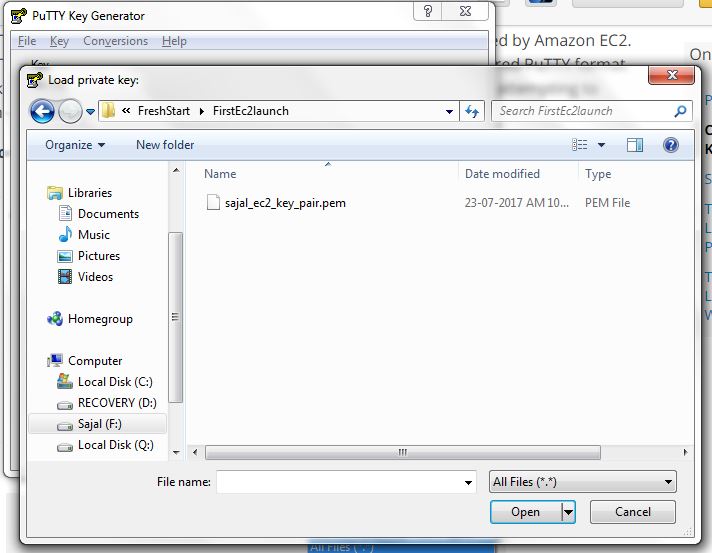
* **Use PuTTYgen to convert the Key File** – Now we need the .pem file we downloaded while creating EC2 instance. Now we will open PuTTYgen app to convert this .pem file to .pkk file which will be used by PuTTY to login to the instance terminal.

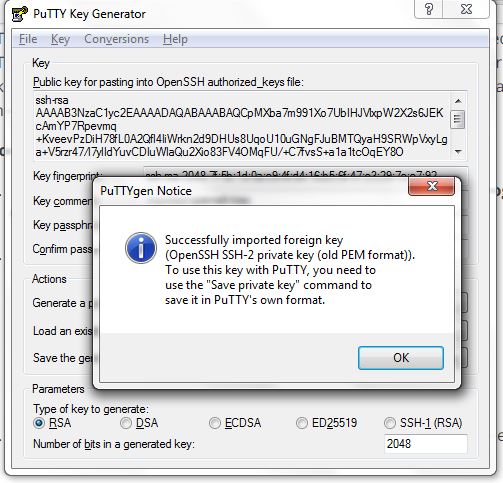
In the PuTTYgen application follow the below step to convert the .pem file.

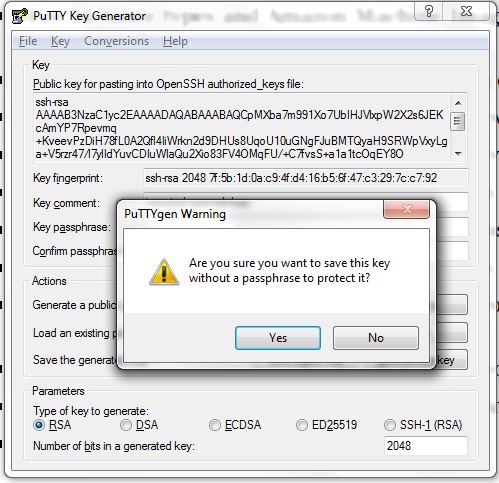
* 1. Run the application PuTTYgen.
  2. Select RSA radio button under Type of Key to Generate.
  3. Click Load.
  4. Because PuTTYgen displays only \*.pkk files, you need to switch the file extension of the File Name field to All Files.
  5. Select the .pem file [in my case it is sajal\_ec2\_key\_pair.pem], and click Open.
  6. Confirm the dialog box.
  7. Click Save Private Key. Ignore the warning about saving the key without a passphrase.

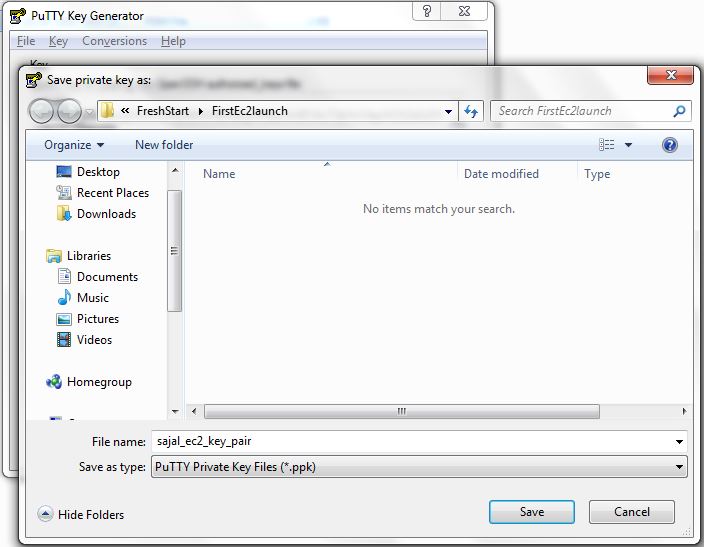
Here are few screen shots for those steps.

[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/putty1.jpg)

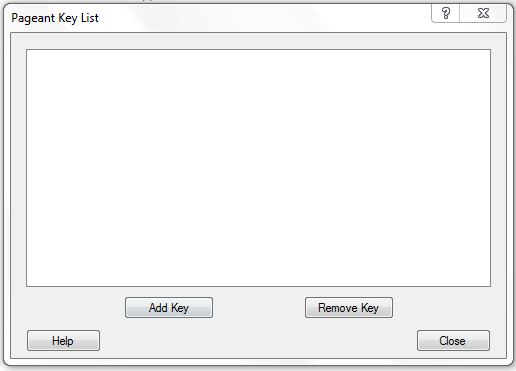
[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/putty2.jpg)

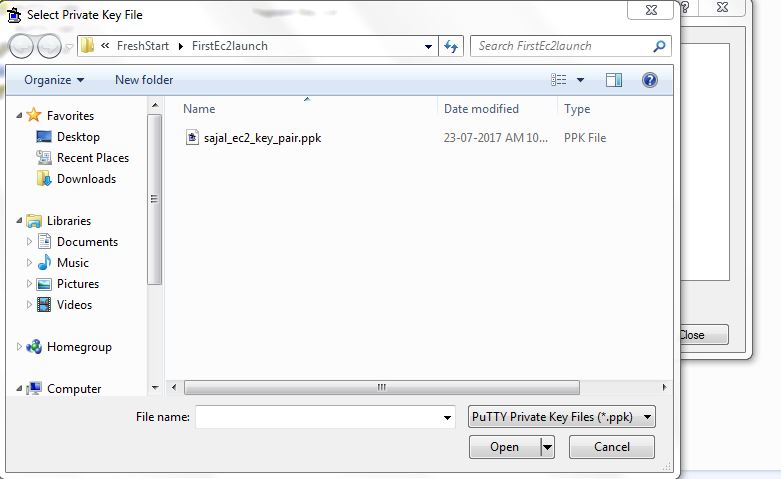
[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/putty3.jpg)

[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/putty4.jpg)

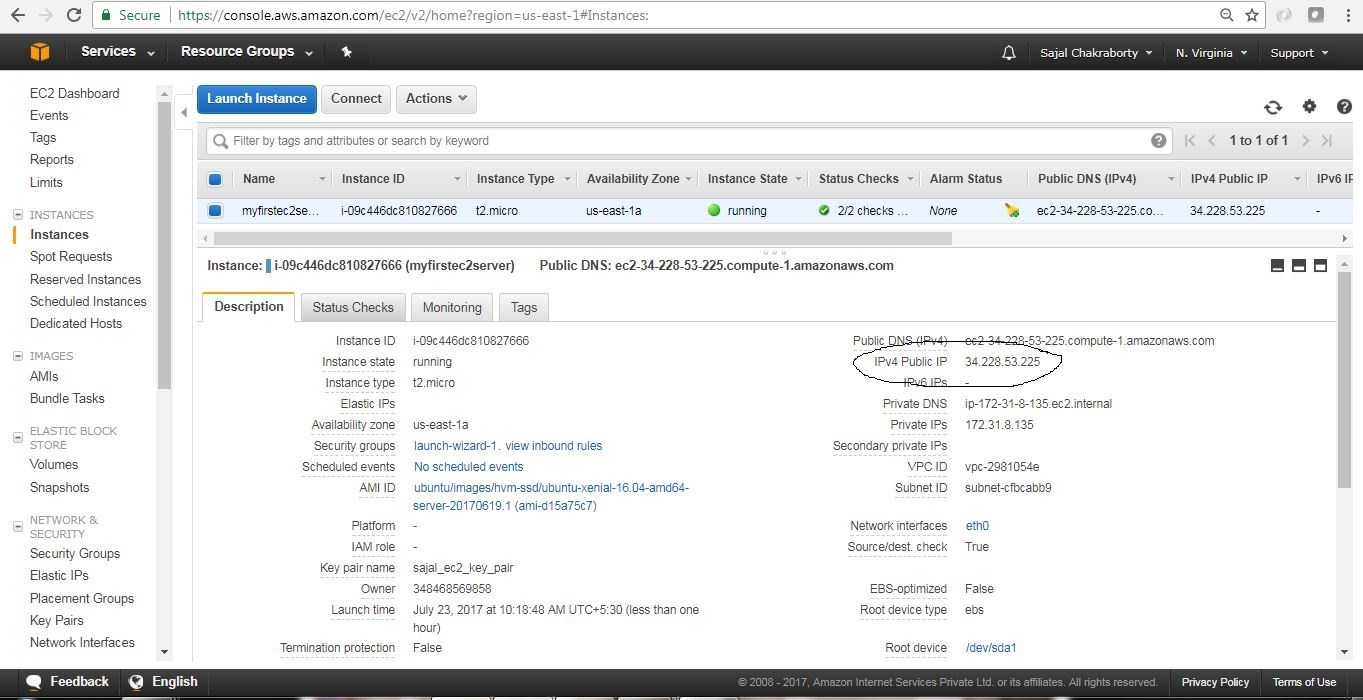
[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/putty5.jpg)

* Start Pageant.EXE and select the .pkk file. Once Pageant stated we will need to add the Key by selecting the .pkk file we have already created. Here are the sample steps for this.

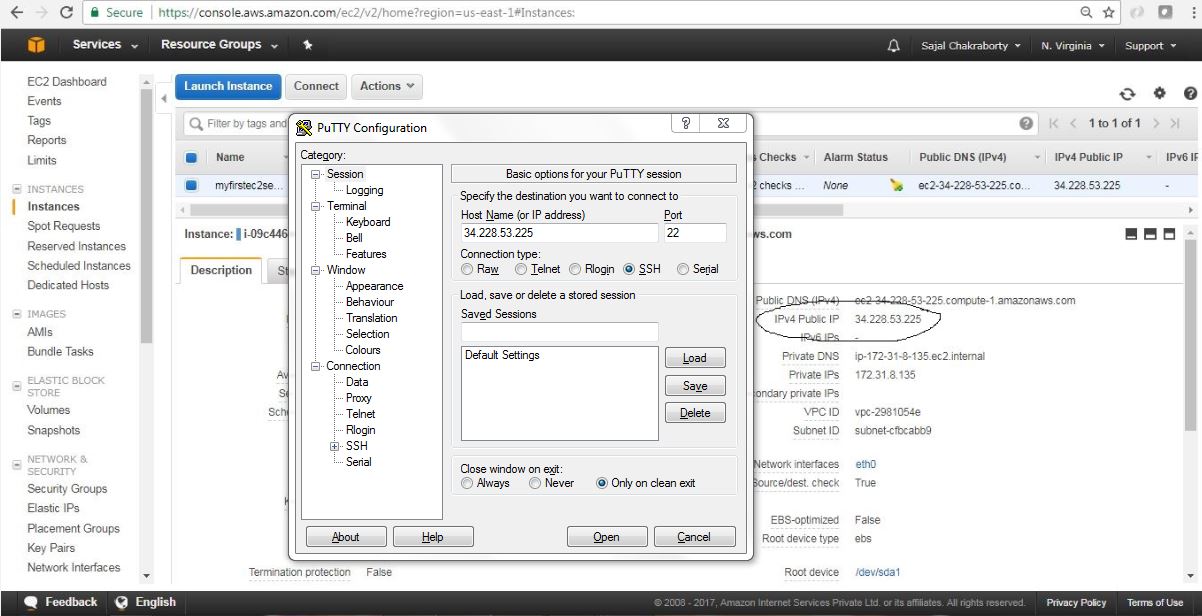
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/PAgent1.jpg)

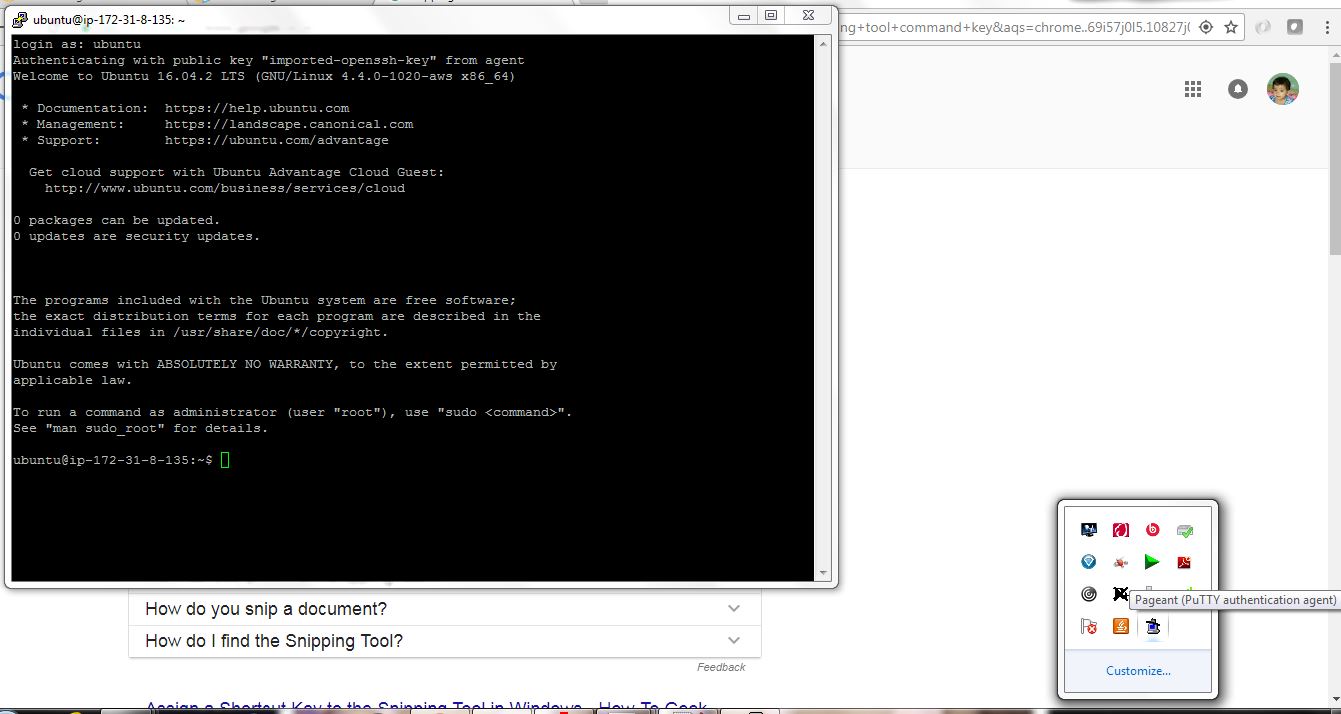
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/PAgent2.jpg)

**Login to EC2 Instance** – Once we have Pageant.exe running and added the key .pkk file, Open Putty.exe and give public IP address of the Instance for connect. We can get the public ip fro the instance details screen in the AWS EC2 page. Now we need to give user name as **ubuntu** in the login as prompt. If everything goes well this would be authenticated and log in to the instance.

