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

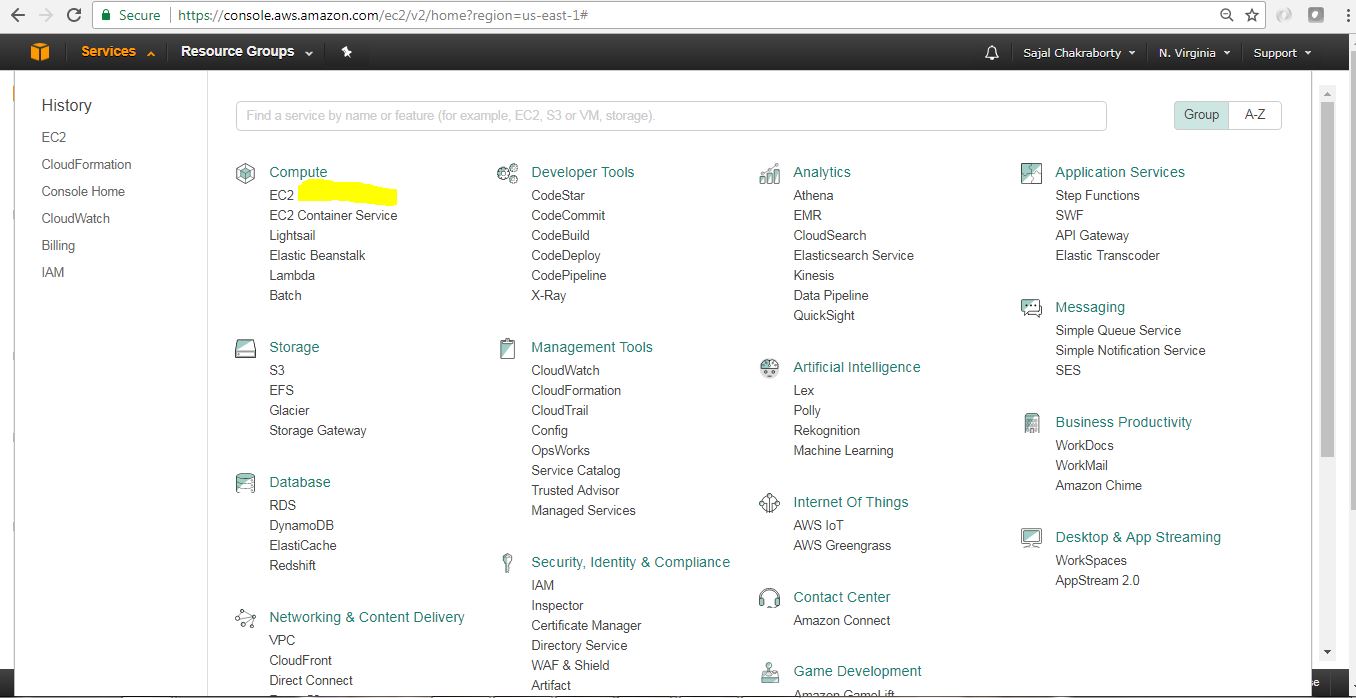
## Prerequisites

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

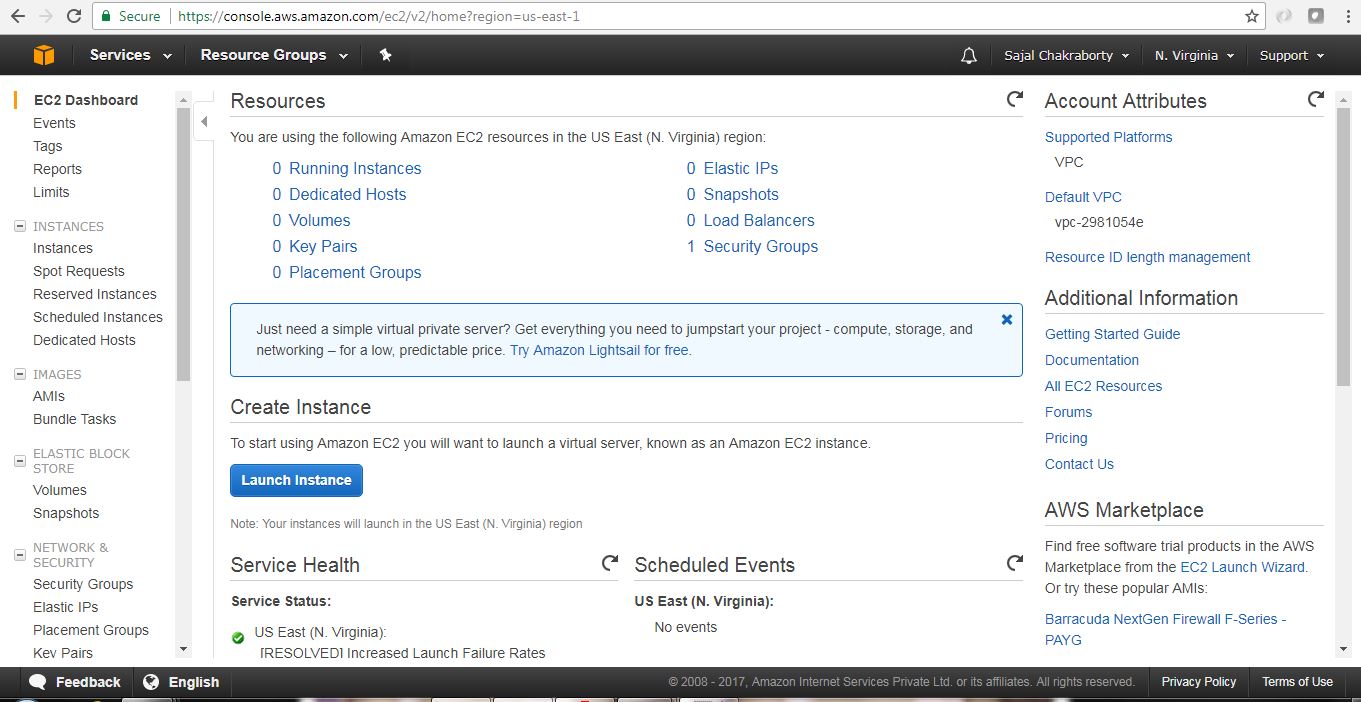
* **AWS account** – Must have to open one AWS account to do this exercise.
* **SSL Client – PuTTY** – This should be installed in Local M/C including PuTTYgen, PUTTY, Pageantapplications. Full set of applications related to PutTTY can be downloaded from [this link](https://the.earth.li/~sgtatham/putty/latest/w64/putty.zip).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

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

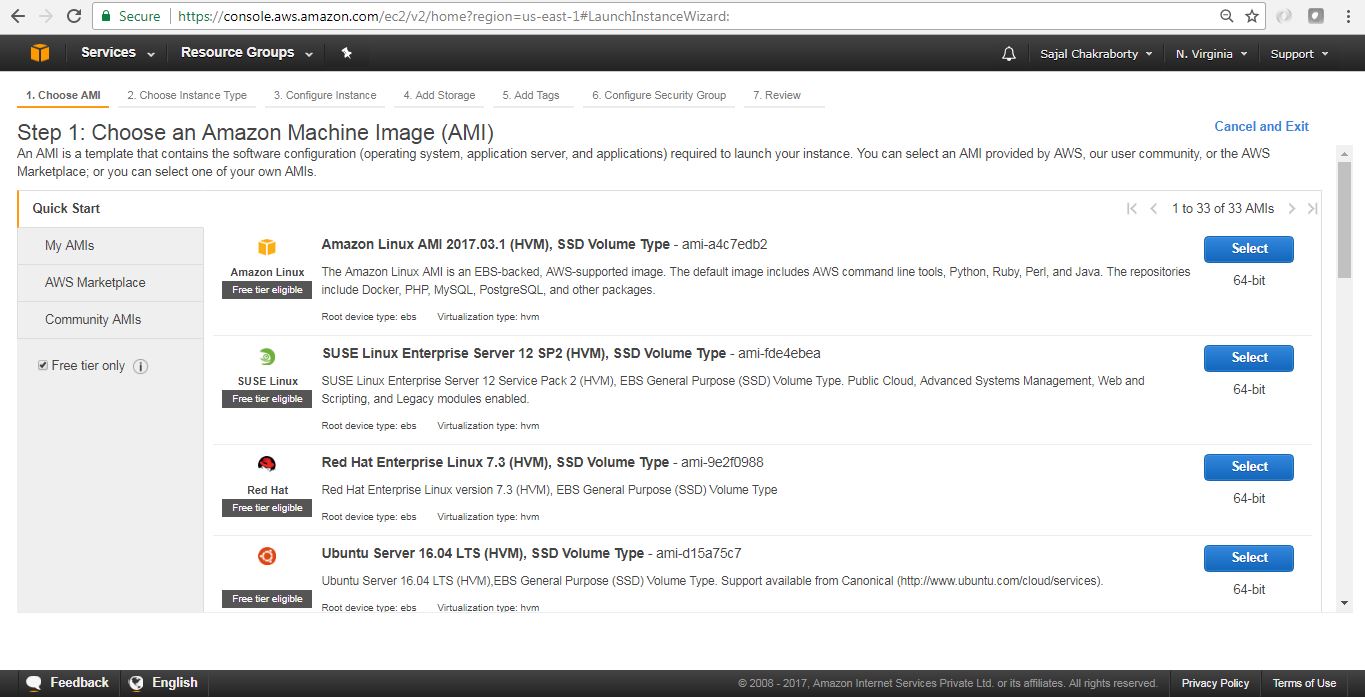
## Steps to Create EC2 Instance

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).

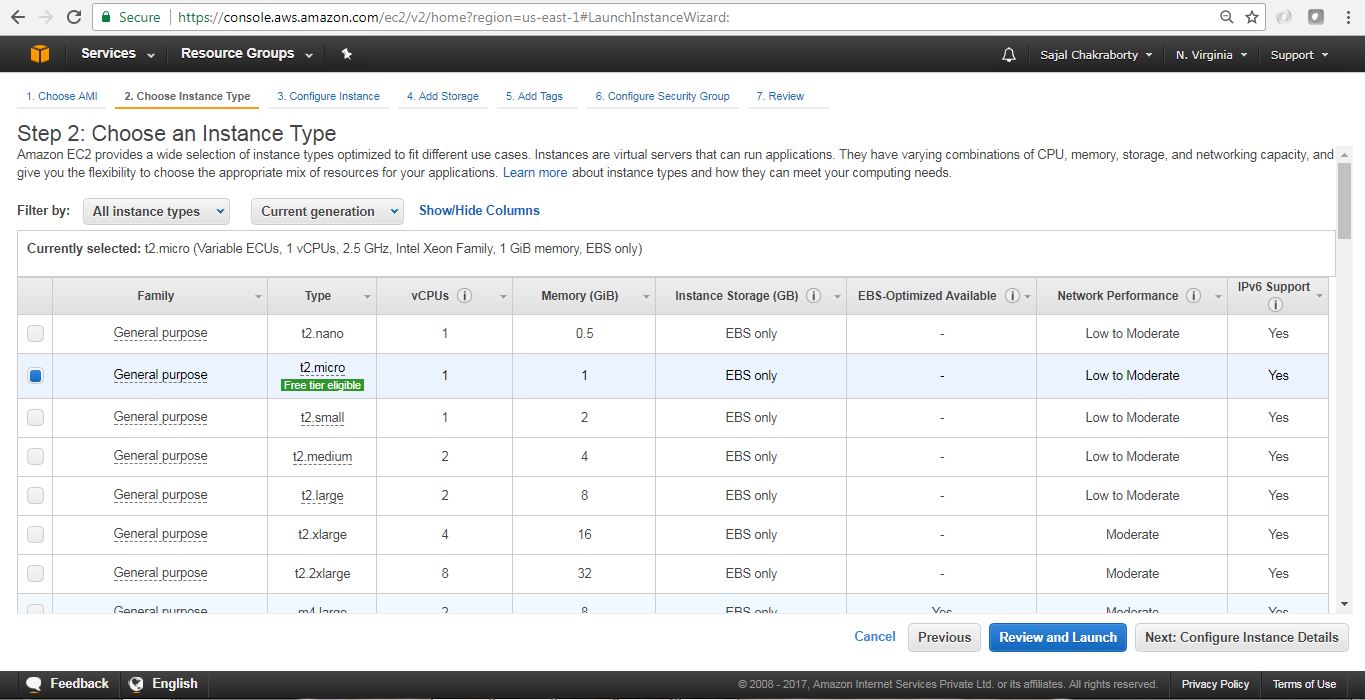
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_1.jpg)Figure 1.1 : EC2 Home Screen

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.

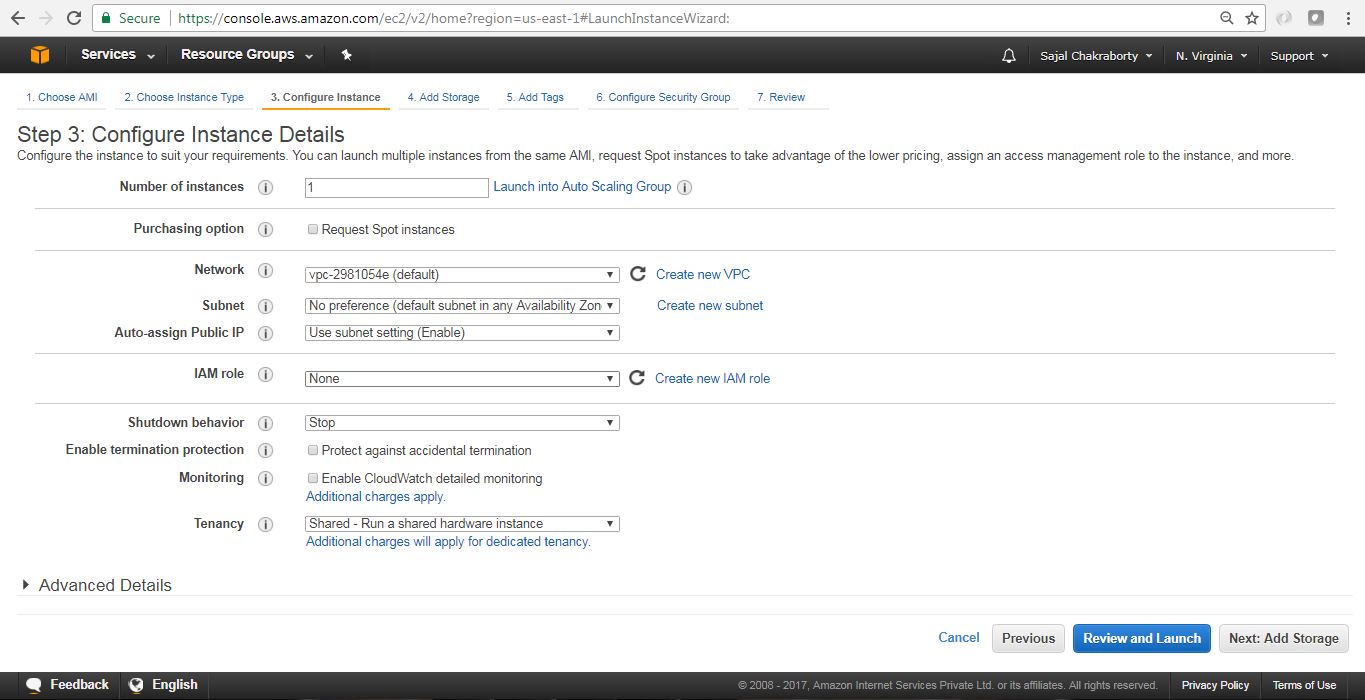
1. **Select Instance Type** – It’s now time to choose the computing power needed for your virtual server. Instance Type indicates the computing power of the instance by allocating the amount of virtual hardware dedicated to the instance. On AWS, computing power is classified into instance types. An instance 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.

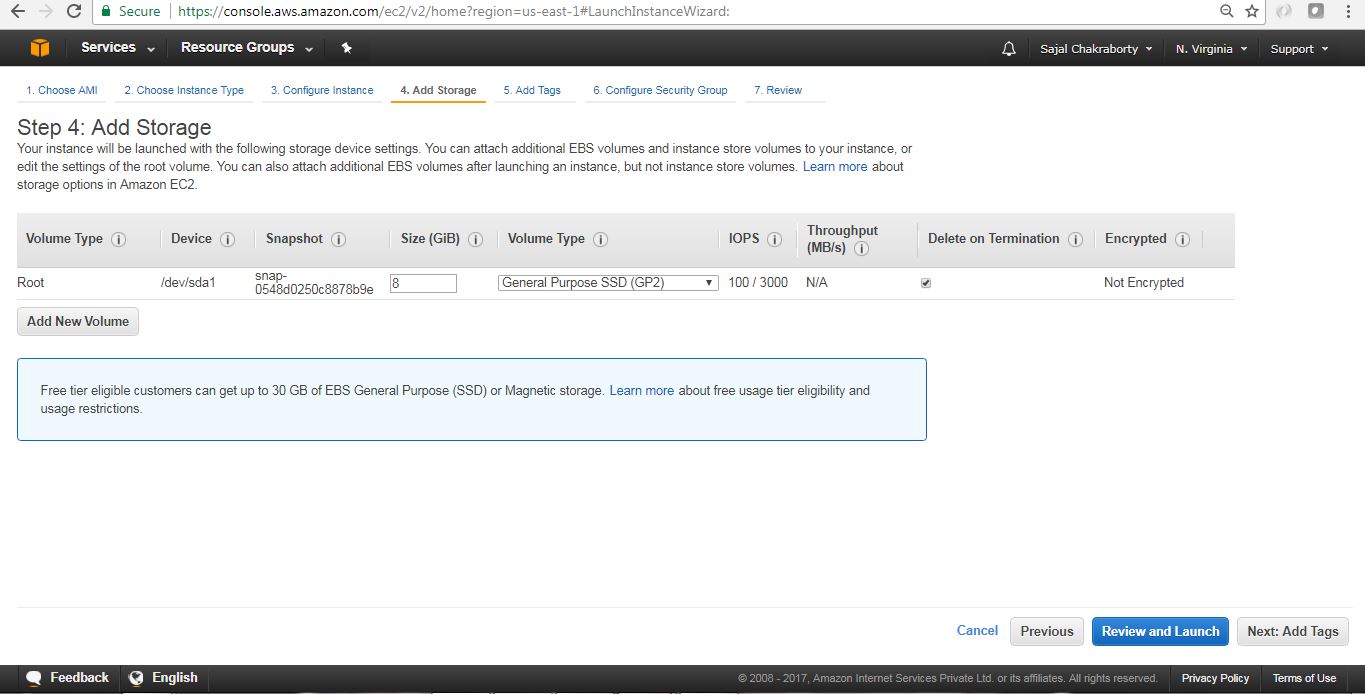
We can Change those later also once the Instance has been created.

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

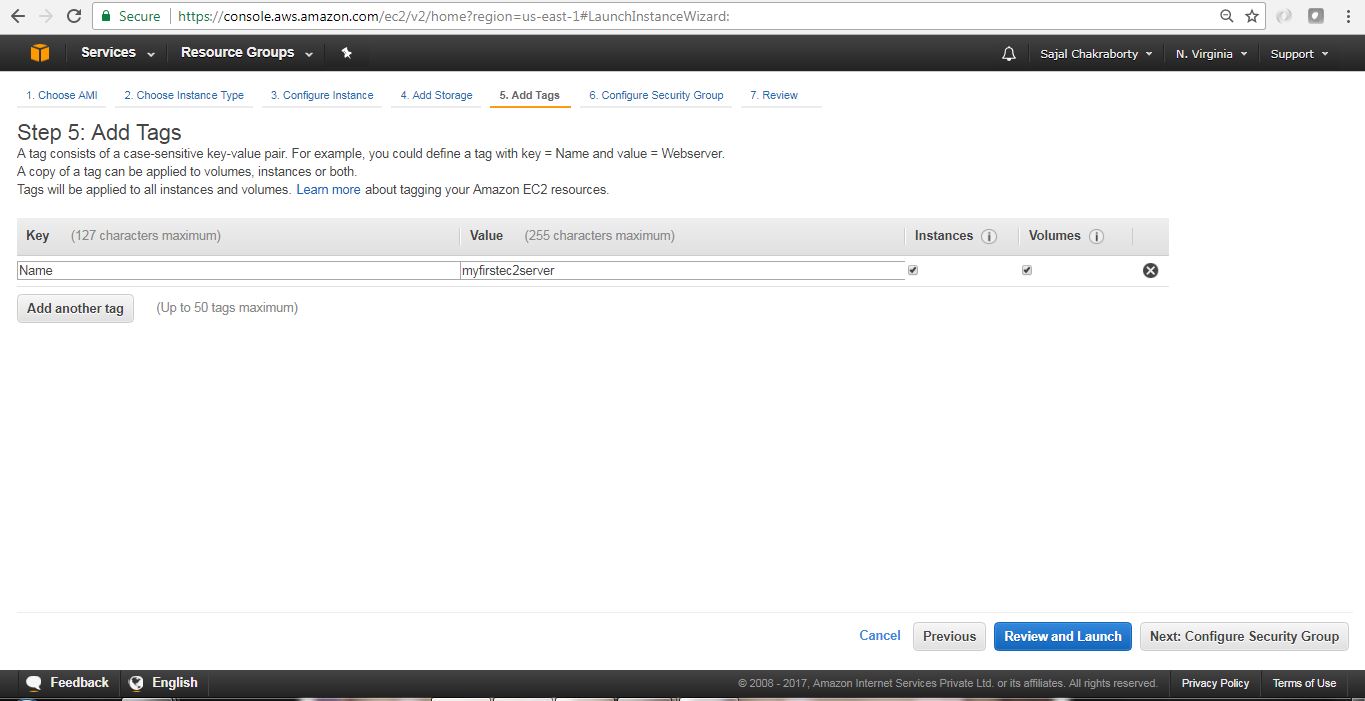
1. **Add EBS Storage** – EBS stands for Elastic block storage. It is basically network-attached storage attached your virtual server. This page will help us to configure Storage that will be associated with the EC2 instance that we are going to launch. We will choose default values to avoid extra cost.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

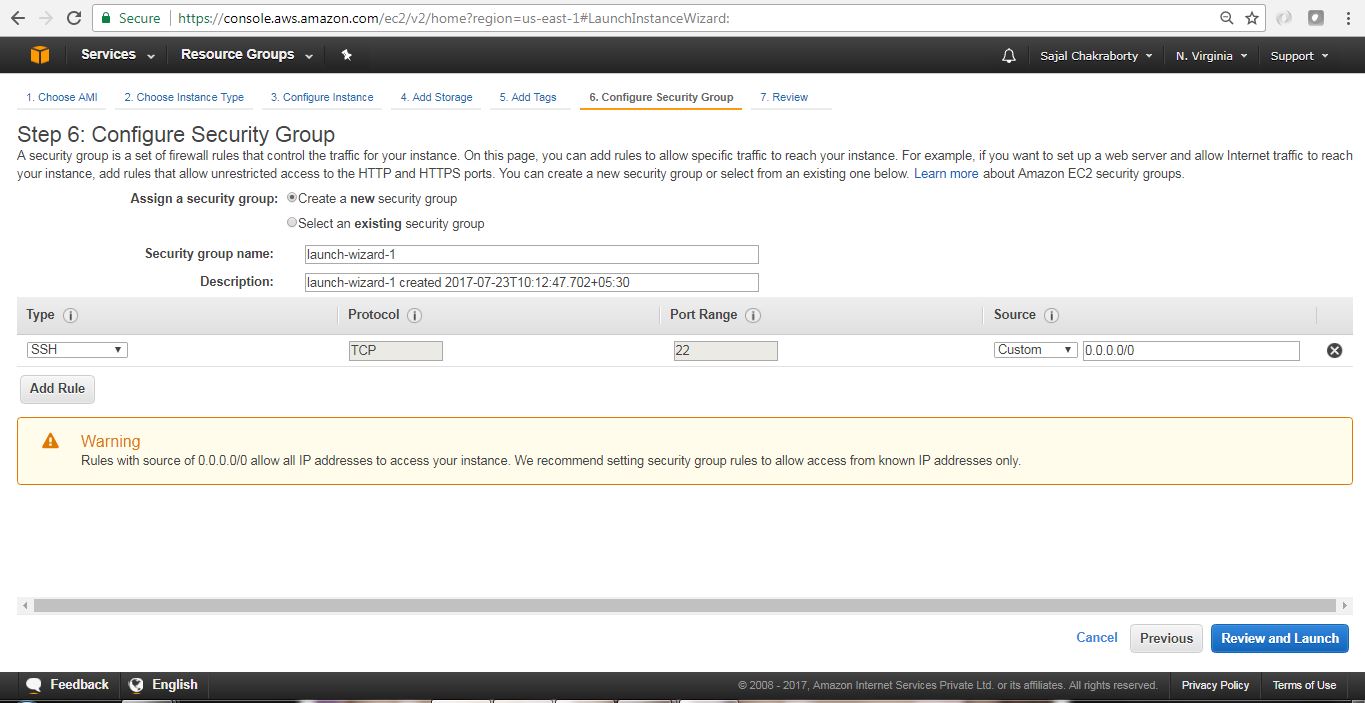
1. **Add Tags** – Tags are kind of an identifier of any AWS resources, with Tags we can easily locate the resource in future and also it helps us to classify the resources. Tags help you to organize resources on AWS. We can add max 50 tags to a particular resources. It is a simple Key value pair associated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

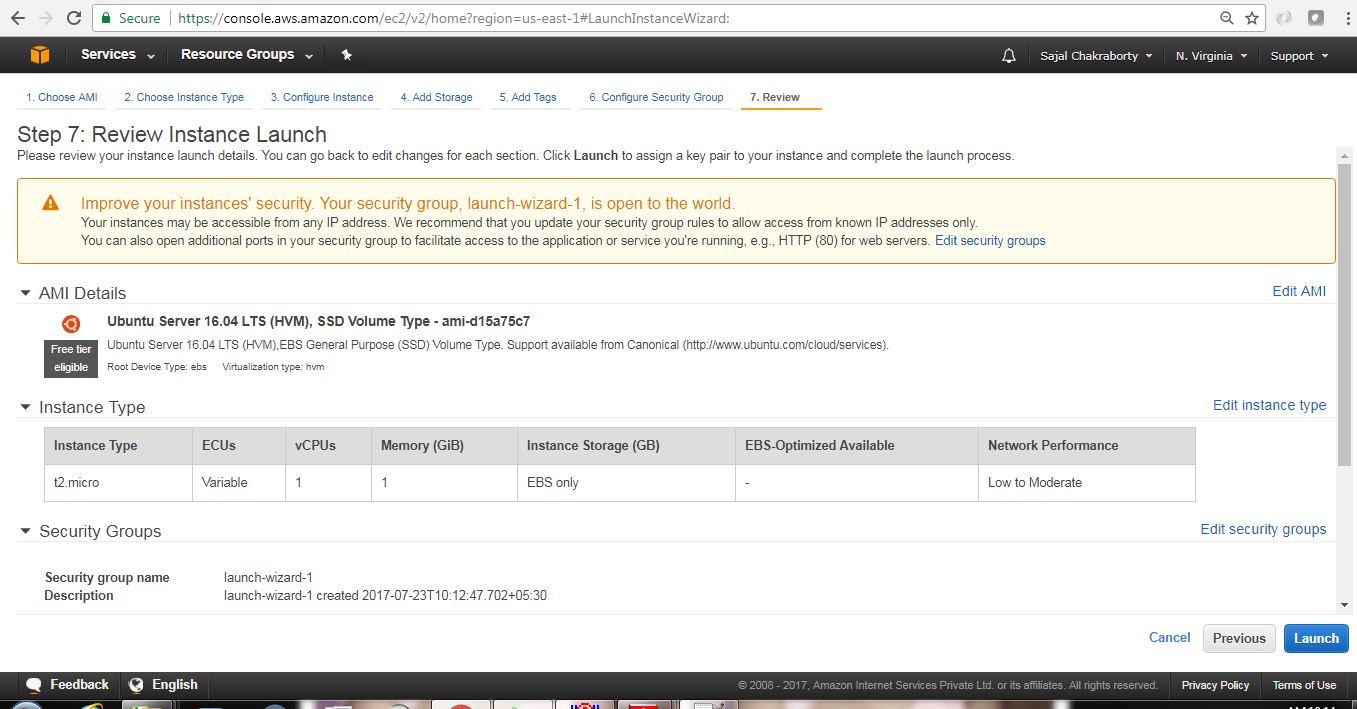
In this step we will add one tag Name with value myfirstec2server. Once tag is created, we will proceed with configuring security group by clicking the “Next: Configure Security Group” button at the bottom right corner of the page. Add Tag screen will look like:

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

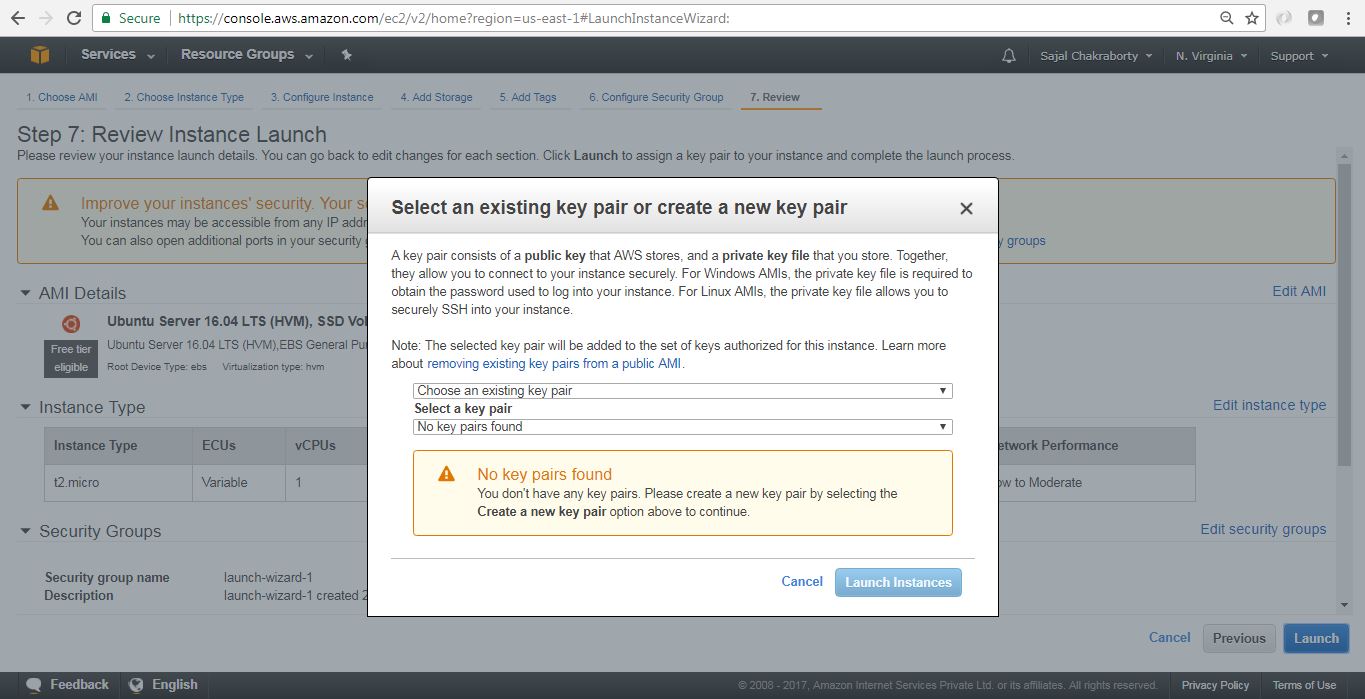
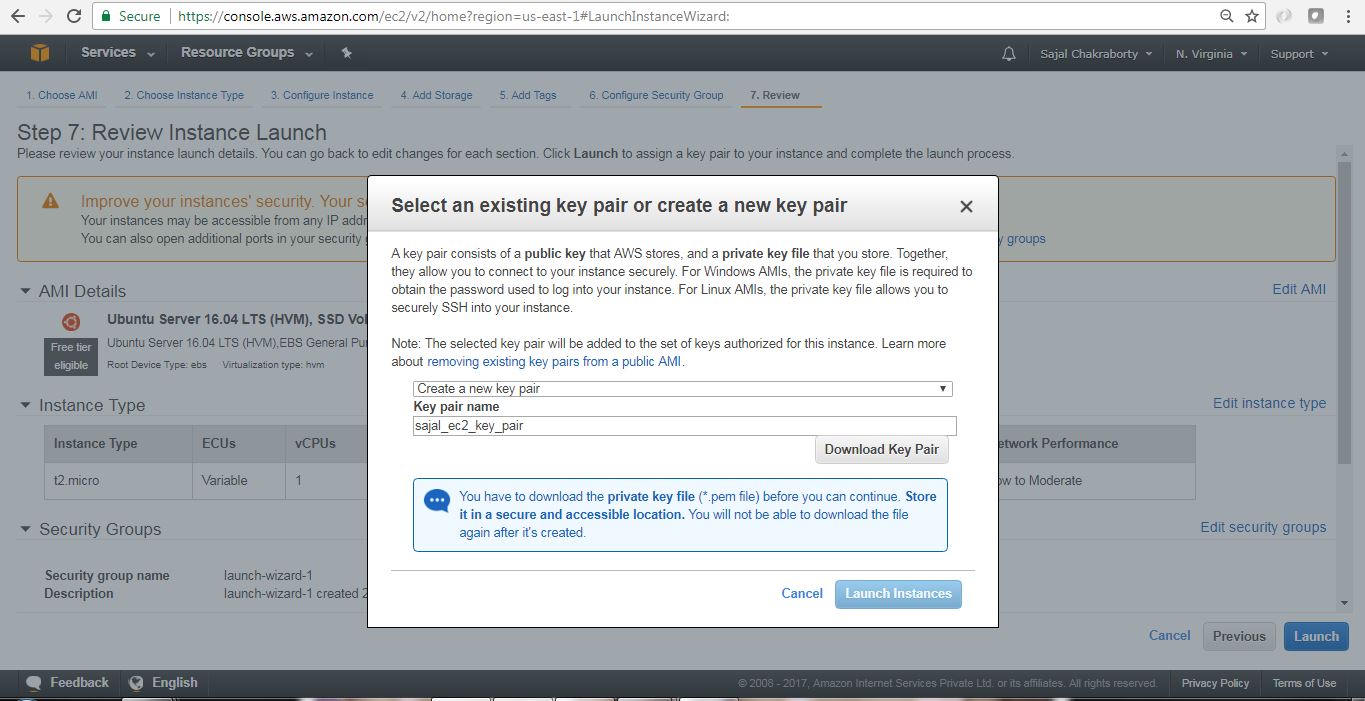
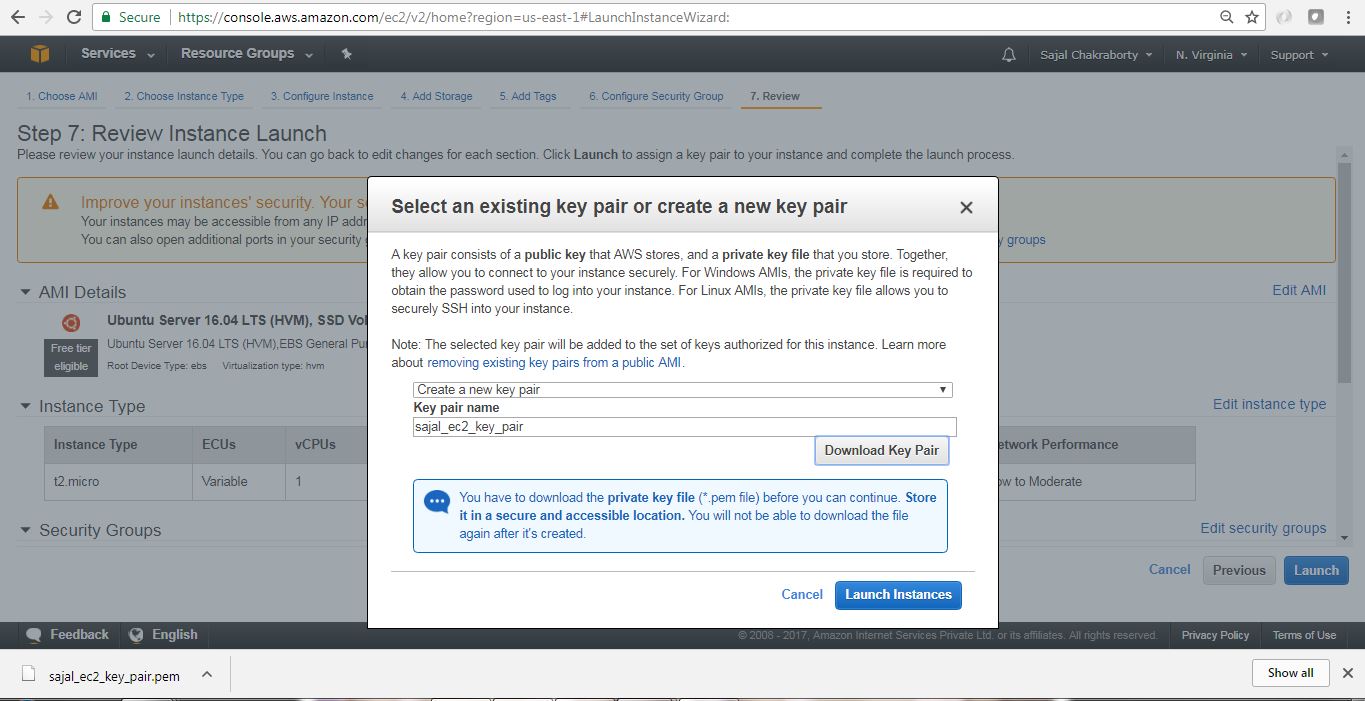
1. **Configure Security Group** – This section is to define a firewall that helps to secure our virtual server. A [security group](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-network-security.html) is a virtual stateful firewall that controls inbound and outbound network traffic to AWS 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:

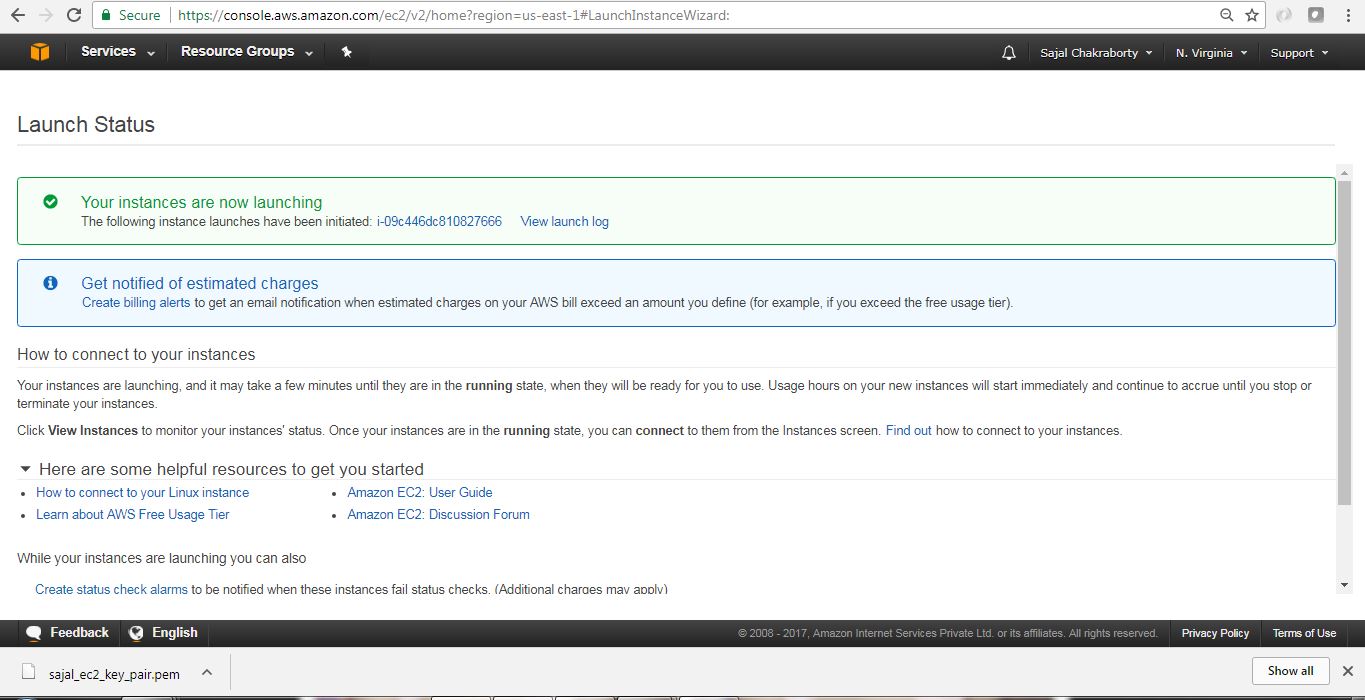
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_7.jpg)Figure 8.0 : Security Group Configuration Screen

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

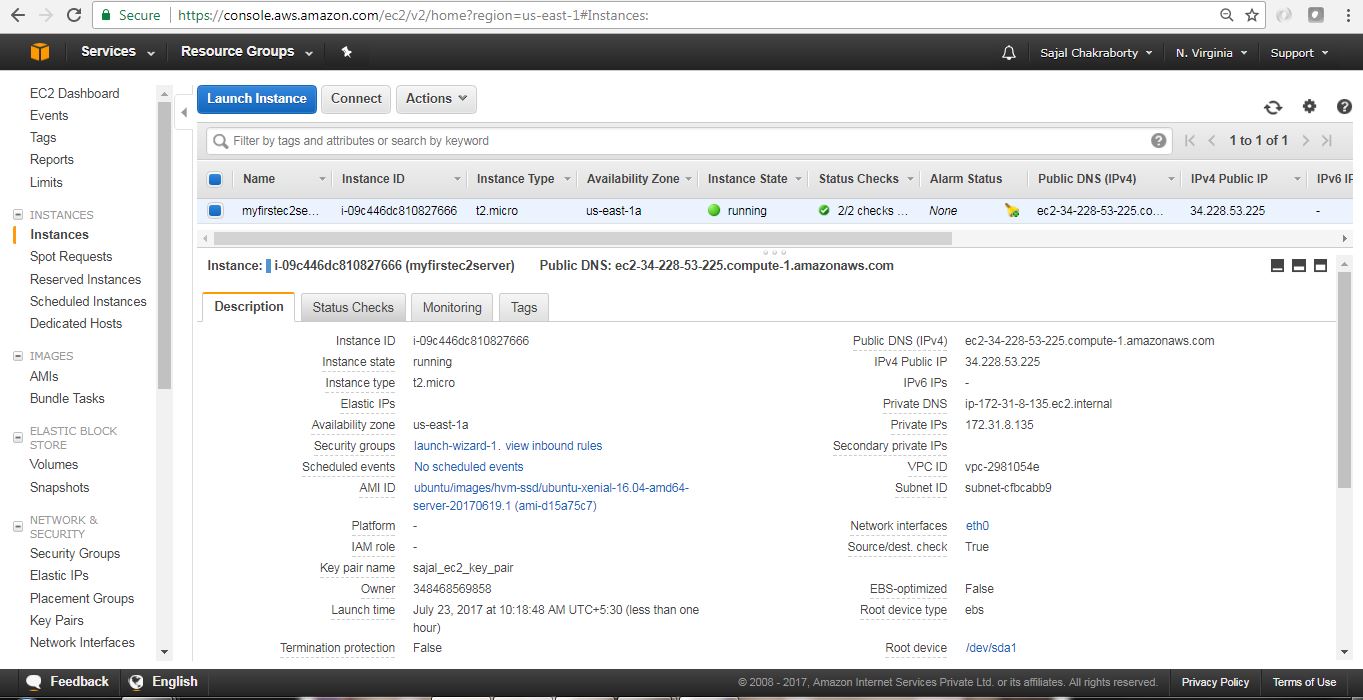
Once we proceed, we will have to select the Key pair which will be used for authentication while connecting with the instance.