[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/EC2_14-1.jpg)

Public IP address for AWS console

[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/Putty-With-Public-IP.jpg)Putty With Public IP

[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/putty7.jpg)Putty Login

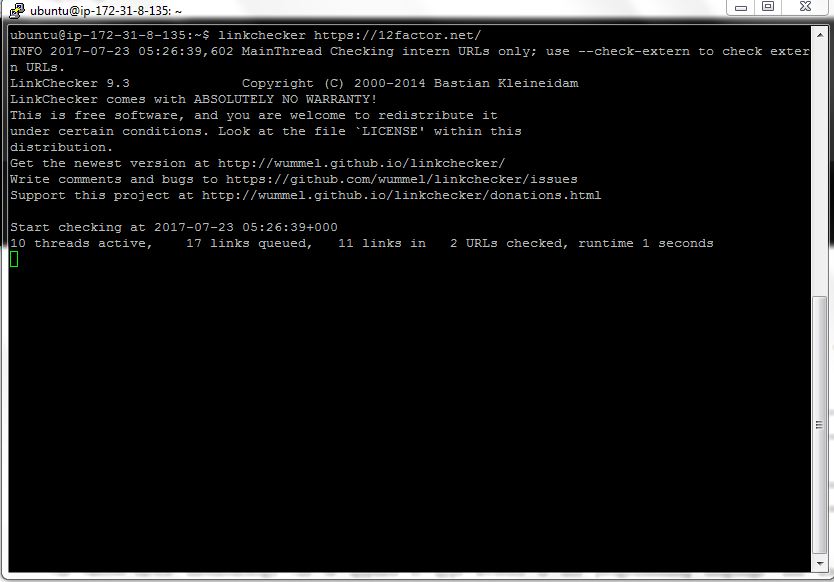
So now we are able to SSH connect to EC2 instance through putty.

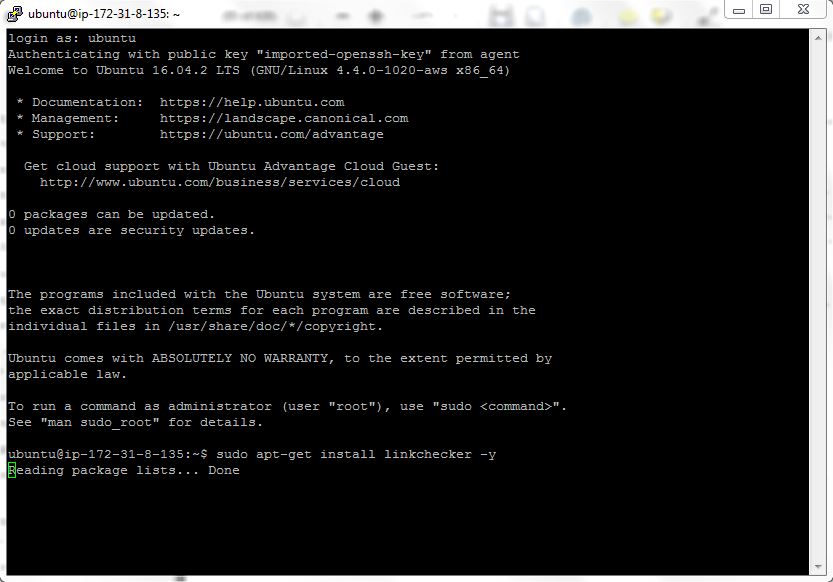
## Installing softwares on EC2 Instance

Now we will install linkchecker software and will use that to do some testing.

* open putty and login ti the terminal as described above and enter the command sudo apt-get install linkchecker -y in the terminal. This will install the linkchecker software in the instance.
* Now test the linkchecker by simply providing some URL like linkchecker https://...

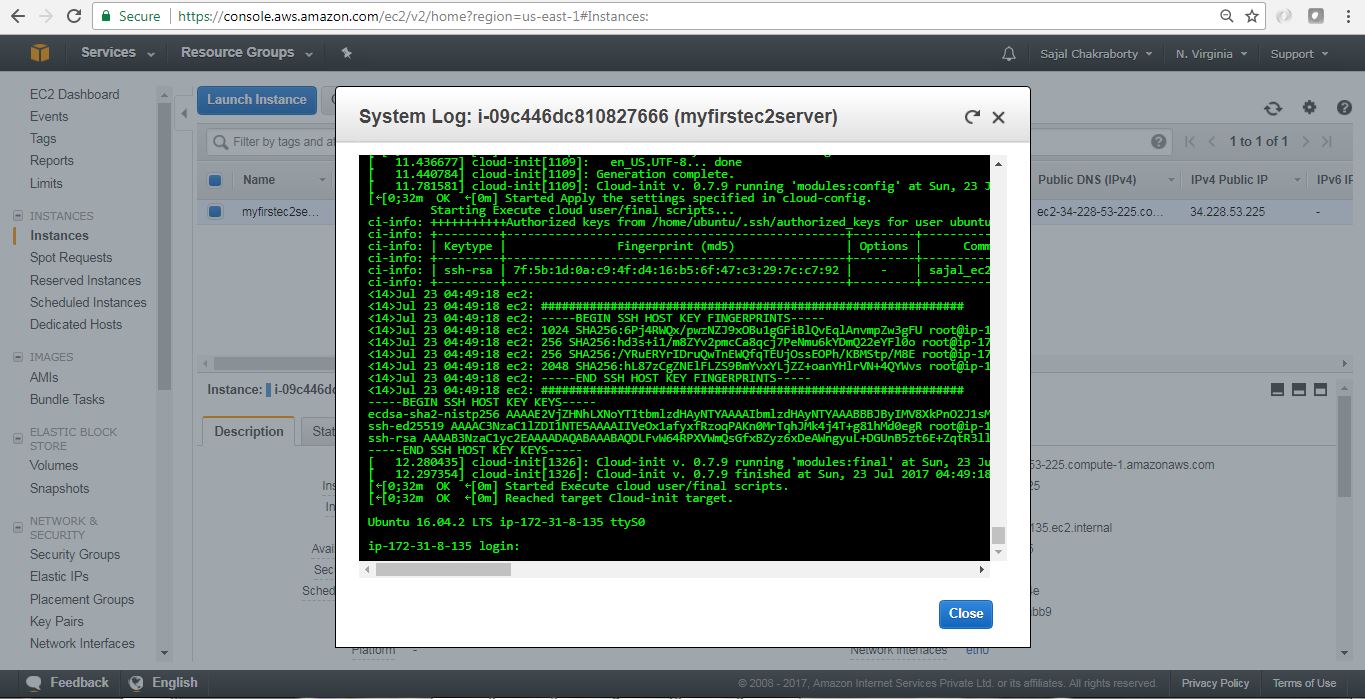
Here is the sample screen related to link checker install and usage.

[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/putty9.jpg)Check 12factor website

[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/putty8.jpg)Install Software

## Checking Logs on EC2 Instance

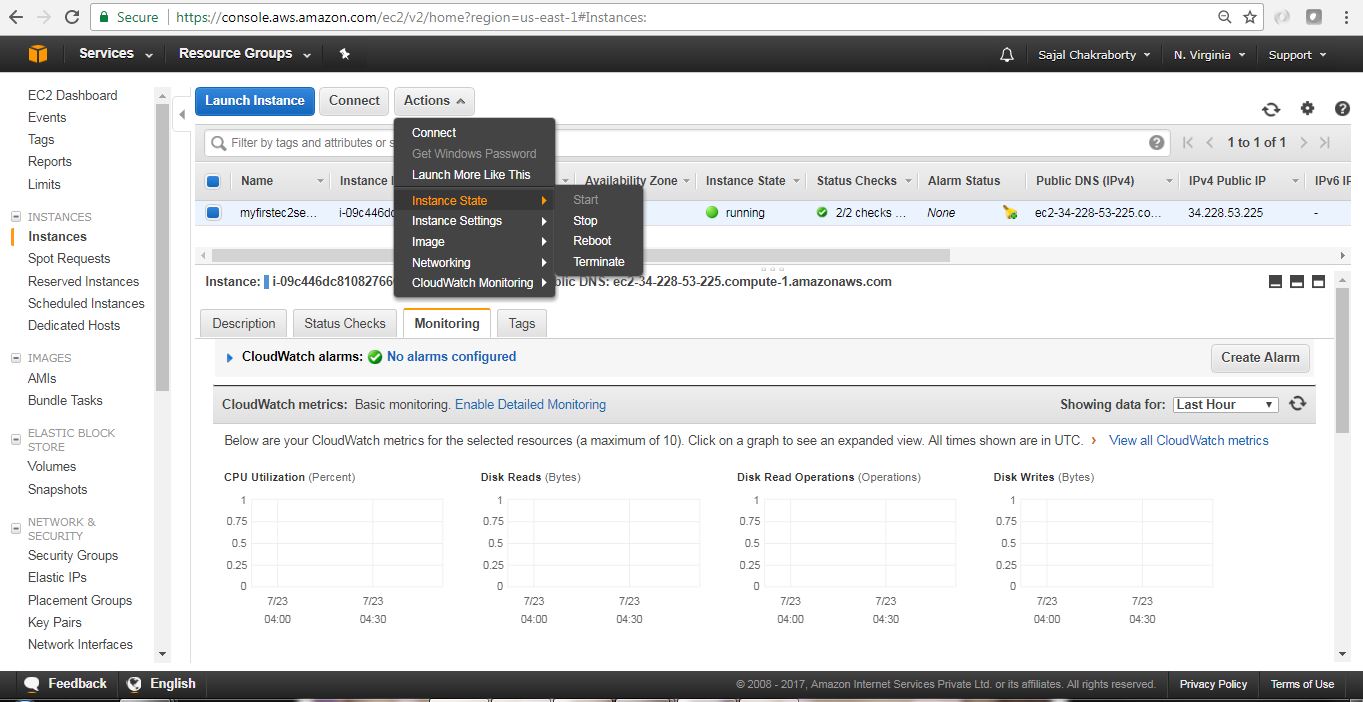
We can view the console log of EC2 instance from the AWS console itself by clicking on the **Actions**menu, choose **Instance Settings > Get System Log**. Here is the sample screen looks like:

[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_viewLogs.jpg)EC2 view System Logs

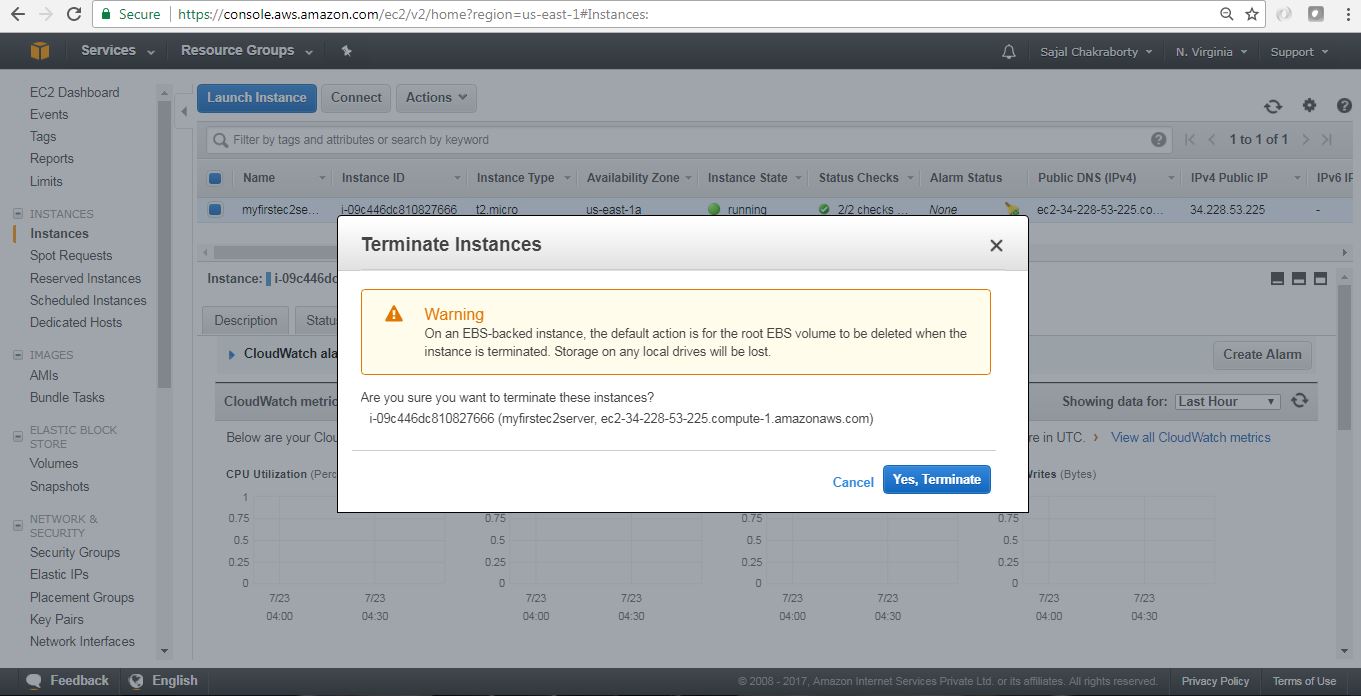
## How to Terminate EC2 Instance

We can terminate EC2 instance by clicking the menu option **Actions > Instance State > Terminate**.

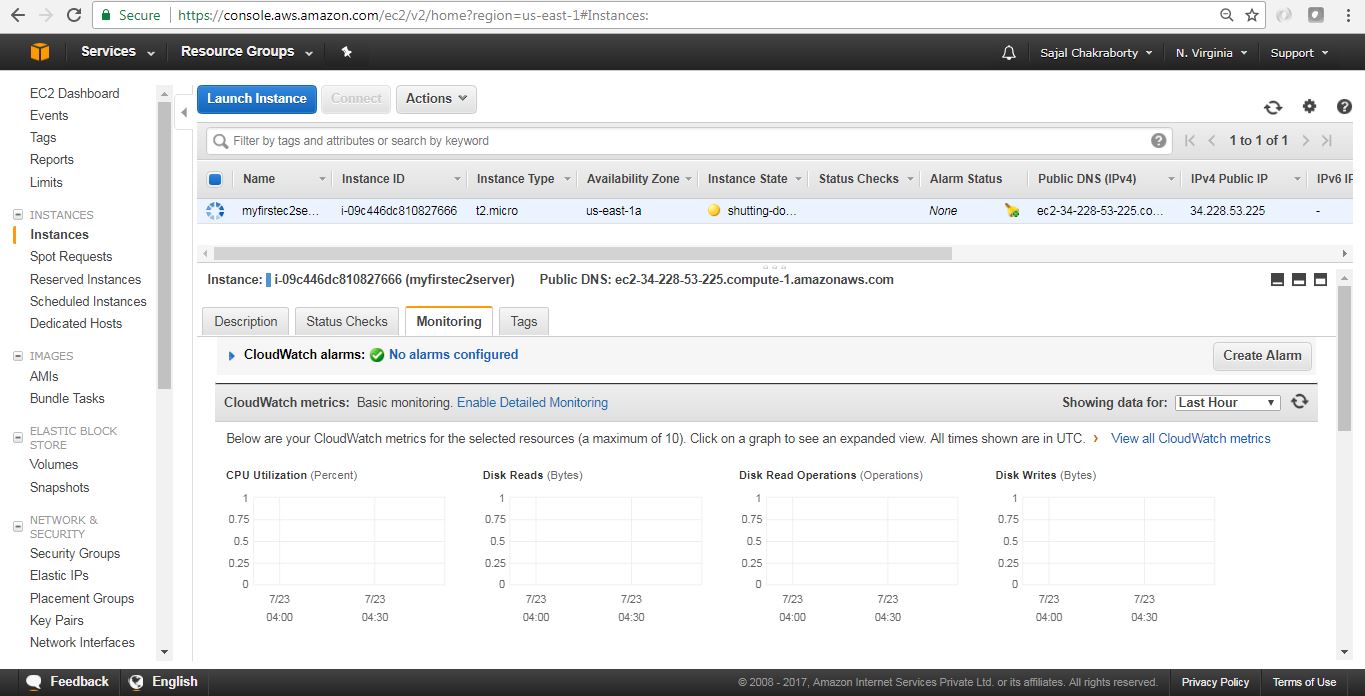
Here are the consequent steps related to termination of the instance. Please note that termination means Deletion/removal of the instance from AWS system. Also we can stop the instance. Please choose accordingly.

[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/EC2_Terminate.jpg)

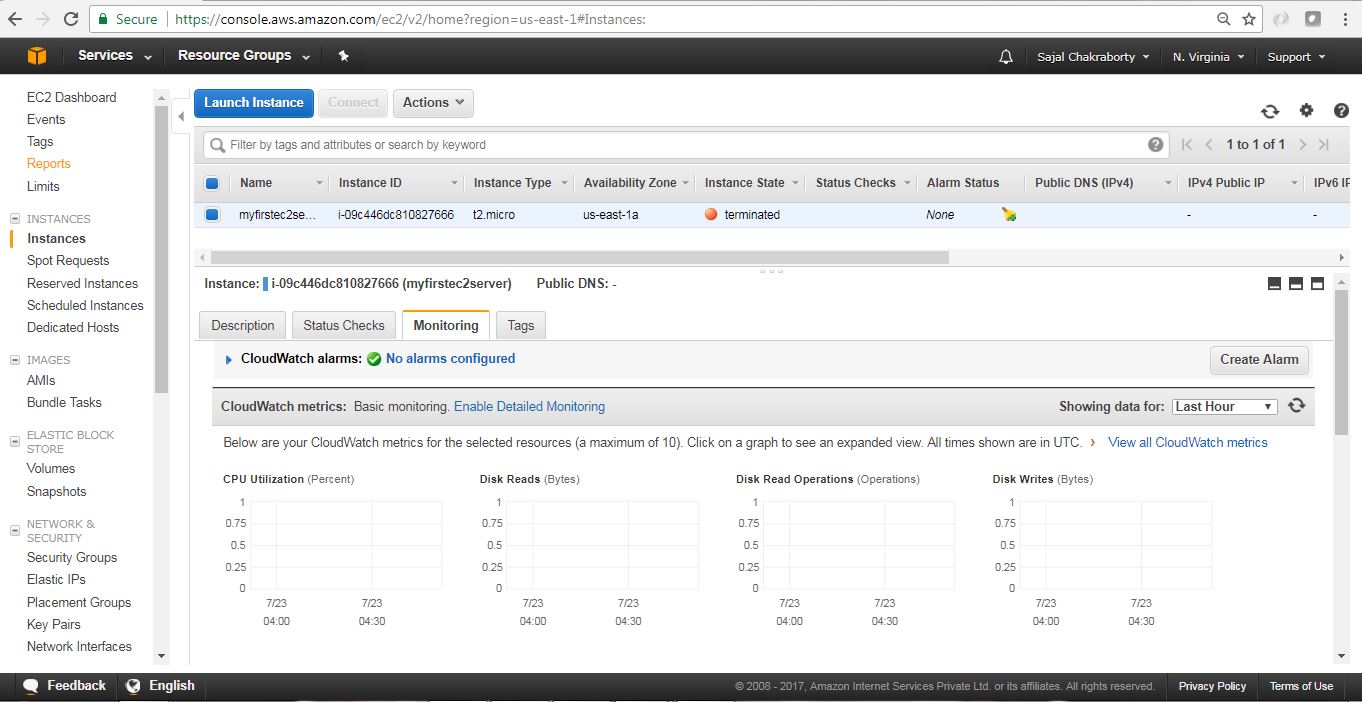
EC2 Instance Termination

[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/EC2_Terminate1.jpg)

EC2 Instance Termination

[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/EC2_Terminate2.jpg)

EC2 Instance Termination

[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/EC2_Terminate3.jpg)EC2 Instance Termination

## Summary

So we have seen how we can spin up one instance in the AWS within the free tire, connected with the instance through SSL client and also installed/used one sample software to get the feel of it. We have also checked how we can see the current log generated in the instance. Last but not the least don’t forget to terminate the instance to avoid cost.

# Amazon Elastic Block Store (Amazon EBS)

Amazon Elastic Block Store (Amazon EBS) provides block level storage volumes for use with EC2 instances. EBS volumes are highly available and reliable storage volumes that can be attached to any running instance that is in the same Availability Zone. EBS volumes that are attached to an EC2 instance are exposed as storage volumes that persist independently from the life of the instance. With Amazon EBS, you pay only for what you use. For more information about Amazon EBS pricing, see the Projecting Costs section of the [Amazon Elastic Block Store page](https://aws.amazon.com/ebs/).

Amazon EBS is recommended when data must be quickly accessible and requires long-term persistence. EBS volumes are particularly well-suited for use as the primary storage for file systems, databases, or for any applications that require fine granular updates and access to raw, unformatted, block-level storage. Amazon EBS is well suited to both database-style applications that rely on random reads and writes, and to throughput-intensive applications that perform long, continuous reads and writes.

For simplified data encryption, you can launch your EBS volumes as encrypted volumes. Amazon EBS encryption offers you a simple encryption solution for your EBS volumes without the need for you to build, manage, and secure your own key management infrastructure. When you create an encrypted EBS volume and attach it to a supported instance type, data stored at rest on the volume, disk I/O, and snapshots created from the volume are all encrypted. The encryption occurs on the servers that hosts EC2 instances, providing encryption of data-in-transit from EC2 instances to EBS storage. For more information, see [Amazon EBS Encryption](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html).

Amazon EBS encryption uses AWS Key Management Service (AWS KMS) master keys when creating encrypted volumes and any snapshots created from your encrypted volumes. The first time you create an encrypted EBS volume in a region, a default master key is created for you automatically. This key is used for Amazon EBS encryption unless you select a Customer Master Key (CMK) that you created separately using the AWS Key Management Service. Creating your own CMK gives you greater flexibility when defining access controls, including the ability to create, rotate, disable, and audit encryption keys that are specific to individual applications and users. For more information, see the [AWS Key Management Service Developer Guide](https://docs.aws.amazon.com/kms/latest/developerguide/).

You can attach multiple volumes to the same instance within the limits specified by your AWS account. Your account has a limit on the number of EBS volumes that you can use, and the total storage available to you. For more information about these limits, and how to request an increase in your limits, see [Request to Increase the Amazon EBS Volume Limit](https://console.aws.amazon.com/support/home#/case/create?issueType=service-limit-increase&limitType=service-code-ebs).

**Contents**

* [Features of Amazon EBS](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AmazonEBS.html#ebs-features)
* [Amazon EBS Volumes](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumes.html)
* [Amazon EBS Snapshots](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSSnapshots.html)
* [Amazon EBS–Optimized Instances](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSOptimized.html)
* [Amazon EBS Encryption](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html)
* [Amazon EBS and NVMe](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/nvme-ebs-volumes.html)
* [Amazon EBS Volume Performance on Linux Instances](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSPerformance.html)
* [Amazon CloudWatch Events for Amazon EBS](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-cloud-watch-events.html)

## Features of Amazon EBS

* You can create EBS General Purpose SSD (gp2), Provisioned IOPS SSD (io1), Throughput Optimized HDD (st1), and Cold HDD (sc1) volumes up to 16 TiB in size. You can mount these volumes as devices on your Amazon EC2 instances. You can mount multiple volumes on the same instance, but each volume can be attached to only one instance at a time. You can dynamically change the configuration of a volume attached to an instance. For more information, see [Creating an Amazon EBS Volume](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-volume.html).
* With General Purpose SSD (gp2) volumes, you can expect base performance of 3 IOPS/GiB, with the ability to burst to 3,000 IOPS for extended periods of time. Gp2 volumes are ideal for a broad range of use cases such as boot volumes, small and medium-size databases, and development and test environments. Gp2 volumes support up to 16,000 IOPS and 250 MiB/s of throughput. For more information, see [General Purpose SSD (gp2) Volumes](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html#EBSVolumeTypes_gp2).
* With Provisioned IOPS SSD (io1) volumes, you can provision a specific level of I/O performance. Io1 volumes support up to 64,000 IOPS and 1,000 MB/s of throughput. This allows you to predictably scale to tens of thousands of IOPS per EC2 instance. For more information, see [Provisioned IOPS SSD (io1) Volumes](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html#EBSVolumeTypes_piops).
* Throughput Optimized HDD (st1) volumes provide low-cost magnetic storage that defines performance in terms of throughput rather than IOPS. With throughput of up to 500 MiB/s, this volume type is a good fit for large, sequential workloads such as Amazon EMR, ETL, data warehouses, and log processing. For more information, see [Throughput Optimized HDD (st1) Volumes](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html#EBSVolumeTypes_st1).
* Cold HDD (sc1) volumes provide low-cost magnetic storage that defines performance in terms of throughput rather than IOPS. With throughput of up to 250 MiB/s, sc1 is a good fit ideal for large, sequential, cold-data workloads. If you require infrequent access to your data and are looking to save costs, sc1 provides inexpensive block storage. For more information, see [Cold HDD (sc1) Volumes](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html#EBSVolumeTypes_sc1).
* EBS volumes behave like raw, unformatted block devices. You can create a file system on top of these volumes, or use them in any other way you would use a block device (like a hard drive). For more information on creating file systems and mounting volumes, see [Making an Amazon EBS Volume Available for Use on Linux](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-using-volumes.html).
* You can use encrypted EBS volumes to meet a wide range of data-at-rest encryption requirements for regulated/audited data and applications. For more information, see [Amazon EBS Encryption](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html).
* You can create point-in-time snapshots of EBS volumes, which are persisted to Amazon S3. Snapshots protect data for long-term durability, and they can be used as the starting point for new EBS volumes. The same snapshot can be used to instantiate as many volumes as you wish. These snapshots can be copied across AWS regions. For more information, see [Amazon EBS Snapshots](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSSnapshots.html).
* EBS volumes are created in a specific Availability Zone, and can then be attached to any instances in that same Availability Zone. To make a volume available outside of the Availability Zone, you can create a snapshot and restore that snapshot to a new volume anywhere in that region. You can copy snapshots to other regions and then restore them to new volumes there, making it easier to leverage multiple AWS regions for geographical expansion, data center migration, and disaster recovery. For more information, see [Creating an Amazon EBS Snapshot](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html), [Restoring an Amazon EBS Volume from a Snapshot](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-restoring-volume.html), and [Copying an Amazon EBS Snapshot](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-copy-snapshot.html).
* A large repository of public data set snapshots can be restored to EBS volumes and seamlessly integrated into AWS cloud-based applications. For more information, see [Using Public Data Sets](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-public-data-sets.html).
* Performance metrics, such as bandwidth, throughput, latency, and average queue length, are available through the AWS Management Console. These metrics, provided by Amazon CloudWatch, allow you to monitor the performance of your volumes to make sure that you are providing enough performance for your applications without paying for resources you don't need. For more information, see [Amazon EBS Volume Performance on Linux Instances](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSPerformance.html).