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

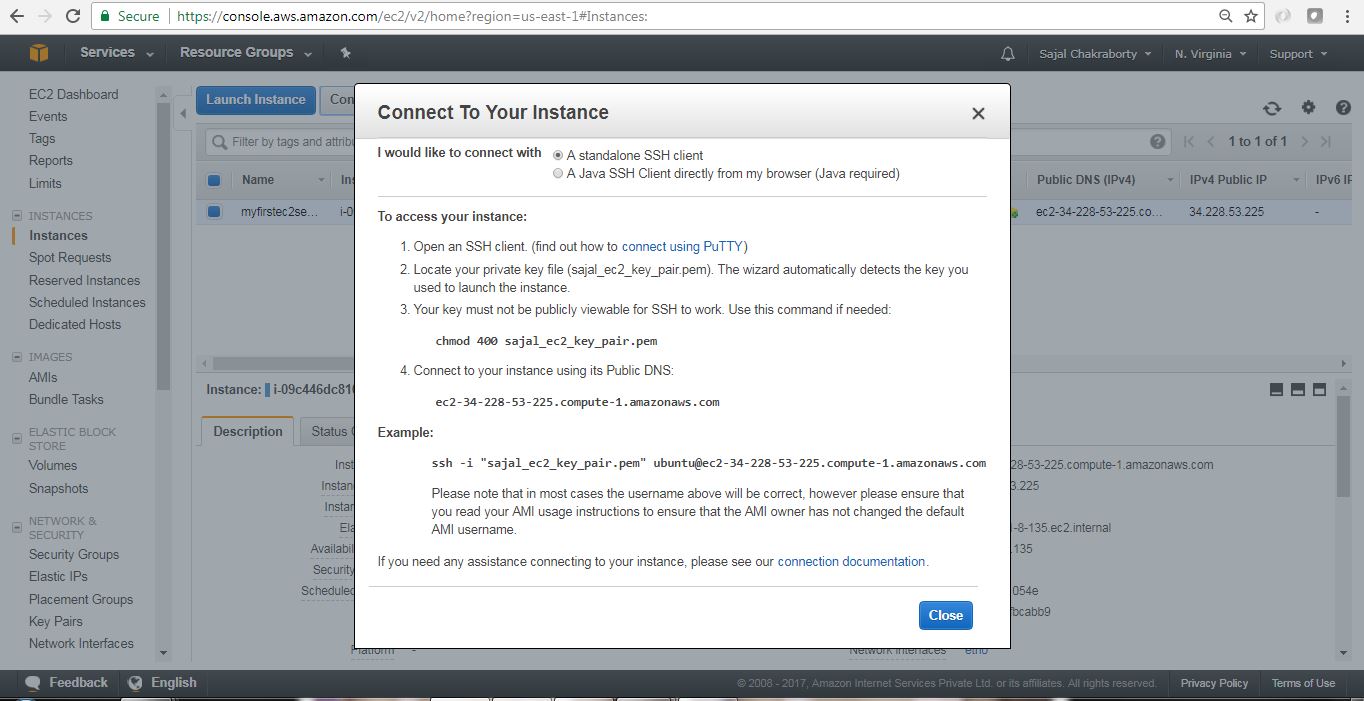
Finally this is the screen where we will see the details of the instance that we have just created.

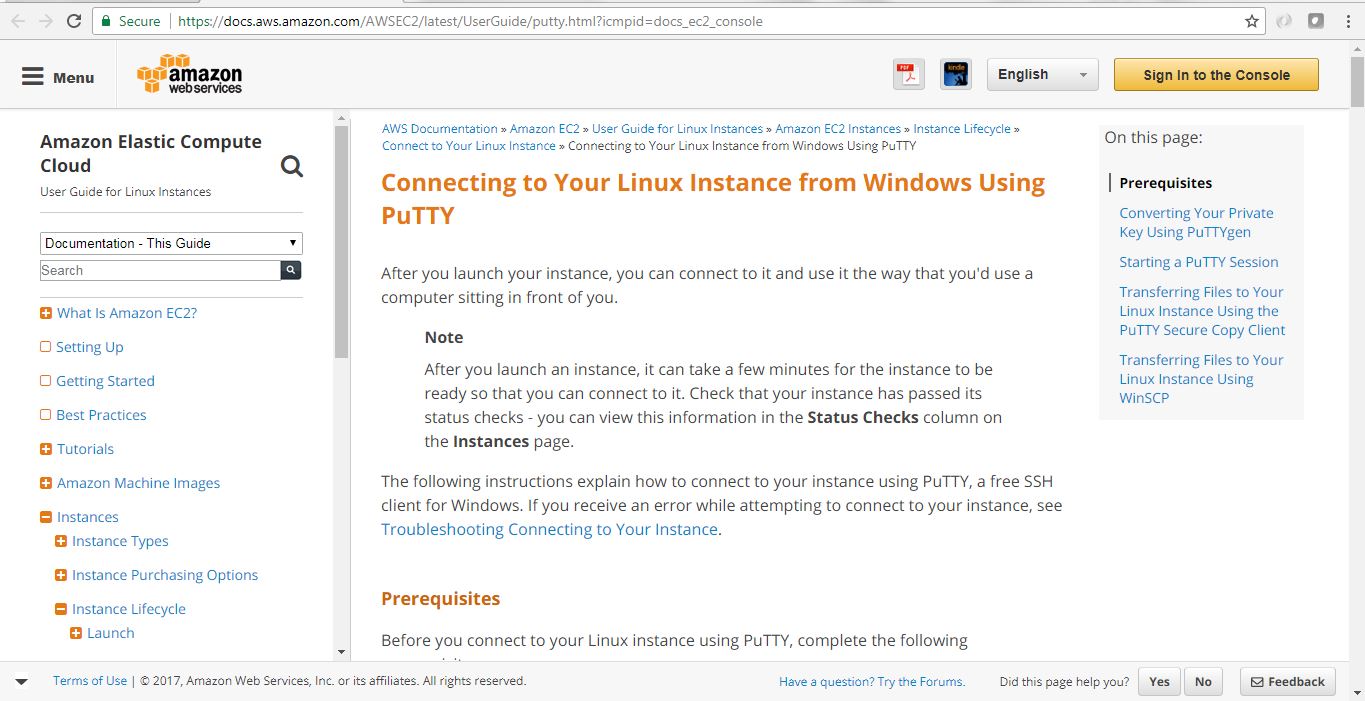
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_14.jpg)Figure 11.1 : Instance Created

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

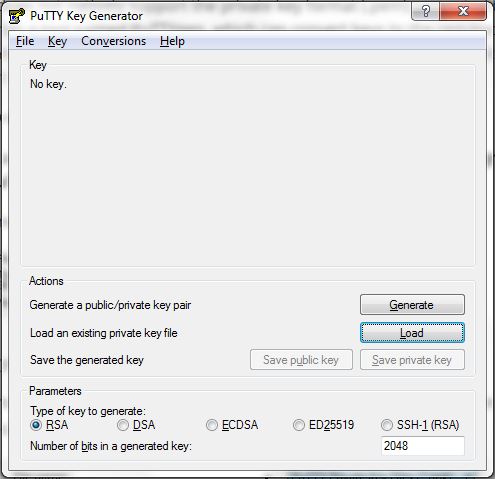
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/EC2_16.jpg)Official AWS steps for PuTTY in Windows

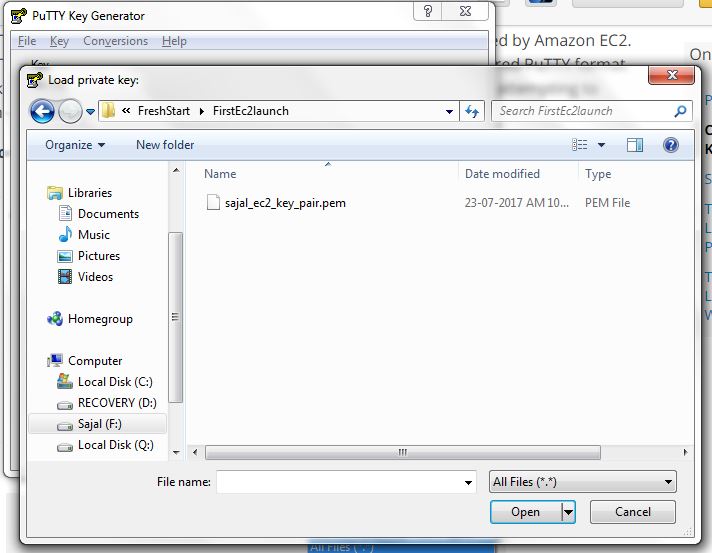
* **Use PuTTYgen to convert the Key File** – Now we need the .pem file we downloaded while creating EC2 instance. Now we will open PuTTYgen app to convert this .pem file to .pkk file which will be used by PuTTY to login to the instance terminal.

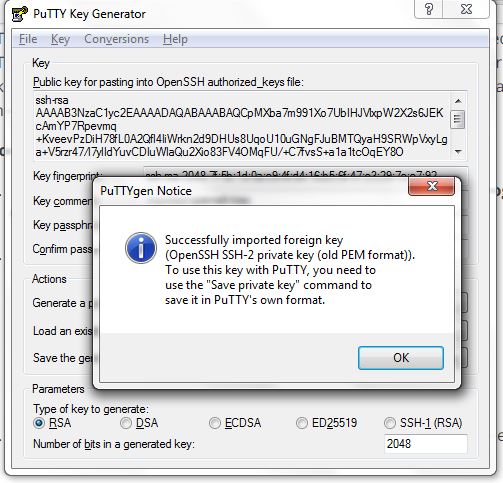
In the PuTTYgen application follow the below step to convert the .pem file.

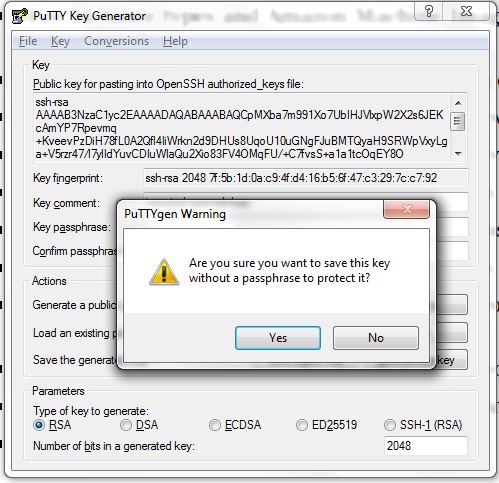
* 1. Run the application PuTTYgen.
  2. Select RSA radio button under Type of Key to Generate.
  3. Click Load.
  4. Because PuTTYgen displays only \*.pkk files, you need to switch the file extension of the File Name field to All Files.
  5. Select the .pem file [in my case it is sajal\_ec2\_key\_pair.pem],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

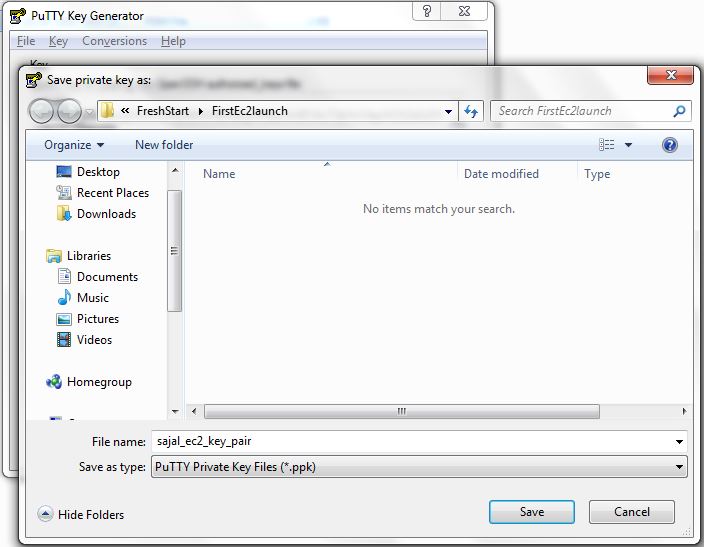
Here are few screen shots for those steps.

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

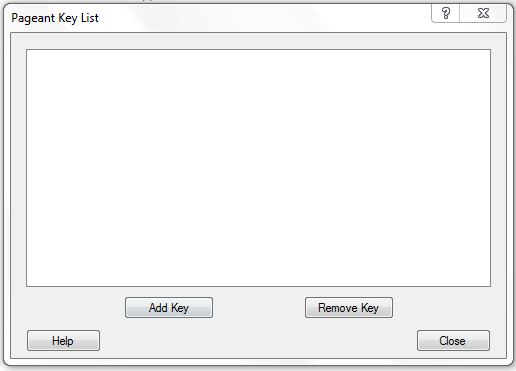
[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/putty2.jpg)

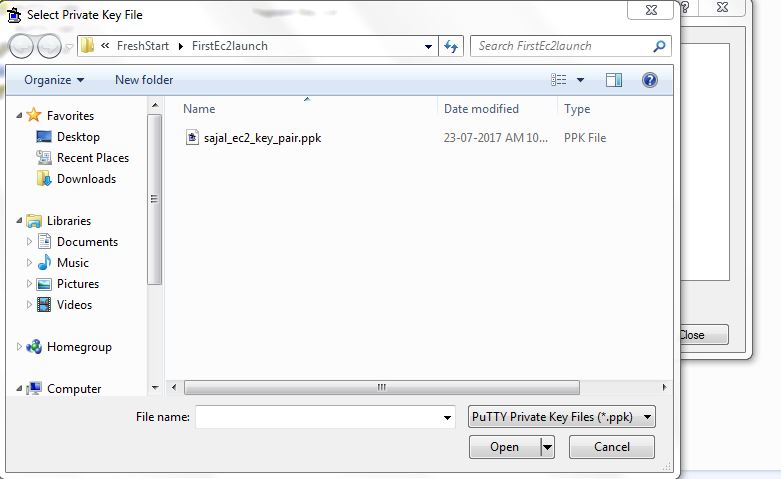
[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/putty3.jpg)