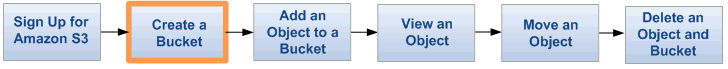
# What Is Amazon S3?

Amazon Simple Storage Service is storage for the Internet. It is designed to make web-scale computing easier for developers.

Amazon S3 has a simple web services interface that you can use to store and retrieve any amount of data, at any time, from anywhere on the web. It gives any developer access to the same highly scalable, reliable, fast, inexpensive data storage infrastructure that Amazon uses to run its own global network of web sites. The service aims to maximize benefits of scale and to pass those benefits on to developers.

This guide explains the core concepts of Amazon S3, such as buckets and objects, and how to work with these resources using the Amazon S3 application programming interface (API).

# Create a Bucket



Now that you've signed up for Amazon S3, you're ready to create a bucket using the AWS Management Console. Every object in Amazon S3 is stored in a bucket. Before you can store data in Amazon S3, you must create a bucket.

**Note**

You are not charged for creating a bucket; you are charged only for storing objects in the bucket and for transferring objects in and out of the bucket. The charges you will incur through following the examples in this guide are minimal (less than $1). For more information about storage charges, see [Amazon S3 Pricing](https://aws.amazon.com/s3/pricing/).

**To create an S3 bucket**

1. Sign in to the AWS Management Console and open the Amazon S3 console at<https://console.aws.amazon.com/s3/>.
2. Choose **Create bucket**.


          Choose Create bucket.
        

1. In the **Bucket name** field, type a unique DNS-compliant name for your new bucket. (The example screen shot uses the bucket name admin-created. You cannot use this name because S3 bucket names must be unique.)

4.Create your own bucket name using the follow naming guidelines:

* + The name must be unique across all existing bucket names in Amazon S3.
  + After you create the bucket you cannot change the name, so choose wisely.
  + Choose a bucket name that reflects the objects in the bucket because the bucket name is visible in the URL that points to the objects that you're going to put in your bucket.

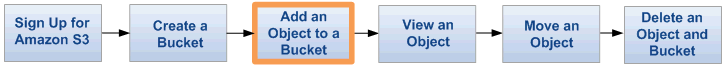
For information about naming buckets, see [Rules for Bucket Naming](https://docs.aws.amazon.com/AmazonS3/latest/dev/BucketRestrictions.html#bucketnamingrules) in the Amazon Simple Storage Service Developer Guide.

1. For **Region**, choose US West (Oregon) as the region where you want the bucket to reside.
2. Choose **Create**.


          Create an S3 bucket page.
        

You've created a bucket in Amazon S3.

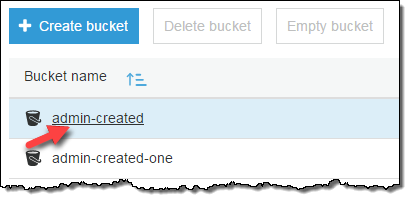
# Add an Object to a Bucket



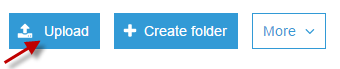
Now that you've created a bucket, you're ready to add an object to it. An object can be any kind of file: a text file, a photo, a video, and so on.

**To upload an object to a bucket**

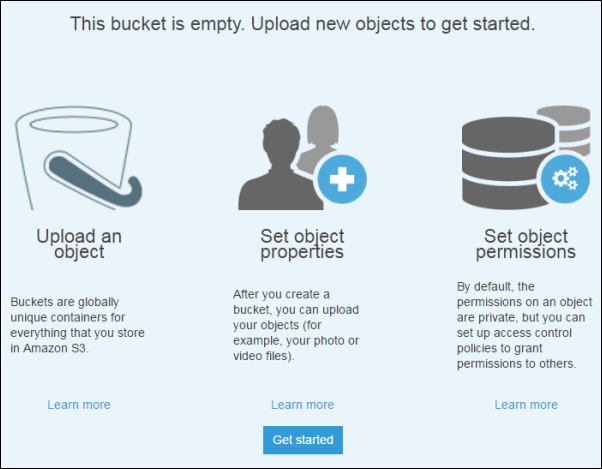
1. In the **Bucket name** list, choose the name of the bucket that you want to upload your object to.



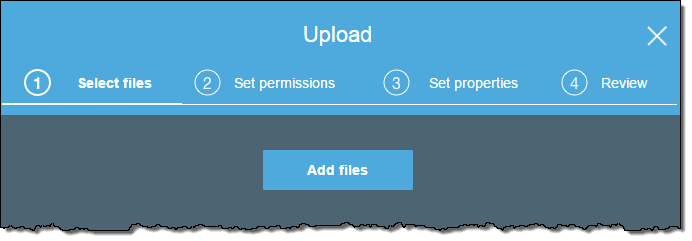
1. Choose **Upload**.



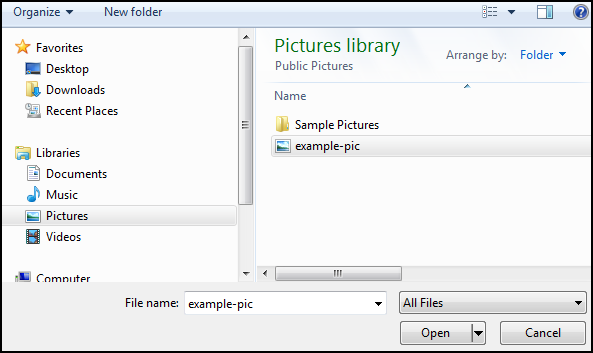
* 1. Or you can choose **Get started**.



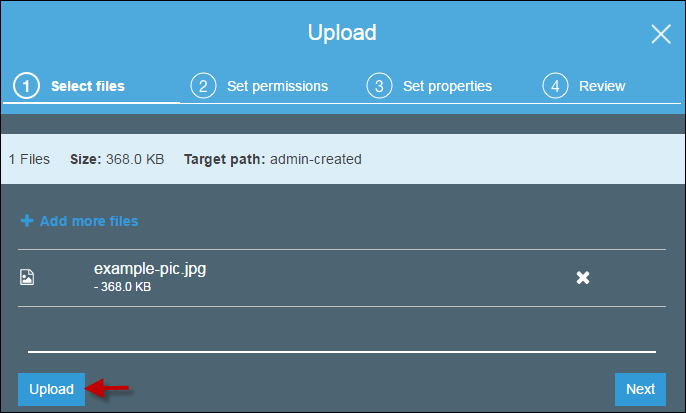
1. In the **Upload** dialog box, choose **Add files** to choose the file to upload.



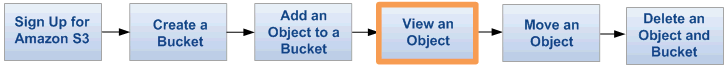
1. Choose a file to upload, and then choose **Open.**



1. Choose **Upload**.



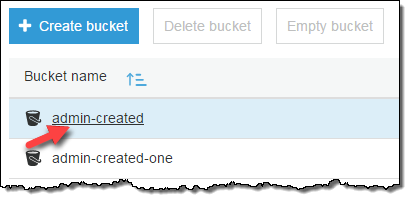
# View an Object



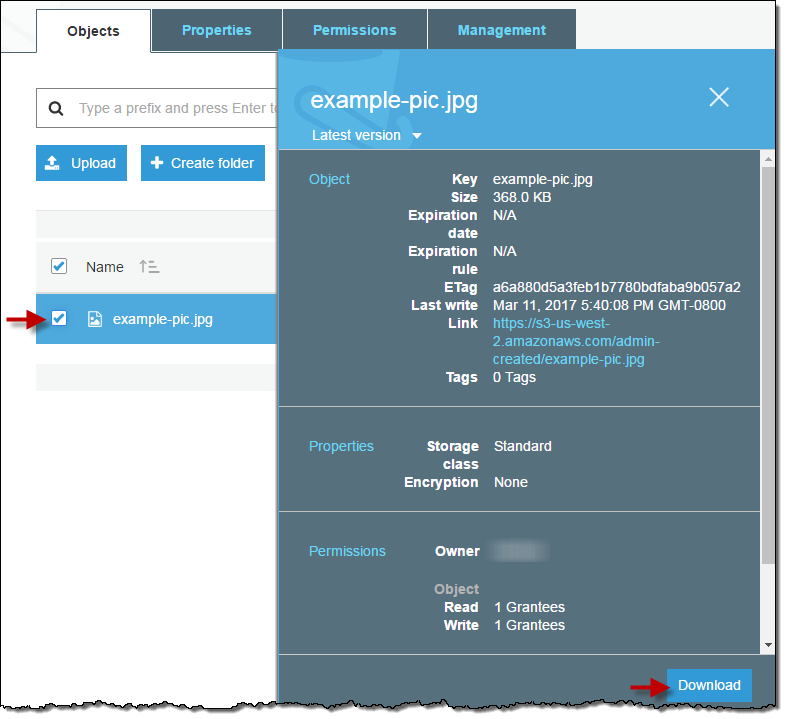
Now that you've added an object to a bucket, you can view information about your object and download the object to your local computer.

**To download an object from a bucket**

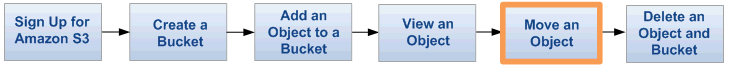
1. In the **Bucket name** list, choose the name of the bucket that you created.



1. In the **Name** list, select the check box next to the object that you uploaded, and then choose **Download** on the object overview panel.



# Move an Object



So far you've added an object to a bucket and downloaded the object. Now we create a folder and move the object into the folder by copying and pasting the object.

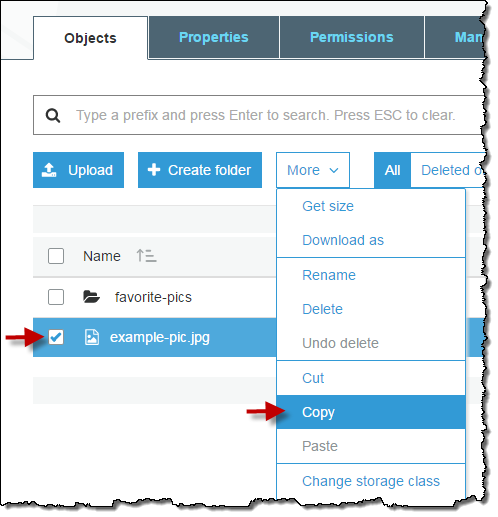
**To copy an object**

1. In the **Bucket name** list, choose the name of the bucket that you created.

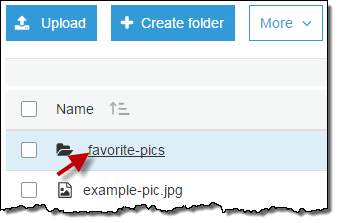

          Type the name of the folder in the Amazon S3 console.
        
          Choose the name of the bucket that you created in the Amazon S3 console.
        

1. Choose **Create Folder**, type **favorite-pics** for the folder name, choose **None** for the encryption setting for the folder object and then choose **Save**.

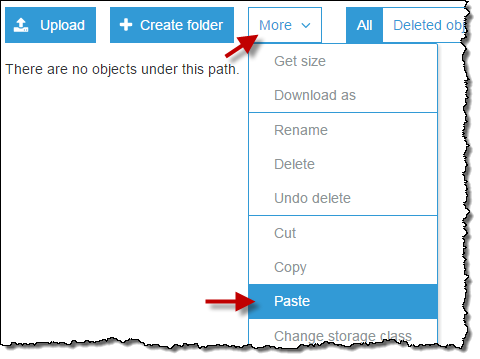
1. In the **Name** list, select the check box next to the object that you want to copy, choose **More**, and then choose **Copy**.



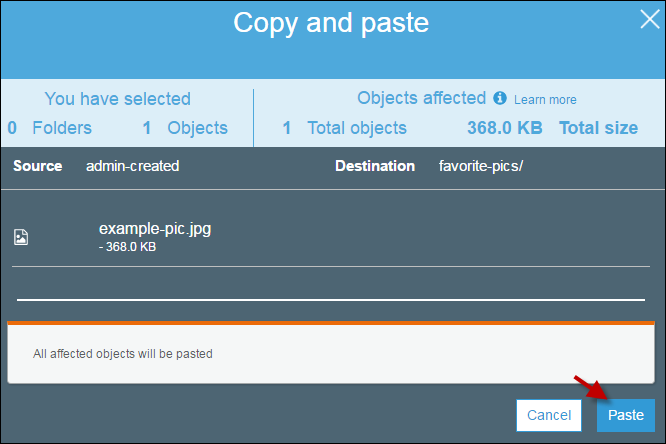
1. In the **Name** list, choose the name of the folder **favorite-pics**.



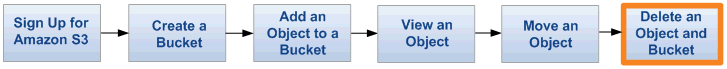
1. Choose **More**, and then choose **Paste**.



1. Choose **Paste**.



# Delete an Object and Bucket



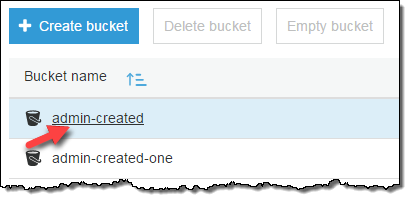
If you no longer need to store the object that you uploaded and made a copy of while going through this guide, you should delete the objects to prevent further charges.

You can delete the objects individually. Or you can empty a bucket, which deletes all the objects in the bucket without deleting the bucket.

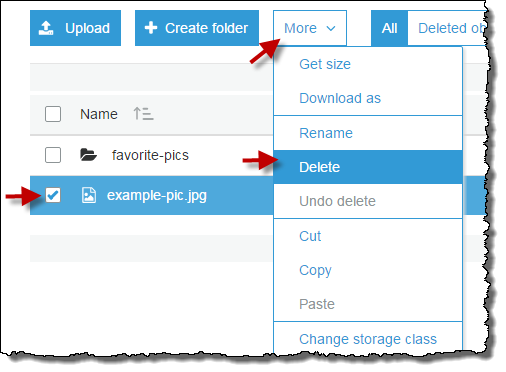
You can also delete a bucket and all the objects contained in the bucket. However, if you want to continue to use the same bucket name, don't delete the bucket. We recommend that you empty the bucket and keep it. After a bucket is deleted, the name becomes available to reuse, but the name might not be available for you to reuse for various reasons. For example, it might take some time before the name can be reused and some other account could create a bucket with that name before you do.

**To delete an object from a bucket**

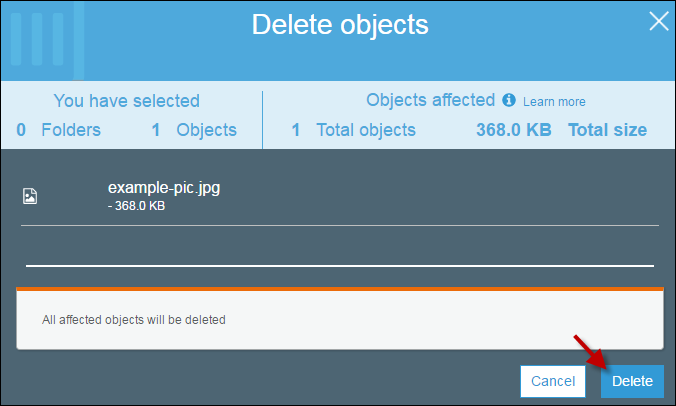
1. In the **Bucket name** list, choose the name of the bucket that you want to delete an object from.



1. In the **Name** list, select the check box next to the object that you want to delete, choose **More**, and then choose **Delete**.



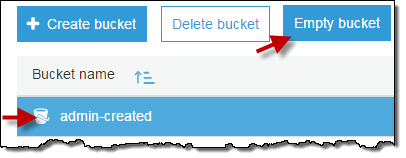
1. In the **Delete objects** dialog box, verify that the name of the object you selected for deletion is listed, and then choose **Delete**.



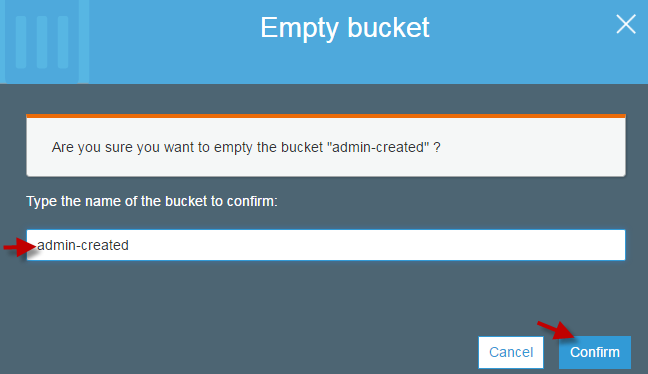
You can empty a bucket, which deletes all the objects in the bucket without deleting the bucket.

**To empty a bucket**

1. In the **Bucket name** list, choose the bucket icon next to the name of the bucket that you want to empty and then choose **Empty bucket**.



1. In the **Empty bucket** dialog box, type the name of the bucket for confirmation and then choose **Confirm**.



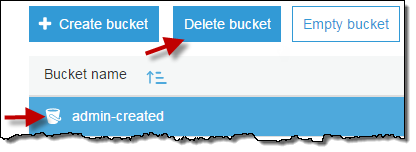
You can delete a bucket and all the objects contained in the bucket.

**Important**

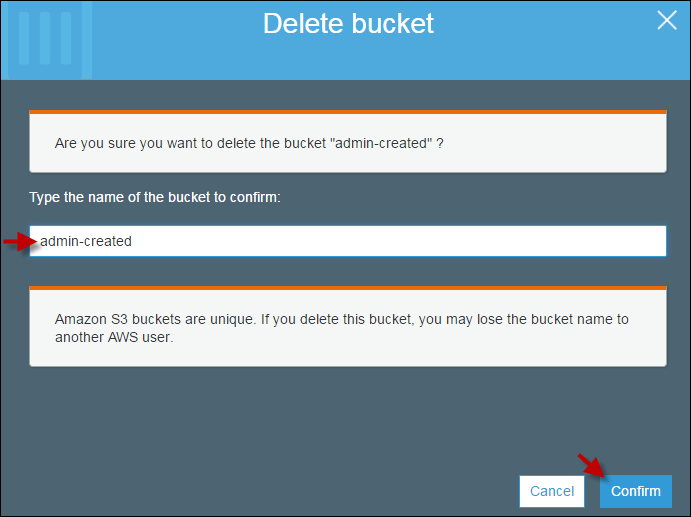
If you want to continue to use the same bucket name, don't delete the bucket. We recommend that you empty the bucket and keep it. After a bucket is deleted, the name becomes available to reuse, but the name might not be available for you to reuse for various reasons.

**To delete a bucket**

1. In the **Bucket name** list, choose the bucket icon next to the name of the bucket that you want to delete and then choose **Delete bucket**.



1. In the **Delete bucket** dialog box, type the name of the bucket for delete confirmation and then choose **Confirm**.



# What Is Amazon S3 Glacier?

Welcome to the Amazon S3 Glacier Developer Guide. Amazon Simple Storage Service Glacier, that is Amazon S3 Glacier (Glacier), is a storage service optimized for infrequently used data, or "cold data."

Glacier is an extremely low-cost storage service that provides durable storage with security features for data archiving and backup. With Glacier, customers can store their data cost effectively for months, years, or even decades. Glacier enables customers to offload the administrative burdens of operating and scaling storage to AWS, so they don't have to worry about capacity planning, hardware provisioning, data replication, hardware failure detection and recovery, or time-consuming hardware migrations. For more service highlights and pricing information, go to the [Glacier detail page](https://aws.amazon.com/glacier).

**Topics**

* [Are You a First-Time Glacier User?](https://docs.aws.amazon.com/amazonglacier/latest/dev/introduction.html#are-you-a-firsttime-glacier-user)
* [Amazon S3 Glacier Data Model](https://docs.aws.amazon.com/amazonglacier/latest/dev/amazon-glacier-data-model.html)
* [Supported Operations in Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/amazon-glacier-supported-operations.html)
* [Accessing Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/amazon-glacier-accessing.html)

## Are You a First-Time Glacier User?

If you are a first-time user of Glacier, we recommend that you begin by reading the following sections:

* **What is Glacier—**The rest of this section describes the underlying data model, the operations it supports, and the AWS SDKs that you can use to interact with the service.
* **Getting Started—**The [Getting Started with Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/amazon-glacier-getting-started.html) section walks you through the process of creating a vault, uploading archives, creating jobs to download archives, retrieving the job output, and deleting archives.

**Important**

Glacier provides a console, which you can use to create and delete vaults. However, all other interactions with Glacier require that you use the AWS Command Line Interface (AWS CLI) or write code. For example, to upload data, such as photos, videos, and other documents, you must either use the AWS CLI or write code to make requests, by using either the REST API directly or by using the AWS SDKs. For more information about using Glacier with the AWS CLI, go to [AWS CLI Reference for Glacier](http://docs.aws.amazon.com/cli/latest/reference/glacier/index.html). To install the AWS CLI, go to [AWS Command Line Interface](http://aws.amazon.com/cli/).

Beyond the getting started section, you'll probably want to learn more about Glacier operations. The following sections provide detailed information about working with Glacier using the REST API and the AWS Software Development Kits (SDKs) for Java and Microsoft .NET:

* [Using the AWS SDKs with Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/using-aws-sdk.html)

This section provides an overview of the AWS SDKs used in various code examples in this guide. A review of this section will help when reading the following sections. It includes an overview of the high-level and the low-level APIs that these SDKs offer, when to use them, and common steps for running the code examples provided in this guide.

* [Working with Vaults in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/working-with-vaults.html)

This section provides details of various vault operations such as creating a vault, retrieving vault metadata, using jobs to retrieve vault inventory, and configuring vault notifications. In addition to using the Glacier console, you can use the AWS SDKs for various vault operations. This section describes the API and provides working samples using the AWS SDK for Java and .NET.

* [Working with Archives in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/working-with-archives.html)

This section provides details of archive operations such as uploading an archive in a single request or using a multipart upload operation to upload large archives in parts. The section also explains creating jobs to download archives asynchronously. The section provides examples using the AWS SDK for Java and .NET.

* [API Reference for Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/amazon-glacier-api.html)

Glacier is a RESTful service. This section describes the REST operations, including the syntax, and example requests and responses for all the operations. Note that the AWS SDK libraries wrap this API, simplifying your programming tasks.

Amazon Simple Storage Service (Amazon S3) supports lifecycle configuration on an S3 bucket, which enables you to transition objects to the Amazon S3 GLACIER storage class for archival. When you transition Amazon S3 objects to the GLACIER storage class, Amazon S3 internally uses Glacier for durable storage at lower cost. Although the objects are stored in Glacier, they remain Amazon S3 objects that you manage in Amazon S3, and you cannot access them directly through Glacier.

# Working with Vaults in Amazon S3 Glacier

A vault is a container for storing archives. When you create a vault, you specify a vault name and a region in which you want to create the vault. For a list of supported regions, see [Accessing Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/amazon-glacier-accessing.html).

You can store an unlimited number of archives in a vault.

**Important**

Amazon S3 Glacier (Glacier) provides a management console. You can use the console to create and delete vaults. However, all other interactions with Glacier require that you use the AWS Command Line Interface (CLI) or write code. For example, to upload data, such as photos, videos, and other documents, you must either use the AWS CLI or write code to make requests, using either the REST API directly or by using the AWS SDKs. For more information about using Glacier with the AWS CLI, go to [AWS CLI Reference for Glacier](http://docs.aws.amazon.com/cli/latest/reference/glacier/index.html). To install the AWS CLI, go to [AWS Command Line Interface](http://aws.amazon.com/cli/).

**Topics**

* [Vault Operations in Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/working-with-vaults.html#vault-operations-quick-intro)
* [Creating a Vault in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/creating-vaults.html)
* [Retrieving Vault Metadata in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/retrieving-vault-info.html)
* [Downloading a Vault Inventory in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/vault-inventory.html)
* [Configuring Vault Notifications in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/configuring-notifications.html)
* [Deleting a Vault in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/deleting-vaults.html)
* [Tagging Your Amazon S3 Glacier Vaults](https://docs.aws.amazon.com/amazonglacier/latest/dev/tagging-vaults.html)
* [Amazon S3 Glacier Vault Lock](https://docs.aws.amazon.com/amazonglacier/latest/dev/vault-lock.html)

## Vault Operations in Glacier

Glacier supports various vault operations. Note that vault operations are region specific. For example, when you create a vault, you create it in a specific region. When you list vaults, Glacier returns the vault list from the region you specified in the request.

### Creating and Deleting Vaults

An AWS account can create up to 1,000 vaults per region. For a list of the AWS regions supported by Glacier, see [Regions and Endpoints](https://docs.aws.amazon.com/general/latest/gr/rande.html#glacier_region) in the AWS General Reference.

You can delete a vault only if there are no archives in the vault as of the last inventory that Glacier computed and there have been no writes to the vault since the last inventory.

**Note**

Glacier prepares an inventory for each vault periodically, every 24 hours. Because the inventory might not reflect the latest information, Glacier ensures the vault is indeed empty by checking if there were any write operations since the last vault inventory.

For more information, see [Creating a Vault in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/creating-vaults.html) and [Deleting a Vault in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/deleting-vaults.html).

### Retrieving Vault Metadata

You can retrieve vault information such as the vault creation date, number of archives in the vault, and the total size of all the archives in the vault. Glacier provides API calls for you to retrieve this information for a specific vault or all the vaults in a specific region in your account. For more information, see [Retrieving Vault Metadata in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/retrieving-vault-info.html).

### Downloading a Vault Inventory

A vault inventory refers to the list of archives in a vault. For each archive in the list, the inventory provides archive information such as archive ID, creation date, and size. Glacier updates the vault inventory approximately once a day, starting on the day the first archive is uploaded to the vault. A vault inventory must exist for you to be able to download it.

Downloading a vault inventory is an asynchronous operation. You must first initiate a job to download the inventory. After receiving the job request, Glacier prepares your inventory for download. After the job completes, you can download the inventory data.

Given the asynchronous nature of the job, you can use Amazon Simple Notification Service (Amazon SNS) notifications to notify you when the job completes. You can specify an Amazon SNS topic for each individual job request or configure your vault to send a notification when specific vault events occur.