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

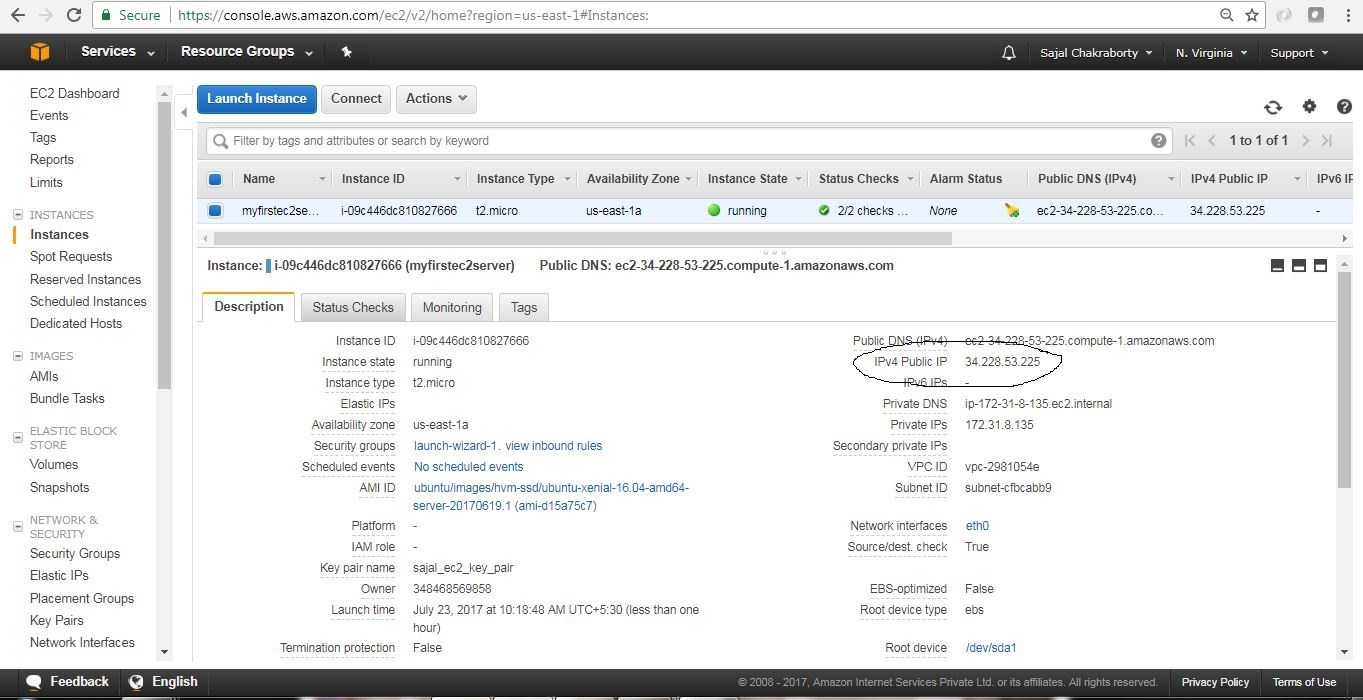
[](https://cdn2.howtodoinjava.com/wp-content/uploads/2017/07/putty5.jpg)

* 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.

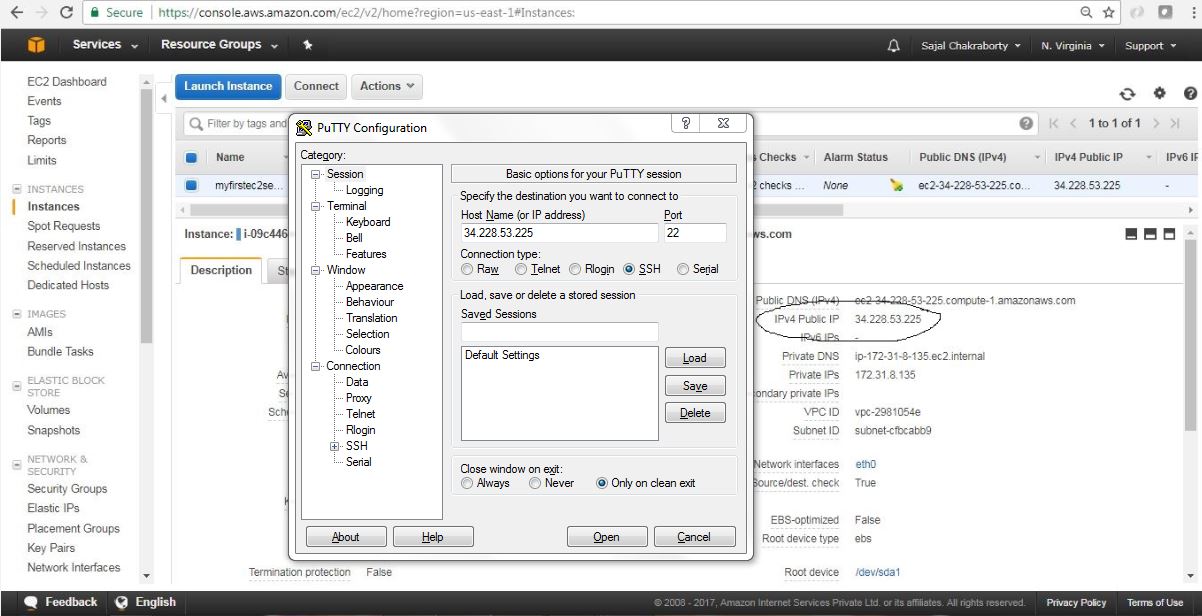
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/PAgent1.jpg)

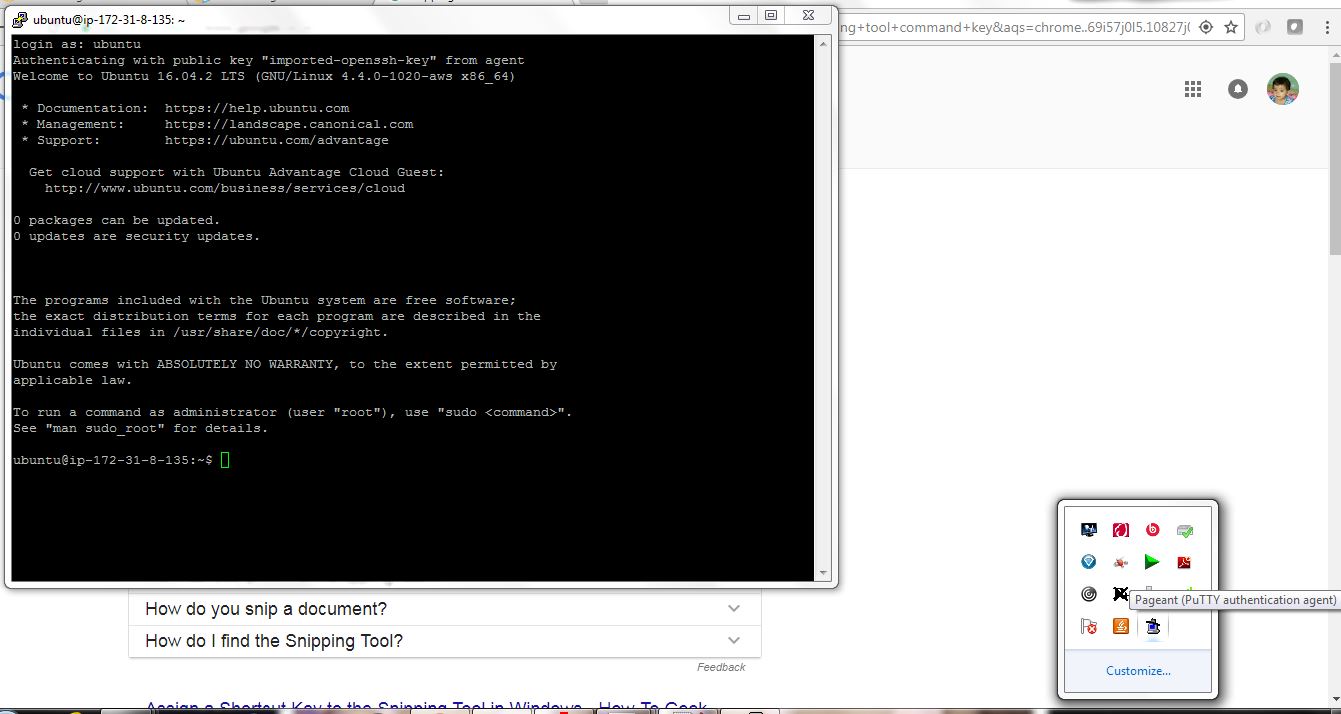
[](https://cdn1.howtodoinjava.com/wp-content/uploads/2017/07/PAgent2.jpg)

**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

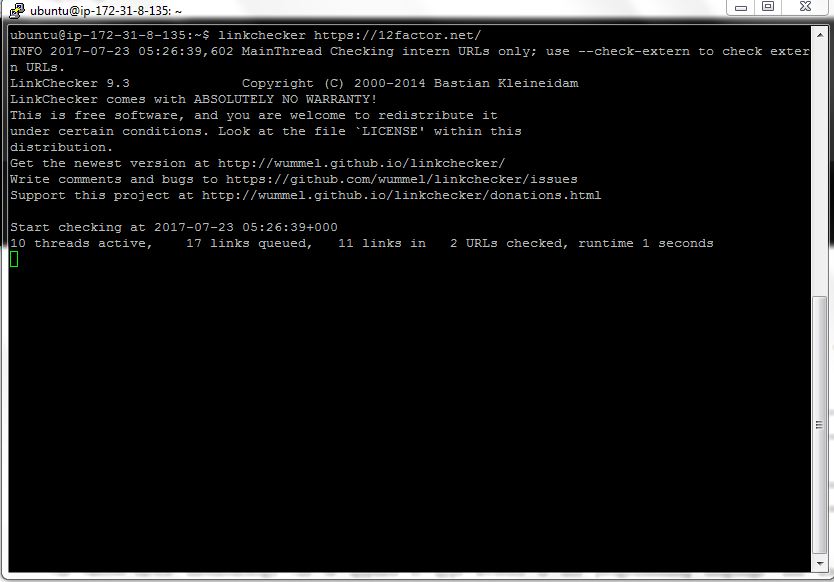
So now we are able to SSH connect to EC2 instance through putty.

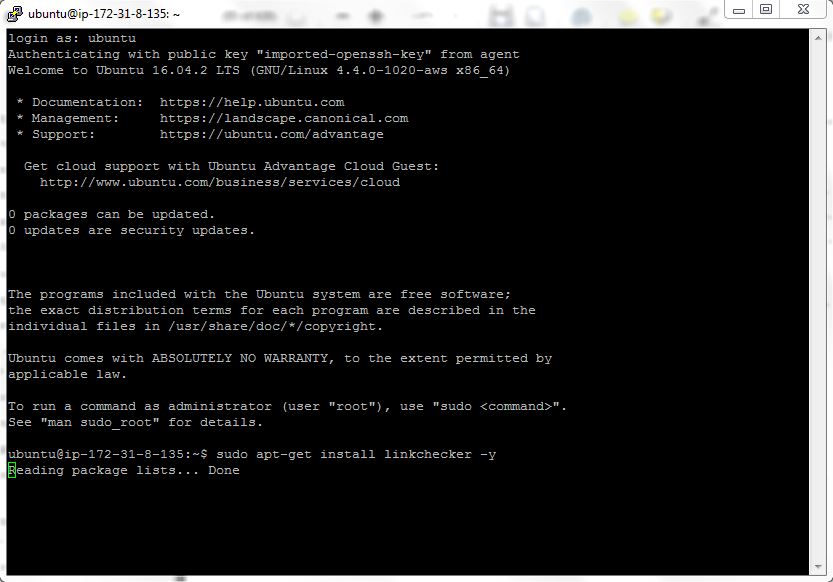
## Installing softwares on EC2 Instance

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://...

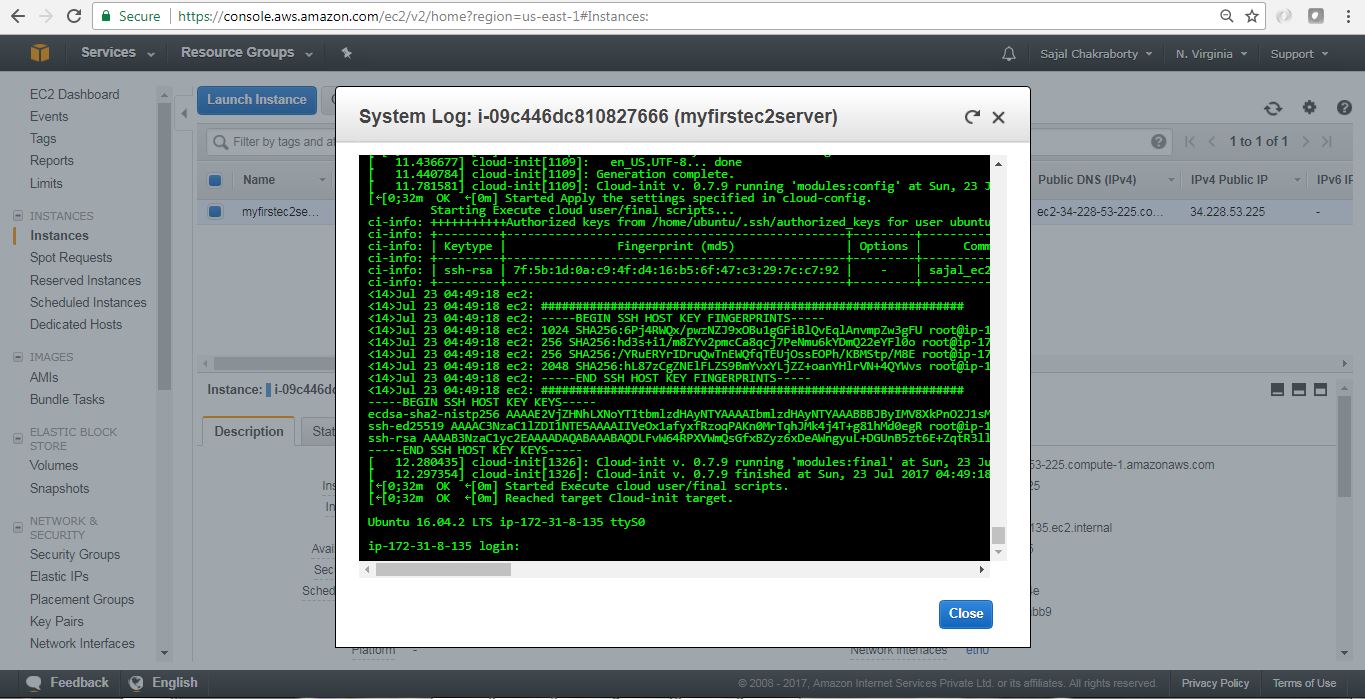
Here is the sample screen related to link checker install and usage.

[](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

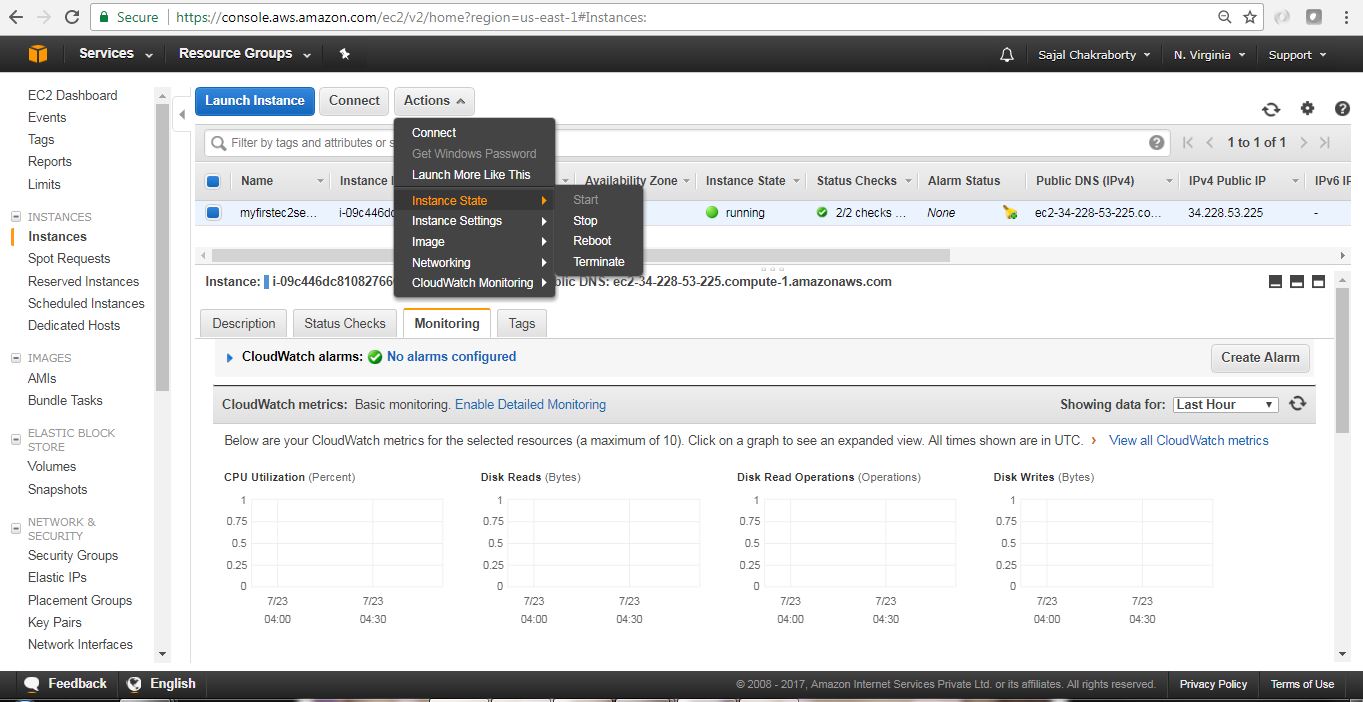
We can view the console log of EC2 instance from the AWS console itself by clicking on the **Actions**menu, choose **Instance Settings > Get System Log**. Here is the sample screen looks like:

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

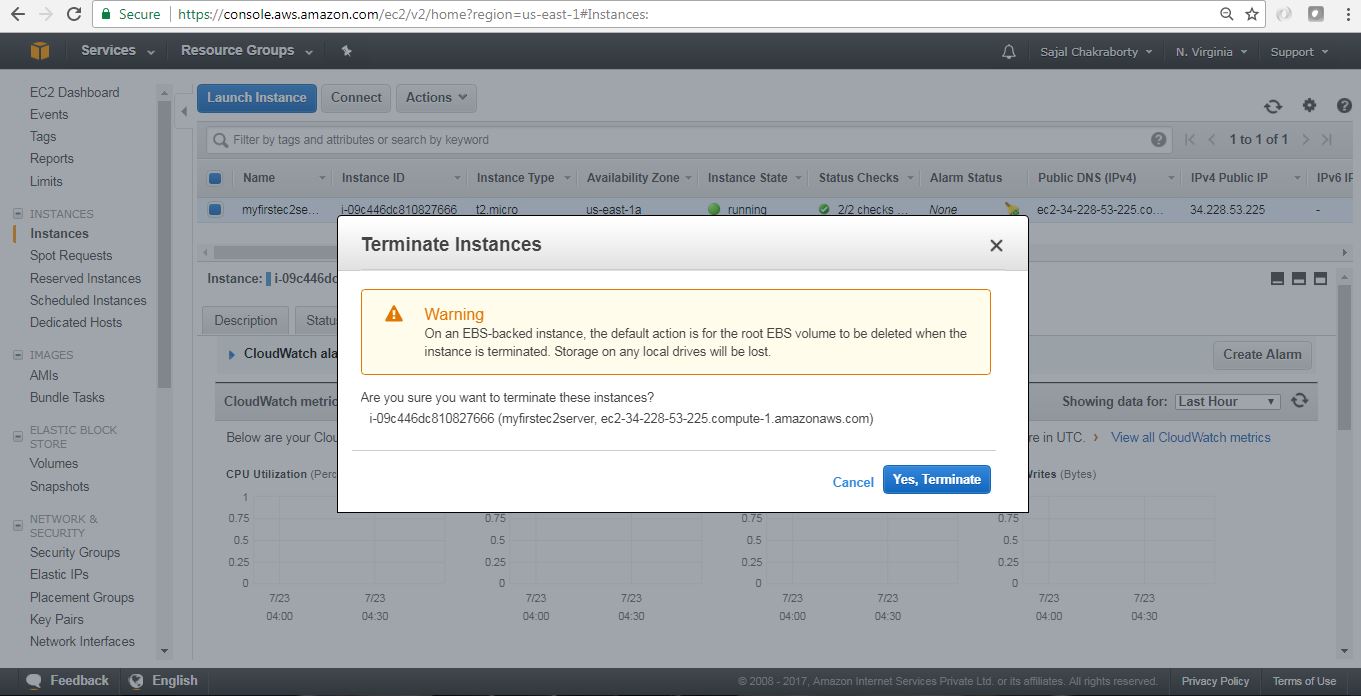
## How to Terminate EC2 Instance

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

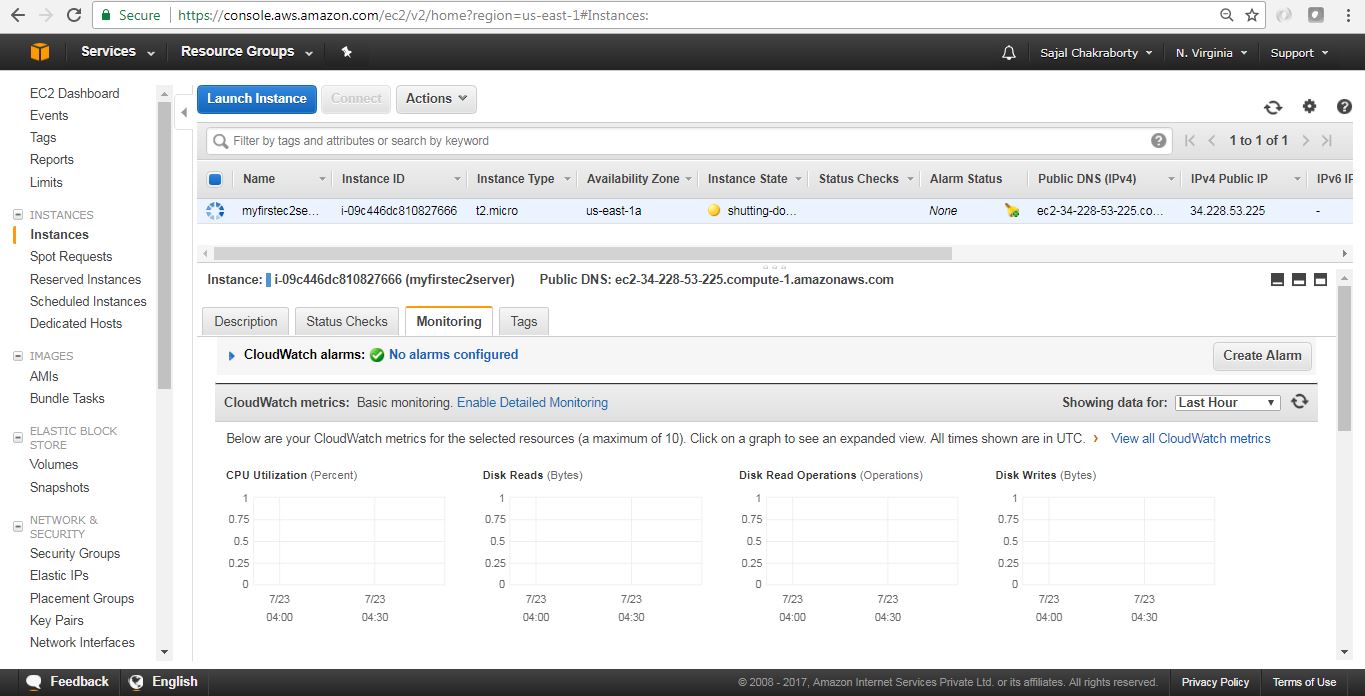
Here are the consequent steps related to termination of the instance. Please note that termination means Deletion/removal of the instance from AWS system. Also we can stop the instance. Please choose accordingly.

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

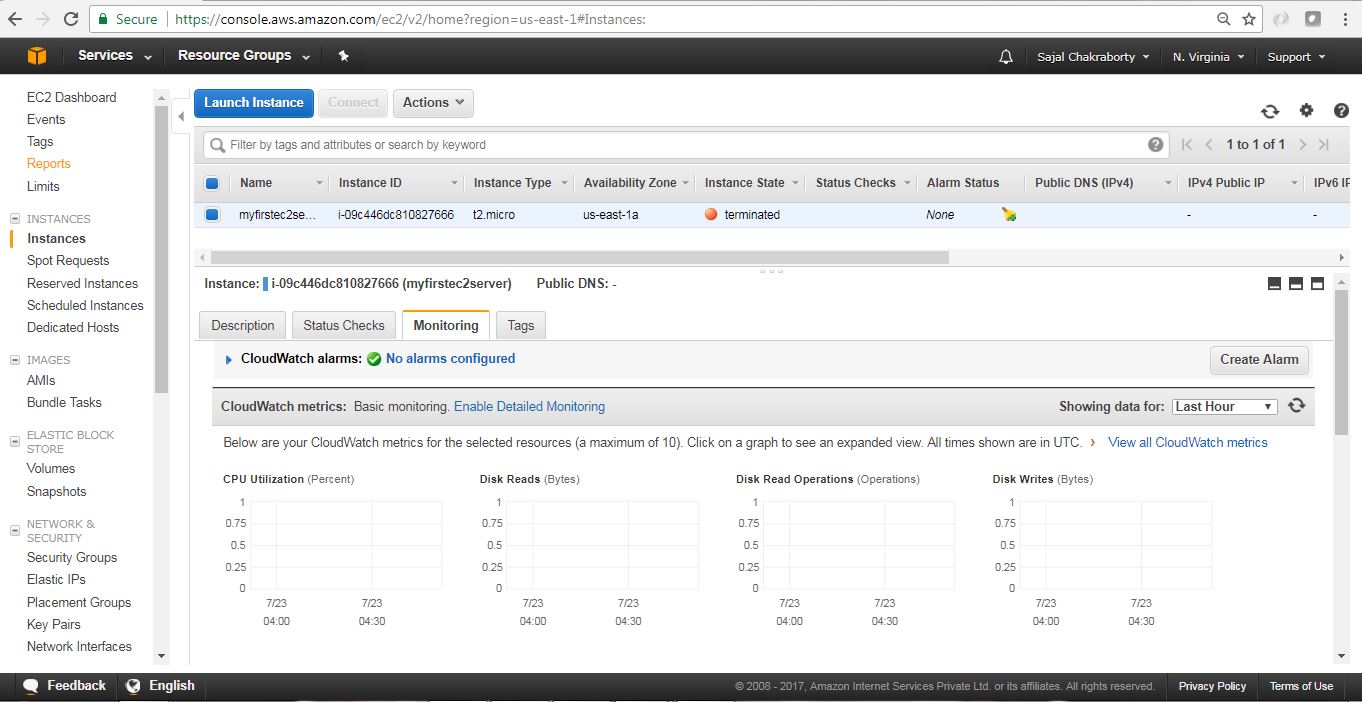
EC2 Instance Termination

[](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).

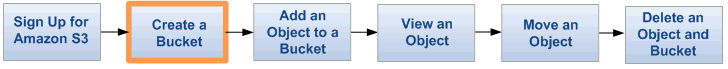
# What Is Amazon S3?

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.

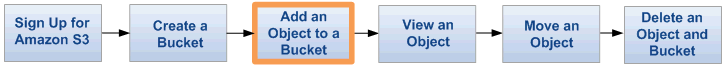
For information about naming buckets, see [Rules for Bucket Naming](https://docs.aws.amazon.com/AmazonS3/latest/dev/BucketRestrictions.html#bucketnamingrules) in the Amazon Simple Storage Service Developer Guide.

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.

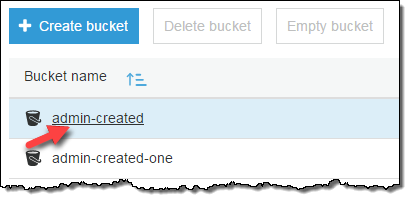
# Add an Object to a Bucket



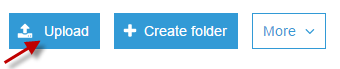
Now that you've created a bucket, you're ready to add an object to it. An object can be any kind of file: a text file, a photo, a video, and so on.

**To upload an object to a bucket**

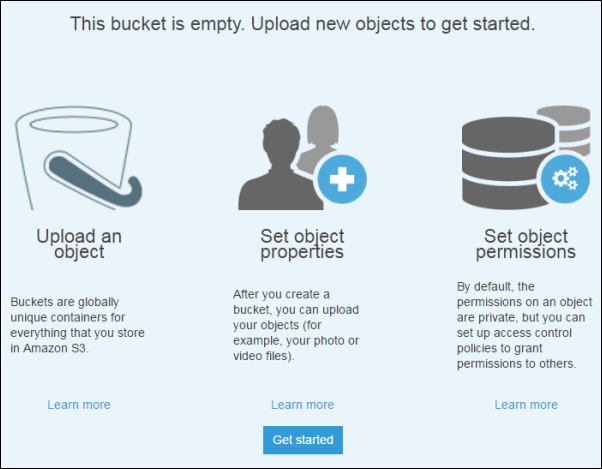
1. In the **Bucket name** list, choose the name of the bucket that you want to upload your object to.