Glacier prepares an inventory for each vault periodically, every 24 hours. If there have been no archive additions or deletions to the vault since the last inventory, the inventory date is not updated. When you initiate a job for a vault inventory, Glacier returns the last inventory it generated, which is a point-in-time snapshot and not real-time data. You might not find it useful to retrieve vault inventory for each archive upload. However, suppose you maintain a database on the client-side associating metadata about the archives you upload to Glacier. Then, you might find the vault inventory useful to reconcile information in your database with the actual vault inventory.

For more information about retrieving a vault inventory, see [Downloading a Vault Inventory in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/vault-inventory.html).

### Configuring Vault Notifications

Retrieving anything from Glacier, such as an archive from a vault or a vault inventory, is a two-step process in which you first initiate a job. After the job completes, you can download the output. You can use Glacier notifications support to know when your job is complete. Glacier sends notification messages to an Amazon Simple Notification Service (Amazon SNS) topic that you provide.

# Working with Archives in Amazon S3 Glacier

An archive is any object, such as a photo, video, or document, that you store in a vault. It is a base unit of storage in Amazon S3 Glacier (Glacier). Each archive has a unique ID and an optional description. When you upload an archive, Glacier returns a response that includes an archive ID. This archive ID is unique in the region in which the archive is stored. The following is an example archive ID.

TJgHcrOSfAkV6hdPqOATYfp\_0ZaxL1pIBOc02iZ0gDPMr2ig-nhwd\_PafstsdIf6HSrjHnP-3p6LCJClYytFT\_CBhT9CwNxbRaM5MetS3I-GqwxI3Y8QtgbJbhEQPs0mJ3KExample

Archive IDs are 138 bytes long. When you upload an archive, you can provide an optional description. You can retrieve an archive using its ID but not its description.

**Important**

Glacier provides a management console. You can use the console to create and delete vaults. However, all other interactions with Glacier require that you use the AWS Command Line Interface (CLI) or write code. For example, to upload data, such as photos, videos, and other documents, you must either use the AWS CLI or write code to make requests, using either the REST API directly or by using the AWS SDKs. For more information about using Glacier with the AWS CLI, go to [AWS CLI Reference for Glacier](http://docs.aws.amazon.com/cli/latest/reference/glacier/index.html). To install the AWS CLI, go to [AWS Command Line Interface](http://aws.amazon.com/cli/).

**Topics**

* [Archive Operations in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/working-with-archives.html#archive-operations-quick-intro)
* [Maintaining Client-Side Archive Metadata](https://docs.aws.amazon.com/amazonglacier/latest/dev/working-with-archives.html#client-side-key-map-concept)
* [Uploading an Archive in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/uploading-an-archive.html)
* [Downloading an Archive in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/downloading-an-archive.html)
* [Deleting an Archive in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/deleting-an-archive.html)
* [Querying an Archives in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/querying-archives.html)

## Archive Operations in Amazon S3 Glacier

Glacier supports the following basic archive operations: upload, download, and delete. Downloading an archive is an asynchronous operation.

### Uploading an Archive in Amazon S3 Glacier

You can upload an archive in a single operation or upload it in parts. The API call you use to upload an archive in parts is referred as Multipart Upload. For more information, see [Uploading an Archive in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/uploading-an-archive.html).

**Important**

Glacier provides a management console. You can use the console to create and delete vaults. However, all other interactions with Glacier require that you use the AWS Command Line Interface (CLI) or write code. For example, to upload data, such as photos, videos, and other documents, you must either use the AWS CLI or write code to make requests, using either the REST API directly or by using the AWS SDKs. For more information about using Glacier with the AWS CLI, go to [AWS CLI Reference for Glacier](http://docs.aws.amazon.com/cli/latest/reference/glacier/index.html). To install the AWS CLI, go to [AWS Command Line Interface](http://aws.amazon.com/cli/).

### Downloading an Archive in Amazon S3 Glacier

Downloading an archive is an asynchronous operation. You must first initiate a job to download a specific archive. After receiving the job request, Glacier prepares your archive for download. After the job completes, you can download your archive data. Because of the asynchronous nature of the job, you can request Glacier to send a notification to an Amazon Simple Notification Service (Amazon SNS) topic when the job completes. You can specify an SNS topic for each individual job request or configure your vault to send a notification when specific events occur. For more information about downloading an archive, see [Downloading an Archive in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/downloading-an-archive.html).

### Deleting an Archive in Amazon S3 Glacier

Glacier provides an API call that you can use to delete one archive at a time. For more information, see [Deleting an Archive in Amazon S3 Glacier](https://docs.aws.amazon.com/amazonglacier/latest/dev/deleting-an-archive.html).

### Updating an Archive in Glacier

After you upload an archive, you cannot update its content or its description. The only way you can update the archive content or its description is by deleting the archive and uploading another archive. Note that each time you upload an archive, Glacier returns to you a unique archive ID.

## Maintaining Client-Side Archive Metadata

Except for the optional archive description, Glacier does not support any additional metadata for the archives. When you upload an archive Glacier assigns an ID, an opaque sequence of characters, from which you cannot infer any meaning about the archive. You might maintain metadata about the archives on the client-side. The metadata can include archive name and some other meaningful information about the archive.

**Note**

If you are an Amazon Simple Storage Service (Amazon S3) customer, you know that when you upload an object to a bucket, you can assign the object an object key such as MyDocument.txt or SomePhoto.jpg. In Glacier, you cannot assign a key name to the archives you upload.

If you maintain client-side archive metadata, note that Glacier maintains a vault inventory that includes archive IDs and any descriptions you provided during the archive upload. You might occasionally download the vault inventory to reconcile any issues in your client-side database you maintain for the archive metadata. However, Glacier takes vault inventory approximately daily. When you request a vault inventory, Glacier returns the last inventory it prepared, a point in time snapshot.

# What Is Amazon Route 53?

Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service. You can use Route 53 to perform three main functions in any combination: domain registration, DNS routing, and health checking. If you choose to use Route 53 for all three functions, perform the steps in this order:

**1. Register domain names**

Your website needs a name, such as example.com. Route 53 lets you register a name for your website or web application, known as a domain name.

* For an overview, see [How Domain Registration Works](https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/welcome-domain-registration.html).
* For a procedure, see [Registering a New Domain](https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/domain-register.html).
* For a tutorial that takes you through registering a domain and creating a simple website in an Amazon S3 bucket, see [Getting Started with Amazon Route 53](https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/getting-started.html).

**2. Route internet traffic to the resources for your domain**

When a user opens a web browser and enters your domain name (example.com) or subdomain name (apex.example.com) in the address bar, Route 53 helps connect the browser with your website or web application.

* For an overview, see [How Internet Traffic Is Routed to Your Website or Web Application](https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/welcome-dns-service.html).
* For procedures, see [Configuring Amazon Route 53 as Your DNS Service](https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/dns-configuring.html).

**3. Check the health of your resources**

Route 53 sends automated requests over the internet to a resource, such as a web server, to verify that it's reachable, available, and functional. You also can choose to receive notifications when a resource becomes unavailable and choose to route internet traffic away from unhealthy resources.

* For an overview, see [How Amazon Route 53 Checks the Health of Your Resources](https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/welcome-health-checks.html).
* For procedures, see [Creating Amazon Route 53 Health Checks and Configuring DNS Failover](https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/dns-failover.html).

# What Is AWS Auto Scaling?

AWS Auto Scaling enables you to configure automatic scaling for the AWS resources that are part of your application in a matter of minutes. The AWS Auto Scaling console provides a single user interface to use the automatic scaling features of multiple AWS services. You can configure automatic scaling for individual resources or for whole applications.

With AWS Auto Scaling, you configure and manage scaling for your resources through a scaling plan. The scaling plan uses dynamic scaling and predictive scaling to automatically scale your application’s resources. This ensures that you add the required computing power to handle the load on your application and then remove it when it's no longer required. The scaling plan lets you choose scaling strategies to define how to optimize your resource utilization. You can optimize for availability, for cost, or a balance of both. Alternatively, you can create custom scaling strategies.

AWS Auto Scaling is useful for applications that experience daily or weekly variations in traffic flow, including the following:

* Cyclical traffic such as high use of resources during regular business hours and low use of resources overnight
* On and off traffic patterns, such as batch processing, testing, or periodic analysis
* Variable traffic patterns, such as marketing campaigns with periods of spiky growth

## Features of AWS Auto Scaling

Use AWS Auto Scaling to automatically scale the following resources:

* **Amazon EC2 Auto Scaling groups**: Launch or terminate EC2 instances in an Auto Scaling group.
* **Amazon EC2 Spot Fleet requests**: Launch or terminate instances from a Spot Fleet request, or automatically replace instances that get interrupted for price or capacity reasons.
* **Amazon ECS**: Adjust the ECS service desired count up or down in response to load variations.
* **Amazon DynamoDB**: Enable a DynamoDB table or a global secondary index to increase or decrease its provisioned read and write capacity to handle increases in traffic without throttling.
* **Amazon Aurora**: Dynamically adjust the number of Aurora read replicas provisioned for an Aurora DB cluster to handle changes in active connections or workload.

The scaling features currently available are dynamic scaling and predictive scaling.

Dynamic scaling creates target tracking scaling policies for the scalable resources in your application. This lets your scaling plan add and remove capacity for each resource as required to maintain resource utilization at the specified target value. The default scaling metrics provided are based on the most commonly used metrics used for automatic scaling.

How predictive scaling works:

* **Load forecasting**: AWS Auto Scaling analyzes up to 14 days of history for a specified load metric and forecasts the future demand for the next two days. This data is available in one-hour intervals and updated daily.
* **Scheduled scaling actions**: AWS Auto Scaling schedules the scaling actions that proactively add and remove resource capacity to reflect the load forecast. At the scheduled time, AWS Auto Scaling updates the resource's minimum capacity with the value specified by the scheduled scaling action. The intention is to maintain resource utilization at the target value specified by the scaling strategy. If your application requires more capacity than is forecast, dynamic scaling is available to add additional capacity.
* **Maximum capacity behavior**: Each resource has a minimum and a maximum capacity limit between which the value specified by the scheduled scaling action is expected to lie. However, you can control whether your application can add resources beyond their maximum capacity when the forecast capacity is higher than the maximum capacity.

Currently, predictive scaling is only available for Amazon EC2 Auto Scaling groups.

## Pricing

AWS Auto Scaling features are enabled by Amazon CloudWatch metrics and alarms. The features are provided at no additional charge beyond the service fees for CloudWatch and the other AWS resources that you use.

## How to Get Started

For an introduction to AWS Auto Scaling, we recommend that you familiarize yourself with the following:

* [How AWS Auto Scaling Works](https://docs.aws.amazon.com/autoscaling/plans/userguide/how-it-works.html)—This introduces the concepts of scaling strategies, dynamic scaling, and predictive scaling to help you get familiar with AWS Auto Scaling.
* [AWS Auto Scaling FAQs](https://aws.amazon.com/autoscaling/faqs/)—The FAQ on the product page provides information about the benefits of this service.
* [AWS Region Table](https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/)—This page shows you the regional availability of AWS Auto Scaling.
* [Amazon EC2 Auto Scaling User Guide](https://docs.aws.amazon.com/autoscaling/ec2/userguide/)—This guide shows you how to create and manage the Auto Scaling groups to use when scaling your fleet of Amazon EC2 instances.

## Top benefits of cloud computing

Cloud computing is a big shift from the traditional way businesses think about IT resources. Here are seven common reasons organisations are turning to cloud computing services:

### Cost

Cloud computing eliminates the capital expense of buying hardware and software and setting up and running on-site datacenters—the racks of servers, the round-the-clock electricity for power and cooling, the IT experts for managing the infrastructure. It adds up fast.

### Speed

Most cloud computing services are provided self service and on demand, so even vast amounts of computing resources can be provisioned in minutes, typically with just a few mouse clicks, giving businesses a lot of flexibility and taking the pressure off capacity planning.

### Global scale

The benefits of cloud computing services include the ability to scale elastically. In cloud speak, that means delivering the right amount of IT resources—for example, more or less computing power, storage, bandwidth—right when it is needed and from the right geographic location

### Productivity

On-site datacenters typically require a lot of “racking and stacking”—hardware set up, software patching and other time-consuming IT management chores. Cloud computing removes the need for many of these tasks, so IT teams can spend time on achieving more important business goals.

### Performance

The biggest cloud computing services run on a worldwide network of secure datacenters, which are regularly upgraded to the latest generation of fast and efficient computing hardware. This offers several benefits over a single corporate datacenter, including reduced network latency for applications and greater economies of scale.

### Security

Many cloud providers offer a broad set of policies, technologies and controls that strengthen your security posture overall, helping protect your data, apps and infrastructure from potential threats.

## Types of cloud computing

Not all clouds are the same and not one type of cloud computing is right for everyone. Several different models, types and services have evolved to help offer the right solution for your needs.

Types of cloud deployments: public, private and hybrid

First, you need to determine the type of cloud deployment or cloud computing architecture, that your cloud services will be implemented on. There are three different ways to deploy cloud services: on a public cloud, private cloud or hybrid cloud.

### Public cloud

Public clouds are owned and operated by a third-party [cloud service providers](https://azure.microsoft.com/en-in/overview/choosing-a-cloud-service-provider/), which deliver their computing resources like servers and storage over the Internet. Microsoft Azure is an example of a public cloud. With a public cloud, all hardware, software and other supporting infrastructure is owned and managed by the cloud provider. You access these services and manage your account using a web browser.

### Private cloud

A private cloud refers to cloud computing resources used exclusively by a single business or organisation. A private cloud can be physically located on the company’s on-site datacenter. Some companies also pay third-party service providers to host their private cloud. A private cloud is one in which the services and infrastructure are maintained on a private network.

### Hybrid cloud

Hybrid clouds combine public and private clouds, bound together by technology that allows data and applications to be shared between them. By allowing data and applications to move between private and public clouds, a hybrid cloud gives your business greater flexibility, more deployment options and helps optimise your existing infrastructure, security and compliance.

Types of cloud services: IaaS, PaaS, serverless and SaaS

Most cloud computing services fall into four broad categories: infrastructure as a service (IaaS), platform as a service (PaaS), serverless and software as a service (SaaS). These are sometimes called the cloud computing stack because they build on top of one another. Knowing what they are and how they are different makes it easier to accomplish your business goals.

### Infrastructure as a service (IaaS)

The most basic category of cloud computing services. With IaaS, you rent IT infrastructure—servers and virtual machines (VMs), storage, networks, operating systems—from a cloud provider on a pay-as-you-go basis.

### Platform as a service (PaaS)

Platform as a service refers to cloud computing services that supply an on-demand environment for developing, testing, delivering and managing software applications. PaaS is designed to make it easier for developers to quickly create web or mobile apps, without worrying about setting up or managing the underlying infrastructure of servers, storage, network and databases needed for development.

### Serverless computing

Overlapping with PaaS, [serverless computing](https://azure.microsoft.com/en-in/overview/what-is-serverless-computing/) focuses on building app functionality without spending time continually managing the servers and infrastructure required to do so. The cloud provider handles the setup, capacity planning and server management for you. Serverless architectures are highly scalable and event-driven, only using resources when a specific function or trigger occurs.

### Software as a service (SaaS)

Software as a service is a method for delivering software applications over the Internet, on demand and typically on a subscription basis. With SaaS, cloud providers host and manage the software application and underlying infrastructure and handle any maintenance, like software upgrades and security patching. Users connect to the application over the Internet, usually with a web browser on their phone, tablet or PC.

## How cloud computing works

While cloud computing services all work a little differently, many provide a friendly, browser-based dashboard that makes it easier for IT professionals and developers to order resources and manage their accounts. Some cloud computing services are also designed to work with REST APIs and a command-line interface, giving developers multiple options.

## Uses of cloud computing

You are probably using cloud computing right now, even if you don’t realise it. If you use an online service to send email, edit documents, watch movies or TV, listen to music, play games or store pictures and other files, it is likely that cloud computing is making it all possible behind the scenes. The first cloud computing services are barely a decade old, but already a variety of organisations—from tiny startups to global corporations, government agencies to non-profits—are embracing the technology for all sorts of reasons.

Here are a few examples of what is possible today with cloud services from a cloud provider:

### Create new apps and services

Quickly build, deploy and scale applications—web, mobile and API—on any platform. Access the resources you need to help meet performance, security and compliance requirements.

### Test and build applications

Reduce application development cost and time by using cloud infrastructures that can easily be scaled up or down.

### Store, back up and recover data

Protect your data more cost-efficiently—and at massive scale—by transferring your data over the Internet to an offsite cloud storage system that is accessible from any location and any device.

### Analyse data

Unify your data across teams, divisions and locations in the cloud. Then use cloud services, such as machine learning and artificial intelligence, to uncover insights for more informed decisions.

### Stream audio and video

Connect with your audience anywhere, anytime, on any device with high-definition video and audio with global distribution.

### Embed intelligence

Use intelligent models to help engage customers and provide valuable insights from the data captured.

## 

# What Is Elastic Load Balancing?

Elastic Load Balancing distributes incoming application or network traffic across multiple targets, such as Amazon EC2 instances, containers, and IP addresses, in multiple Availability Zones. Elastic Load Balancing scales your load balancer as traffic to your application changes over time, and can scale to the vast majority of workloads automatically.

## Load Balancer Benefits

A load balancer distributes workloads across multiple compute resources, such as virtual servers. Using a load balancer increases the availability and fault tolerance of your applications.

You can add and remove compute resources from your load balancer as your needs change, without disrupting the overall flow of requests to your applications.

You can configure health checks, which are used to monitor the health of the compute resources so that the load balancer can send requests only to the healthy ones. You can also offload the work of encryption and decryption to your load balancer so that your compute resources can focus on their main work.

## Features of Elastic Load Balancing

Elastic Load Balancing supports three types of load balancers: Application Load Balancers, Network Load Balancers, and Classic Load Balancers. You can select a load balancer based on your application needs. For more information, see [Comparison of Elastic Load Balancing Products](https://aws.amazon.com/elasticloadbalancing/details/#compare).

For more information about using each load balancer, see the [User Guide for Application Load Balancers](https://docs.aws.amazon.com/elasticloadbalancing/latest/application/), the [User Guide for Network Load Balancers](https://docs.aws.amazon.com/elasticloadbalancing/latest/network/), and the [User Guide for Classic Load Balancers](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/).

## Accessing Elastic Load Balancing

You can create, access, and manage your load balancers using any of the following interfaces:

* **AWS Management Console**— Provides a web interface that you can use to access Elastic Load Balancing.
* **AWS Command Line Interface (AWS CLI)** — Provides commands for a broad set of AWS services, including Elastic Load Balancing, and is supported on Windows, Mac, and Linux. For more information, see [AWS Command Line Interface](https://aws.amazon.com/cli/).
* **AWS SDKs** — Provides language-specific APIs and takes care of many of the connection details, such as calculating signatures, handling request retries, and error handling. For more information, see [AWS SDKs](http://aws.amazon.com/tools/#SDKs).
* **Query API**— Provides low-level API actions that you call using HTTPS requests. Using the Query API is the most direct way to access Elastic Load Balancing, but it requires that your application handle low-level details such as generating the hash to sign the request, and error handling. For more information, see the following:
  + Application Load Balancers and Network Load Balancers — [API version 2015-12-01](https://docs.aws.amazon.com/elasticloadbalancing/latest/APIReference/)
  + Classic Load Balancers — [API version 2012-06-01](https://docs.aws.amazon.com/elasticloadbalancing/2012-06-01/APIReference/)

## Related Services

Elastic Load Balancing works with the following services to improve the availability and scalability of your applications.

* **Amazon EC2** — Virtual servers that run your applications in the cloud. You can configure your load balancer to route traffic to your EC2 instances. For more information, see the [Amazon EC2 User Guide for Linux Instances](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/) or the [Amazon EC2 User Guide for Windows Instances](https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/).
* **Amazon EC2 Auto Scaling** — Ensures that you are running your desired number of instances, even if an instance fails, and enables you to automatically increase or decrease the number of instances as the demand on your instances changes. If you enable Auto Scaling with Elastic Load Balancing, instances that are launched by Auto Scaling are automatically registered with the load balancer, and instances that are terminated by Auto Scaling are automatically de-registered from the load balancer. For more information, see the [Amazon EC2 Auto Scaling User Guide](https://docs.aws.amazon.com/autoscaling/latest/userguide/).
* **AWS Certificate Manager** — When you create an HTTPS listener, you can specify certificates provided by ACM. The load balancer uses certificates to terminate connections and decrypt requests from clients.
* **Amazon CloudWatch** — Enables you to monitor your load balancer and take action as needed. For more information, see the [Amazon CloudWatch User Guide](https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/).
* **Amazon ECS** — Enables you to run, stop, and manage Docker containers on a cluster of EC2 instances. You can configure your load balancer to route traffic to your containers. For more information, see the [Amazon Elastic Container Service Developer Guide](https://docs.aws.amazon.com/AmazonECS/latest/developerguide/).
* **Route 53** — Provides a reliable and cost-effective way to route visitors to websites by translating domain names (such as www.example.com) into the numeric IP addresses (such as 192.0.2.1) that computers use to connect to each other. AWS assigns URLs to your resources, such as load balancers. However, you might want a URL that is easy for users to remember. For example, you can map your domain name to a load balancer. For more information, see the [Amazon Route 53 Developer Guide](https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/).
* **AWS WAF** — You can use AWS WAF with your Application Load Balancer to allow or block requests based on the rules in a web access control list (web ACL). For more information, see the [AWS WAF Developer Guide](https://docs.aws.amazon.com/waf/latest/developerguide/).

# How Elastic Load Balancing Works

A load balancer accepts incoming traffic from clients and routes requests to its registered targets (such as EC2 instances) in one or more Availability Zones. The load balancer also monitors the health of its registered targets and ensures that it routes traffic only to healthy targets. When the load balancer detects an unhealthy target, it stops routing traffic to that target, and then resumes routing traffic to that target when it detects that the target is healthy again.

You configure your load balancer to accept incoming traffic by specifying one or more listeners. A listener is a process that checks for connection requests. It is configured with a protocol and port number for connections from clients to the load balancer and a protocol and port number for connections from the load balancer to the targets.

Elastic Load Balancing supports three types of load balancers: Application Load Balancers, Network Load Balancers, and Classic Load Balancers. There is a key difference between the way you configure these load balancers. With Application Load Balancers and Network Load Balancers, you register targets in target groups, and route traffic to the target groups. With Classic Load Balancers, you register instances with the load balancer.

## Availability Zones and Load Balancer Nodes

When you enable an Availability Zone for your load balancer, Elastic Load Balancing creates a load balancer node in the Availability Zone. If you register targets in an Availability Zone but do not enable the Availability Zone, these registered targets do not receive traffic. Note that your load balancer is most effective if you ensure that each enabled Availability Zone has at least one registered target.

We recommend that you enable multiple Availability Zones. (Note that with an Application Load Balancer, we require you to enable multiple Availability Zones.) With this configuration, if one Availability Zone becomes unavailable or has no healthy targets, the load balancer can continue to route traffic to the healthy targets in another Availability Zone.

After you disable an Availability Zone, the targets in that Availability Zone remain registered with the load balancer, but the load balancer will not route traffic to them.

### Cross-Zone Load Balancing

The nodes for your load balancer distribute requests from clients to registered targets. When cross-zone load balancing is enabled, each load balancer node distributes traffic across the registered targets in all enabled Availability Zones. When cross-zone load balancing is disabled, each load balancer node distributes traffic across the registered targets in its Availability Zone only.

The following diagrams demonstrate the effect of cross-zone load balancing. There are two enabled Availability Zones, with 2 targets in Availability Zone A and 8 targets in Availability Zone B. Clients send requests, and Amazon Route 53 responds to each request with the IP address of one of the load balancer nodes. This distributes traffic such that each load balancer node receives 50% of the traffic from the clients. Each load balancer node distributes its share of the traffic across the registered targets in its scope.

If cross-zone load balancing is enabled, each of the 10 targets receives 10% of the traffic. This is because each load balancer node can route its 50% of the client traffic to all 10 targets.


                    When cross-zone load balancing is enabled
                

If cross-zone load balancing is disabled, each of the 2 targets in Availability Zone A receives 25% of the traffic and each of the 8 targets in Availability Zone B receives 6.25% of the traffic. This is because each load balancer node can route its 50% of the client traffic only to targets in its Availability Zone.


                    When cross-zone load balancing is disabled
                

With Application Load Balancers, cross-zone load balancing is always enabled.

With Network Load Balancers, cross-zone load balancing is disabled by default. After you create a Network Load Balancer, you can enable or disable cross-zone load balancing at any time. For more information, see [Cross-Zone Load Balancing](https://docs.aws.amazon.com/elasticloadbalancing/latest/network/network-load-balancers.html#cross-zone-load-balancing) in the User Guide for Network Load Balancers.

When you create a Classic Load Balancer, the default for cross-zone load balancing depends on how you create the load balancer. With the API or CLI, cross-zone load balancing is disabled by default. With the AWS Management Console, the option to enable cross-zone load balancing is selected by default. After you create a Classic Load Balancer, you can enable or disable cross-zone load balancing at any time. For more information, see [Enable Cross-Zone Load Balancing](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/enable-disable-crosszone-lb.html#enable-cross-zone) in the User Guide for Classic Load Balancers.

## Request Routing

Before a client sends a request to your load balancer, it resolves the load balancer's domain name using a Domain Name System (DNS) server. The DNS entry is controlled by Amazon, because your load balancers are in the amazonaws.com domain. The Amazon DNS servers return one or more IP addresses to the client, which are the IP addresses of the load balancer nodes for your load balancer. With Network Load Balancers, Elastic Load Balancing creates a network interface for each Availability Zone you enable. Each load balancer node in the Availability Zone uses this network interface to get a static IP address. You can optionally associate one Elastic IP address with each network interface when you create the load balancer.

As traffic to your application changes over time, Elastic Load Balancing scales your load balancer and updates the DNS entry. Note that the DNS entry also specifies the time-to-live (TTL) as 60 seconds, which ensures that the IP addresses can be remapped quickly in response to changing traffic.

The client determines which IP address to use to send requests to the load balancer. The load balancer node that receives the request selects a healthy registered target and sends the request to the target using its private IP address.

### Routing Algorithm

With Application Load Balancers, the load balancer node that receives the request evaluates the listener rules in priority order to determine which rule to apply, and then selects a target from the target group for the rule action using the round robin routing algorithm. Routing is performed independently for each target group, even when a target is registered with multiple target groups.

With Network Load Balancers, the load balancer node that receives the connection selects a target from the target group for the default rule using a flow hash algorithm, based on the protocol, source IP address, source port, destination IP address, destination port, and TCP sequence number. The TCP connections from a client have different source ports and sequence numbers, and can be routed to different targets. Each individual TCP connection is routed to a single target for the life of the connection.

With Classic Load Balancers, the load balancer node that receives the request selects a registered instance using the round robin routing algorithm for TCP listeners and the least outstanding requests routing algorithm for HTTP and HTTPS listeners.

### HTTP Connections

Classic Load Balancers use pre-open connections but Application Load Balancers do not. Both Classic Load Balancers and Application Load Balancers use connection multiplexing. This means that requests from multiple clients on multiple front-end connections can be routed to a given target through a single back-end connection. Connection multiplexing improves latency and reduces the load on your applications. To prevent connection multiplexing, disable HTTP keep-alives by setting the Connection: close header in your HTTP responses.

Classic Load Balancers support the following protocols on front-end connections (client to load balancer): HTTP/0.9, HTTP/1.0, and HTTP/1.1.

Application Load Balancers support the following protocols on front-end connections: HTTP/0.9, HTTP/1.0, HTTP/1.1, and HTTP/2. You can use HTTP/2 only with HTTPS listeners, and send up to 128 requests in parallel using one HTTP/2 connection. Application Load Balancers also support connection upgrades from HTTP to Websockets.