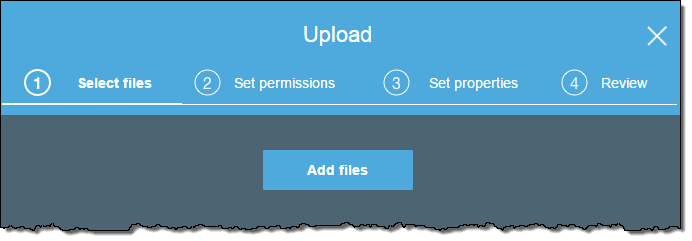
1. Choose **Upload**.



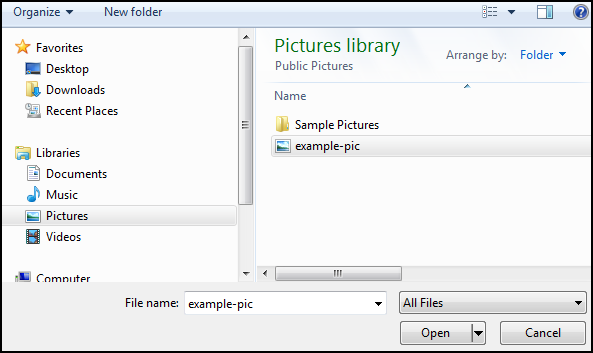
* 1. Or you can choose **Get started**.



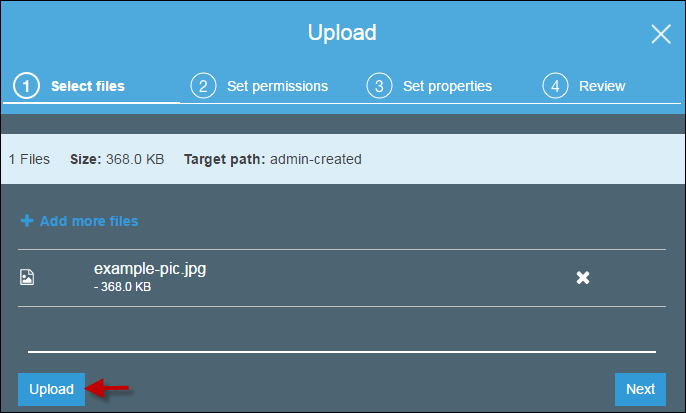
1. In the **Upload** dialog box, choose **Add files** to choose the file to upload.



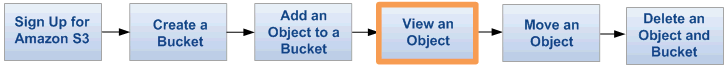
1. Choose a file to upload, and then choose **Open.**



1. Choose **Upload**.



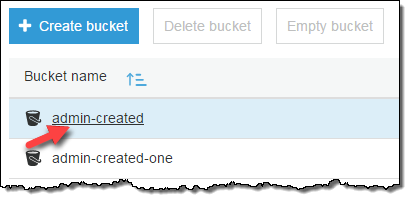
# View an Object



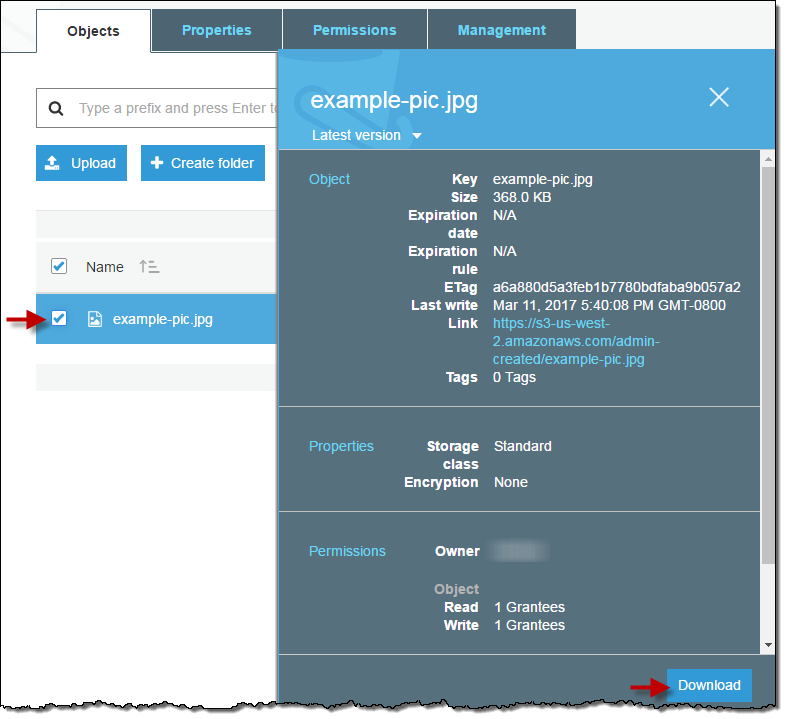
Now that you've added an object to a bucket, you can view information about your object and download the object to your local computer.

**To download an object from a bucket**

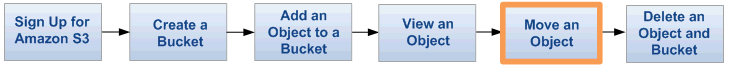
1. In the **Bucket name** list, choose the name of the bucket that you created.



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.

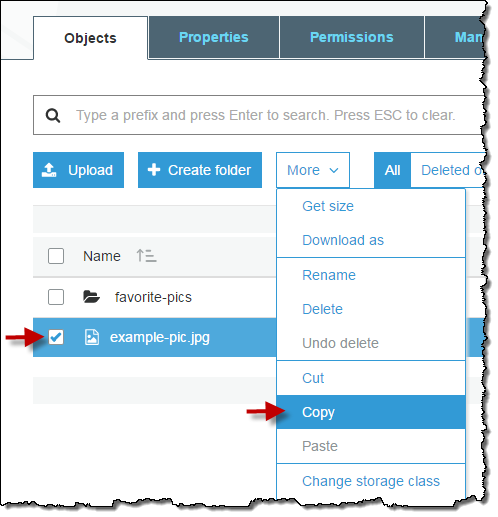
**To copy an object**

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

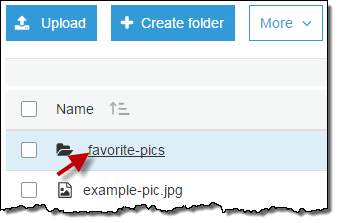

          Type the name of the folder in the Amazon S3 console.
        
          Choose the name of the bucket that you created in the Amazon S3 console.
        

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**.

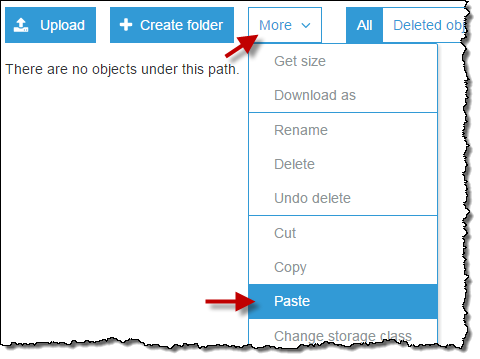
1. In the **Name** list, select the check box next to the object that you want to copy, choose **More**, and then choose **Copy**.



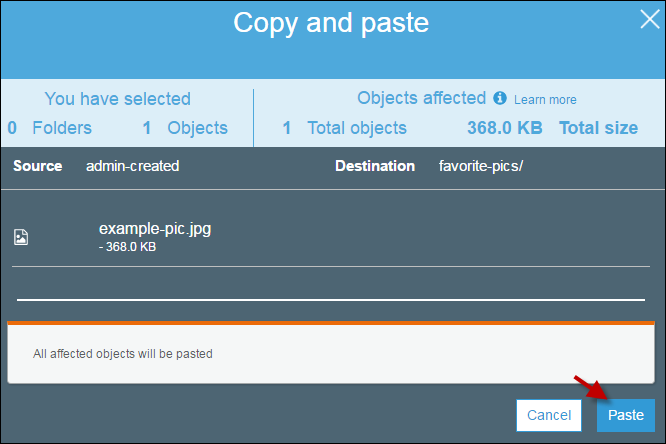
1. In the **Name** list, choose the name of the folder **favorite-pics**.



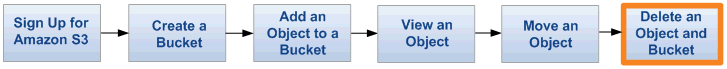
1. Choose **More**, and then choose **Paste**.



1. Choose **Paste**.



# Delete an Object and Bucket



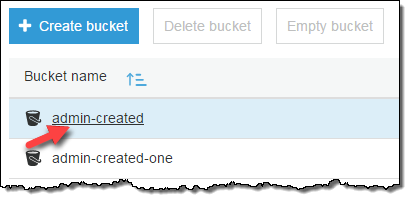
If you no longer need to store the object that you uploaded and made a copy of while going through this guide, you should delete the objects to prevent further charges.

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

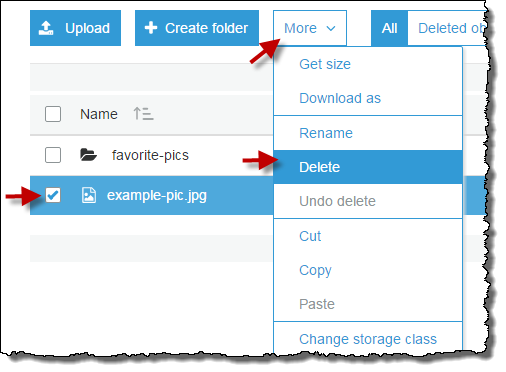
You can also delete a bucket and all the objects contained in the bucket. However, if you want to continue to use the same bucket name, don't delete the bucket. We recommend that you empty the bucket and keep it. After a bucket is deleted, the name becomes available to reuse, but the name might not be 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**

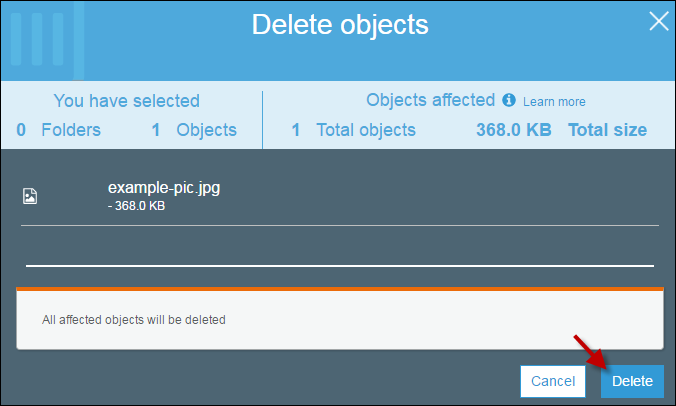
1. In the **Bucket name** list, choose the name of the bucket that you want to delete an object from.



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



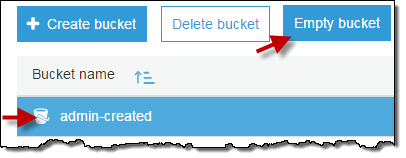
1. In the **Delete objects** dialog box, verify that the name of the object you selected for deletion is listed, and then choose **Delete**.



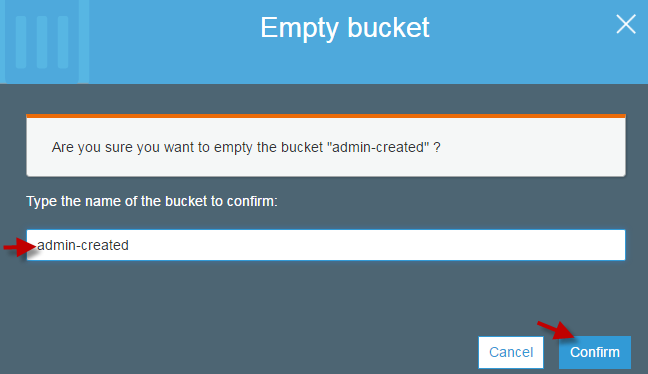
You can empty a bucket, which deletes all the objects in the bucket without deleting the bucket.

**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**.



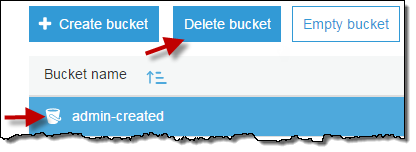
You can delete a bucket and all the objects contained in the bucket.

**Important**

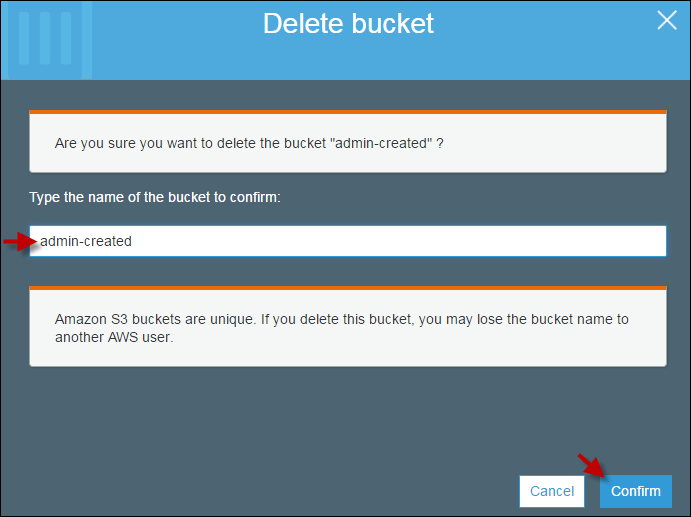
If you want to continue to use the same bucket name, don't delete the bucket. We recommend that you empty the bucket and keep it. After a bucket is deleted, the name becomes available to reuse, but the name might not be available for you to reuse for various reasons.

**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.

## 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/).

# 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.

# 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).

## 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).

# Amazon EBS Volumes

An Amazon EBS volume is a durable, block-level storage device that you can attach to a single EC2 instance. You can use EBS volumes as primary storage for data that requires frequent updates, such as the system drive for an instance or storage for a database application. You can also use them for throughput-intensive applications that perform continuous disk scans. EBS volumes persist independently from the running life of an EC2 instance.

After a volume is attached to an instance, you can use it like any other physical hard drive. EBS volumes are flexible. For current-generation volumes attached to current-generation instance types, you can dynamically increase size, modify the provisioned IOPS capacity, and change volume type on live production volumes.

Amazon EBS provides the following volume types: General Purpose SSD (gp2), Provisioned IOPS SSD (io1), Throughput Optimized HDD (st1), Cold HDD (sc1), and Magnetic (standard, a previous-generation type). They differ in performance characteristics and price, allowing you to tailor your storage performance and cost to the needs of your applications. For more information, see [Amazon EBS Volume Types](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html).

## Benefits of Using EBS Volumes

EBS volumes provide several benefits that are not supported by instance store volumes.