Both Application Load Balancers and Classic Load Balancers use HTTP/1.1 on back-end connections (load balancer to registered target). Keep-alive is supported on back-end connections by default. For HTTP/1.0 requests from clients that do not have a host header, the load balancer generates a host header for the HTTP/1.1 requests sent on the back-end connections. For Application Load Balancer, the host header contains the DNS name of the load balancer. For Classic Load Balancer, the host header contains the IP address of the load balancer node.

You can set an idle timeout value for both Application Load Balancers and Classic Load Balancers. The default value is 60 seconds. With an Application Load Balancer, the idle timeout value applies only to front-end connections. With a Classic Load Balancer, if a front-end connection or a back-end connection is idle for longer than the idle timeout value, the connection is torn down and the client receives an error response. A registered target can use a keep-alive timeout to keep a back-end connection open until it is ready to tear it down.

Application Load Balancers and Classic Load Balancers support pipelined HTTP on front-end connections. They do not support pipelined HTTP on back-end connections.

### HTTP Headers

Application Load Balancers and Classic Load Balancers support **X-Forwarded-For**, **X-Forwarded-Proto**, and **X-Forwarded-Port** headers.

For front-end connections that use HTTP/2, the header names are in lowercase. Before the request is sent to the target using HTTP/1.1, the following header names are converted to mixed case: **X-Forwarded-For**, **X-Forwarded-Proto**, **X-Forwarded-Port**, **Host**, **X-Amzn-Trace-Id**, **Upgrade**, and **Connection**. All other header names are in lowercase.

Application Load Balancers and Classic Load Balancers honor the connection header from the incoming client request after proxying the response back to the client.

### HTTP Header Limits

HTTP/1.x headers for Application Load Balancers have the following size limits:

* Request line: 16K
* Single header: 16K
* Whole header: 64K

HTTP/2 headers for Application Load Balancers have the following size limits:

* Request line: 8K
* Single header: 8K
* Whole header: 64K

## Load Balancer Scheme

When you create a load balancer, you must choose whether to make it an internal load balancer or an Internet-facing load balancer. Note that when you create a Classic Load Balancer in EC2-Classic, it must be an Internet-facing load balancer.

The nodes of an Internet-facing load balancer have public IP addresses. The DNS name of an Internet-facing load balancer is publicly resolvable to the public IP addresses of the nodes. Therefore, Internet-facing load balancers can route requests from clients over the Internet.

The nodes of an internal load balancer have only private IP addresses. The DNS name of an internal load balancer is publicly resolvable to the private IP addresses of the nodes. Therefore, internal load balancers can only route requests from clients with access to the VPC for the load balancer.

Note that both Internet-facing and internal load balancers route requests to your targets using private IP addresses. Therefore, your targets do not need public IP addresses to receive requests from an internal or an Internet-facing load balancer.

If your application has multiple tiers, for example web servers that must be connected to the Internet and database servers that are only connected to the web servers, you can design an architecture that uses both internal and Internet-facing load balancers. Create an Internet-facing load balancer and register the web servers with it. Create an internal load balancer and register the database servers with it. The web servers receive requests from the Internet-facing load balancer and send requests for the database servers to the internal load balancer. The database servers receive requests from the internal load balancer.

# Getting Started with Elastic Load Balancing

There are three types of load balancers: Application Load Balancers, Network Load Balancers, and Classic Load Balancers. You can select a load balancer based on your application needs. For more information, see [Comparison of Elastic Load Balancing Products](https://aws.amazon.com/elasticloadbalancing/details/#compare).

If you have an existing Classic Load Balancer, you can migrate to an Application Load Balancer or a Network Load Balancer. For more information, see [Migrate Your Classic Load Balancer](https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/migrate-to-application-load-balancer.html).

## Create an Application Load Balancer

To create an Application Load Balancer using the AWS Management Console, see [Getting Started with Application Load Balancers](https://docs.aws.amazon.com/elasticloadbalancing/latest/application/application-load-balancer-getting-started.html) in the User Guide for Application Load Balancers.

To create an Application Load Balancer using the AWS CLI, see [Create an Application Load Balancer Using the AWS CLI](https://docs.aws.amazon.com/elasticloadbalancing/latest/application/tutorial-application-load-balancer-cli.html) in the User Guide for Application Load Balancers.

## Create a Network Load Balancer

To create a Network Load Balancer using the AWS Management Console, see [Getting Started with Network Load Balancers](https://docs.aws.amazon.com/elasticloadbalancing/latest/network/network-load-balancer-getting-started.html) in the User Guide for Network Load Balancers.

To create a Network Load Balancer using the AWS CLI, see [Create a Network Load Balancer Using the AWS CLI](https://docs.aws.amazon.com/elasticloadbalancing/latest/network/network-load-balancer-cli.html) in the User Guide for Network Load Balancers.

## Create a Classic Load Balancer

To create a Classic Load Balancer using the AWS Management Console, see [Create a Classic Load Balancer](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html) in the User Guide for Classic Load Balancers.

# Tutorial: Create a Classic Load Balancer

This tutorial provides a hands-on introduction to Classic Load Balancers through the AWS Management Console, a web-based interface. You'll create a load balancer that receives public HTTP traffic and sends it to your EC2 instances.

Note that you can create your load balancer for use with EC2-Classic or a VPC. Some of the tasks described in this tutorial apply only to load balancers in a VPC.

**Tasks**

* [Before You Begin](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html#getting-started-prerequisites)
* [Step 1: Select a Load Balancer Type](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html#select-load-balancer-type)
* [Step 2: Define Your Load Balancer](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html#define-load-balancer)
* [Step 3: Assign Security Groups to Your Load Balancer in a VPC](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html#select-vpc-security-group)
* [Step 4: Configure Health Checks for Your EC2 Instances](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html#configure-health-check)
* [Step 5: Register EC2 Instances with Your Load Balancer](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html#register-ec2instances)
* [Step 6: Tag Your Load Balancer (Optional)](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html#create-tags)
* [Step 7: Create and Verify Your Load Balancer](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html#create-load-balancer)
* [Step 8: Delete Your Load Balancer (Optional)](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-getting-started.html#delete-load-balancer)

## Before You Begin

* Complete the steps in [Prepare Your VPC and EC2 Instances](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-backend-instances.html#set-up-ec2).
* Launch the EC2 instances that you plan to register with your load balancer. Ensure that the security groups for these instances allow HTTP access on port 80.
* Install a web server, such as Apache or Internet Information Services (IIS), on each instance, enter its DNS name into the address field of an Internet-connected web browser, and verify that the browser displays the default page of the server.

## Step 1: Select a Load Balancer Type

Elastic Load Balancing supports three types of load balancers: Application Load Balancers, Network Load Balancers, and Classic Load Balancers. For this tutorial, you create a Classic Load Balancer. Alternatively, to create an Application Load Balancer, see [Getting Started with Application Load Balancers](https://docs.aws.amazon.com/elasticloadbalancing/latest/application/application-load-balancer-getting-started.html) in the User Guide for Application Load Balancers. To create a Network Load Balancer, see [Getting Started with Network Load Balancers](https://docs.aws.amazon.com/elasticloadbalancing/latest/network/network-load-balancer-getting-started.html) in the User Guide for Network Load Balancers.

**To create a Classic Load Balancer**

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. On the navigation bar, choose a region for your load balancer. Be sure to select the same region that you selected for your EC2 instances.
3. On the navigation pane, under **LOAD BALANCING**, choose **Load Balancers**.
4. Choose **Create Load Balancer**.
5. For **Classic Load Balancer**, choose **Create**.

## Step 2: Define Your Load Balancer

You must provide a basic configuration for your load balancer, such as a name, a network, and a listener.

A listener is a process that checks for connection requests. It is configured with a protocol and a port for front-end (client to load balancer) connections and a protocol and a port for back-end (load balancer to instance) connections. In this tutorial, you configure a listener that accepts HTTP requests on port 80 and sends them to your instances on port 80 using HTTP.

**To define your load balancer and listener**

1. For **Load Balancer name**, type a name for your load balancer.

The name of your Classic Load Balancer must be unique within your set of Classic Load Balancers for the region, can have a maximum of 32 characters, can contain only alphanumeric characters and hyphens, and must not begin or end with a hyphen.

1. For **Create LB inside**, select the same network that you selected for your instances: EC2-Classic or a specific VPC.
2. [Default VPC] If you selected a default VPC and would like to choose the subnets for your load balancer, select **Enable advanced VPC configuration**.
3. Leave the default listener configuration.


      Define your load balancer
     

1. [EC2-VPC] For **Available subnets**, select at least one available public subnet using its add icon. The subnet is moved under **Selected subnets**. To improve the availability of your load balancer, select more than one public subnet.

**Note**

If you selected EC2-Classic as your network, or you have a default VPC but did not select **Enable advanced VPC configuration**, you do not see the user interface to select subnets.

You can add at most one subnet per Availability Zone. If you select a subnet from an Availability Zone where there is already an selected subnet, this subnet replaces the currently selected subnet for the Availability Zone.


       Select Subnets
      

1. Choose **Next: Assign Security Groups**.

## Step 3: Assign Security Groups to Your Load Balancer in a VPC

If you selected a VPC as your network, you must assign your load balancer a security group that allows inbound traffic to the ports that you specified for your load balancer and the health checks for your load balancer.

**Note**

If you selected EC2-Classic as your network, you can continue to the next step. By default, Elastic Load Balancing provides a security group for load balancers in EC2-Classic.

**To assign security group to your load balancer**

1. On the **Assign Security Groups** page, select **Create a new security group**.
2. Type a name and description for your security group, or leave the default name and description. This new security group contains a rule that allows traffic to the port that you configured your load balancer to use.


       Select security groups
      

1. Choose **Next: Configure Security Settings**.
2. For this tutorial, you are not using a secure listener. Choose **Next: Configure Health Check** to continue to the next step.

## Step 4: Configure Health Checks for Your EC2 Instances

Elastic Load Balancing automatically checks the health of the EC2 instances for your load balancer. If Elastic Load Balancing finds an unhealthy instance, it stops sending traffic to the instance and reroutes traffic to healthy instances. In this step, you customize the health checks for your load balancer.

**To configure health checks for your instances**

1. On the **Configure Health Check** page, leave **Ping Protocol** set to HTTP and **Ping Port** set to 80.
2. For **Ping Path**, replace the default value with a single forward slash ("/"). This tells Elastic Load Balancing to send health check queries to the default home page for your web server, such as index.html.


       Configure Health Check
      

1. For **Advanced Details**, leave the default values.
2. Choose **Next: Add EC2 Instances**.

## Step 5: Register EC2 Instances with Your Load Balancer

Your load balancer distributes traffic between the instances that are registered to it.

**Note**

When you register an instance with an elastic network interface (ENI) attached, the load balancer routes traffic to the primary IP address of the primary interface (eth0) of the instance.

**To register EC2 instances with your load balancer**

1. On the **Add EC2 Instances** page, select the instances to register with your load balancer.
2. Leave cross-zone load balancing and connection draining enabled.
3. Choose **Next: Add Tags**.

Alternatively, you can register instances with your load balancer later on using the following options:

* Select running instances after you create the load balancer. For more information, see [Register Instances with Your Load Balancer](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/elb-deregister-register-instances.html).
* Set up Auto Scaling to register the instances automatically when it launches them. For more information, see [Set Up a Scaled and Load-Balanced Application](https://docs.aws.amazon.com/autoscaling/latest/userguide/as-register-lbs-with-asg.html) in the Amazon EC2 Auto Scaling User Guide.

## Step 6: Tag Your Load Balancer (Optional)

You can tag your load balancer, or continue to the next step. Note that you can tag your load balancer later on; for more information, see [Tag Your Classic Load Balancer](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/add-remove-tags.html).

**To add tags to your load balancer**

1. On the **Add Tags** page, specify a key and a value for the tag.
2. To add another tag, choose **Create Tag** and specify a key and a value for the tag.
3. After you are finished adding tags, choose **Review and Create**.

## Step 7: Create and Verify Your Load Balancer

Before you create the load balancer, review the settings that you selected. After creating the load balancer, you can verify that it's sending traffic to your EC2 instances.

**To create and test your load balancer**

1. On the **Review** page, choose **Create**.
2. After you are notified that your load balancer was created, choose **Close**.
3. Select your new load balancer.
4. On the **Description** tab, check the **Status** row. If it indicates that some of your instances are not in service, its probably because they are still in the registration process. For more information, see [Troubleshoot a Classic Load Balancer: Instance Registration](https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/ts-elb-register-instance.html).
5. After at least one of your EC2 instances is in service, you can test your load balancer. Copy the string from **DNS name** (for example, my-load-balancer-1234567890.us-west-2.elb.amazonaws.com) and paste it into the address field of an Internet-connected web browser. If your load balancer is working, you see the default page of your server.

## Step 8: Delete Your Load Balancer (Optional)

As soon as your load balancer becomes available, you are billed for each hour or partial hour that you keep it running. When you no longer need a load balancer, you can delete it. As soon as the load balancer is deleted, you stop incurring charges for it. Note that deleting a load balancer does not affect the instances registered with the load balancer.

**To delete your load balancer**

1. If you have a CNAME record for your domain that points to your load balancer, point it to a new location and wait for the DNS change to take effect before deleting your load balancer.
2. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
3. On the navigation pane, under **LOAD BALANCING**, choose **Load Balancers**.
4. Select the load balancer.
5. Choose **Actions**, **Delete**.
6. When prompted for confirmation, choose **Yes, Delete**.
7. (Optional) After you delete a load balancer, the EC2 instances associated with the load balancer continue to run, and you are billed for each hour or partial hour that you keep them running. For information about stopping or terminating your instances, see [Stop and Start Your Instance](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Stop_Start.html) or [Terminate Your Instance](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/terminating-instances.html) in the Amazon EC2 User Guide for Linux Instances.