* **Data availability**

When you create an EBS volume in an Availability Zone, it is automatically replicated within that zone to prevent data loss due to failure of any single hardware component. After you create a volume, you can attach it to any EC2 instance in the same Availability Zone. After you attach a volume, it appears as a native block device similar to a hard drive or other physical device. At that point, the instance can interact with the volume just as it would with a local drive. The instance can format the EBS volume with a file system, such as ext3, and then install applications.

An EBS volume can be attached to only one instance at a time, but multiple volumes can be attached to a single instance. If you attach multiple volumes to a device that you have named, you can stripe data across the volumes for increased I/O and throughput performance.

An EBS volume and the instance to which it attaches must be in the same Availability Zone.

You can get monitoring data for your EBS volumes, including root device volumes for EBS-backed instances, at no additional charge. For more information about monitoring metrics, see [Monitoring Volumes with CloudWatch](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring-volume-status.html#using_cloudwatch_ebs). For information about tracking the status of your volumes, see [Amazon CloudWatch Events for Amazon EBS](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-cloud-watch-events.html).

* **Data persistence**

An EBS volume is off-instance storage that can persist independently from the life of an instance. You continue to pay for the volume usage as long as the data persists.

By default, EBS volumes that are attached to a running instance automatically detach from the instance with their data intact when that instance is terminated. The volume can then be reattached to a new instance, enabling quick recovery. If you are using an EBS-backed instance, you can stop and restart that instance without affecting the data stored in the attached volume. The volume remains attached throughout the stop-start cycle. This enables you to process and store the data on your volume indefinitely, only using the processing and storage resources when required. The data persists on the volume until the volume is deleted explicitly. The physical block storage used by deleted EBS volumes is overwritten with zeroes before it is allocated to another account. If you are dealing with sensitive data, you should consider encrypting your data manually or storing the data on a volume protected by Amazon EBS encryption. For more information, see [Amazon EBS Encryption](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html).

By default, EBS volumes that are created and attached to an instance at launch are deleted when that instance is terminated. You can modify this behavior by changing the value of the flag DeleteOnTermination to falsewhen you launch the instance. This modified value causes the volume to persist even after the instance is terminated, and enables you to attach the volume to another instance.

* **Data encryption**

For simplified data encryption, you can create encrypted EBS volumes with the Amazon EBS encryption feature. All EBS volume types support encryption. You can use encrypted EBS volumes to meet a wide range of data-at-rest encryption requirements for regulated/audited data and applications. Amazon EBS encryption uses 256-bit Advanced Encryption Standard algorithms (AES-256) and an Amazon-managed key infrastructure. The encryption occurs on the server that hosts the EC2 instance, providing encryption of data-in-transit from the EC2 instance to Amazon 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 AWS KMS. Creating your own CMK gives you more flexibility, including the ability to create, rotate, disable, define access controls, and audit the encryption keys used to protect your data. For more information, see the [AWS Key Management Service Developer Guide](https://docs.aws.amazon.com/kms/latest/developerguide/).

* **Snapshots**

Amazon EBS provides the ability to create snapshots (backups) of any EBS volume and write a copy of the data in the volume to Amazon S3, where it is stored redundantly in multiple Availability Zones. The volume does not need to be attached to a running instance in order to take a snapshot. As you continue to write data to a volume, you can periodically create a snapshot of the volume to use as a baseline for new volumes. These snapshots can be used to create multiple new EBS volumes or move volumes across Availability Zones. Snapshots of encrypted EBS volumes are automatically encrypted.

When you create a new volume from a snapshot, it's an exact copy of the original volume at the time the snapshot was taken. EBS volumes that are restored from encrypted snapshots are automatically encrypted. By optionally specifying a different Availability Zone, you can use this functionality to create a duplicate volume in that zone. The snapshots can be shared with specific AWS accounts or made public. When you create snapshots, you incur charges in Amazon S3 based on the volume's total size. For a successive snapshot of the volume, you are only charged for any additional data beyond the volume's original size.

Snapshots are incremental backups, meaning that only the blocks on the volume that have changed after your most recent snapshot are saved. If you have a volume with 100 GiB of data, but only 5 GiB of data have changed since your last snapshot, only the 5 GiB of modified data is written to Amazon S3. Even though snapshots are saved incrementally, the snapshot deletion process is designed so that you need to retain only the most recent snapshot in order to restore the volume.

To help categorize and manage your volumes and snapshots, you can tag them with metadata of your choice. For more information, see [Tagging Your Amazon EC2 Resources](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Using_Tags.html).

* **Flexibility**

EBS volumes support live configuration changes while in production. You can modify volume type, volume size, and IOPS capacity without service interruptions.

# Creating an Amazon EBS Volume

You can create an Amazon EBS volume that you can then attach to any EC2 instance within the same Availability Zone. You can choose to create an encrypted EBS volume, but encrypted volumes can only be attached to selected instance types. For more information, see [Supported Instance Types](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html#EBSEncryption_supported_instances). You can use IAM policies to enforce encryption on new volumes. For more information, see the example IAM policies in [Working with Volumes](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ExamplePolicies_EC2.html#iam-example-manage-volumes) and [Launching Instances (RunInstances)](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ExamplePolicies_EC2.html#iam-example-runinstances).

You can also create and attach EBS volumes when you launch instances by specifying a block device mapping. For more information, see [Launching an Instance Using the Launch Instance Wizard](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/launching-instance.html) and [Block Device Mapping](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/block-device-mapping-concepts.html). You can restore volumes from previously created snapshots. For more information, see [Restoring an Amazon EBS Volume from a Snapshot](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-restoring-volume.html).

You can apply tags to EBS volumes at the time of creation. With tagging, you can simplify tracking of your Amazon EC2 resource inventory. Tagging on creation can be combined with an IAM policy to enforce tagging on new volumes. For more information, see [Tagging Your Resources](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Using_Tags.html).

If you are creating a volume for a high-performance storage scenario, you should make sure to use a Provisioned IOPS SSD (io1) volume and attach it to an instance with enough bandwidth to support your application, such as an EBS-optimized instance or an instance with 10-Gigabit network connectivity. The same advice holds for Throughput Optimized HDD (st1) and Cold HDD (sc1) volumes. For more information, see [Amazon EC2 Instance Configuration](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-ec2-config.html).

New EBS volumes receive their maximum performance the moment that they are available and do not require initialization (formerly known as pre-warming). However, storage blocks on volumes that were restored from snapshots must be initialized (pulled down from Amazon S3 and written to the volume) before you can access the block. This preliminary action takes time and can cause a significant increase in the latency of an I/O operation the first time each block is accessed. For most applications, amortizing this cost over the lifetime of the volume is acceptable. Performance is restored after the data is accessed once. For more information, see [Initializing Amazon EBS Volumes](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-initialize.html).

**To create an EBS volume using the console**

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. From the navigation bar, select the region in which you would like to create your volume. This choice is important because some Amazon EC2 resources can be shared between regions, while others can't. For more information, see [Resource Locations](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/resources.html).
3. In the navigation pane, choose **ELASTIC BLOCK STORE**, **Volumes**.
4. Choose **Create Volume**.
5. For **Volume Type**, choose a volume type. For more information, see [Amazon EBS Volume Types](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html).

**Note**

Some AWS accounts created before 2012 might have access to Availability Zones in us-west-1 or ap-northeast-1 that do not support Provisioned IOPS SSD (io1) volumes. If you are unable to create an io1 volume (or launch an instance with an io1 volume in its block device mapping) in one of these regions, try a different Availability Zone in the region. You can verify that an Availability Zone supports io1 volumes by creating a 4 GiB io1 volume in that zone.

1. For **Size (GiB)**, type the size of the volume.
2. With a Provisioned IOPS SSD volume, for **IOPS**, type the maximum number of input/output operations per second (IOPS) that the volume should support.
3. For **Availability Zone**, choose the Availability Zone in which to create the volume. EBS volumes can only be attached to EC2 instances within the same Availability Zone.
4. (Optional) To create an encrypted volume, select the **Encrypted** box and choose the master key you want to use when encrypting the volume. You can choose the default master key for your account, or you can choose any customer master key (CMK) that you have previously created using the AWS Key Management Service. Available keys are visible in the **Master Key** menu, or you can paste the full ARN of any key that you have access to. For more information, see the [AWS Key Management Service Developer Guide](https://docs.aws.amazon.com/kms/latest/developerguide/).

**Note**

Encrypted volumes can only be attached to selected instance types. For more information, see [Supported Instance Types](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html#EBSEncryption_supported_instances).

1. (Optional) Choose **Create additional tags** to add tags to the volume. For each tag, provide a tag key and a tag value.
2. Choose **Create Volume**.

**To create an EBS volume using the command line**

You can use one of the following commands. For more information about these command line interfaces, see [Accessing Amazon EC2](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html#access-ec2).

* [create-volume](https://docs.aws.amazon.com/cli/latest/reference/ec2/create-volume.html) (AWS CLI)
* [New-EC2Volume](https://docs.aws.amazon.com/powershell/latest/reference/items/New-EC2Volume.html) (AWS Tools for Windows PowerShell)

# Attaching an Amazon EBS Volume to an Instance

You can attach an available EBS volume to one of your instances that is in the same Availability Zone as the volume.

**Prerequisites**

* Determine how many volumes you can attach to your instance. For more information, see [Instance Volume Limits](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/volume_limits.html).
* If a volume is encrypted, it can only be attached to an instance that supports Amazon EBS encryption. For more information, see [Supported Instance Types](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html#EBSEncryption_supported_instances).
* If a volume has an AWS Marketplace product code:
  + The volume can only be attached to a stopped instance.
  + You must be subscribed to the AWS Marketplace code that is on the volume.
  + The configuration (instance type, operating system) of the instance must support that specific AWS Marketplace code. For example, you cannot take a volume from a Windows instance and attach it to a Linux instance.
  + AWS Marketplace product codes are copied from the volume to the instance.

**To attach an EBS volume to an instance using the console**

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the navigation pane, choose **Elastic Block Store**, **Volumes**.
3. Select an available volume and choose **Actions**, **Attach Volume**.
4. For **Instance**, start typing the name or ID of the instance. Select the instance from the list of options (only instances that are in the same Availability Zone as the volume are displayed).
5. For **Device**, you can keep the suggested device name, or type a different supported device name. For more information, see [Device Naming on Linux Instances](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/device_naming.html).
6. Choose **Attach**.
7. Connect to your instance and mount the volume. For more information, see [Making an Amazon EBS Volume Available for Use on Linux](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-using-volumes.html).

**To attach an EBS volume to an instance using the command line**

You can use one of the following commands. For more information about these command line interfaces, see [Accessing Amazon EC2](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html#access-ec2).

* [attach-volume](https://docs.aws.amazon.com/cli/latest/reference/ec2/attach-volume.html) (AWS CLI)
* [Add-EC2Volume](https://docs.aws.amazon.com/powershell/latest/reference/items/Add-EC2Volume.html) (AWS Tools for Windows PowerShell)

# Making an Amazon EBS Volume Available for Use on Linux

After you attach an Amazon EBS volume to your instance, it is exposed as a block device. You can format the volume with any file system and then mount it. After you make the EBS volume available for use, you can access it in the same ways that you access any other volume. Any data written to this file system is written to the EBS volume and is transparent to applications using the device.

You can take snapshots of your EBS volume for backup purposes or to use as a baseline when you create another volume. For more information, see [Amazon EBS Snapshots](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSSnapshots.html).

You can get directions for volumes on a Windows instance from [Making a Volume Available for Use on Windows](https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/ebs-using-volumes.html) in the Amazon EC2 User Guide for Windows Instances.

**To make an EBS volume available for use on Linux**

1. Connect to your instance using SSH. For more information, see [Connect to Your Linux Instance](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstances.html).
2. Depending on the block device driver of the kernel, the device could be attached with a different name than you specified. For example, if you specify a device name of /dev/sdh, your device could be renamed /dev/xvdh or /dev/hdh. In most cases, the trailing letter remains the same. In some versions of Red Hat Enterprise Linux (and its variants, such as CentOS), even the trailing letter could change (/dev/sda could become /dev/xvde). In these cases, the trailing letter of each device name is incremented the same number of times. For example, if /dev/sdbis renamed /dev/xvdf, then /dev/sdc is renamed /dev/xvdg. Amazon Linux creates a symbolic link for the name you specified to the renamed device. Other operating systems could behave differently.

Use the **lsblk** command to view your available disk devices and their mount points (if applicable) to help you determine the correct device name to use.

[ec2-user ~]$ **lsblk**

NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT

xvdf 202:80 0 100G 0 disk

xvda1 202:1 0 8G 0 disk /

The output of **lsblk** removes the /dev/ prefix from full device paths. In this example, /dev/xvda1 is mounted as the root device (note that MOUNTPOINT is listed as /, the root of the Linux file system hierarchy), and /dev/xvdfis attached, but it has not been mounted yet.

EBS volumes are exposed as NVMe block devices on [Nitro-based instances](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-types.html#ec2-nitro-instances). The device names that you specify are renamed using NVMe device names (/dev/nvme[0-26]n1). For more information, see [Amazon EBS and NVMe](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/nvme-ebs-volumes.html).

1. Determine whether to create a file system on the volume. New volumes are raw block devices, and you must create a file system on them before you can mount and use them. Volumes that have been restored from snapshots likely have a file system on them already; if you create a new file system on top of an existing file system, the operation overwrites your data. Use the **sudo file -s *device*** command to list special information, such as file system type.
2. [ec2-user ~]$ **sudo file -s /dev/xvdf**

/dev/xvdf: data

If the output of the previous command shows simply data for the device, then there is no file system on the device and you must create one. You can go on to [Step 4](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-using-volumes.html#create_file_system_step). If you run this command on a device that contains a file system, then your output will be different.

[ec2-user ~]$ **sudo file -s /dev/xvda1**

/dev/xvda1: Linux rev 1.0 ext4 filesystem data, UUID=1701d228-e1bd-4094-a14c-8c64d6819362 (needs journal recovery) (extents) (large files) (huge files)

In the previous example, the device contains **Linux rev 1.0 ext4 filesystem data**, so this volume does not need a file system created (you can skip [Step 4](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-using-volumes.html#create_file_system_step) if your output shows file system data).

1. (Conditional) Use the following command to create an ext4 file system on the volume. Substitute the device name (such as /dev/xvdf) for *device\_name*. Depending on the requirements of your application or the limitations of your operating system, you can choose a different file system type, such as ext3 or XFS.

**Warning**

This step assumes that you're mounting an empty volume. If you're mounting a volume that already has data on it (for example, a volume that was restored from a snapshot), don't use **mkfs** before mounting the volume (skip to the next step instead). Otherwise, you'll format the volume and delete the existing data.

[ec2-user ~]$ **sudo mkfs -t ext4 *device\_name***

1. Use the following command to create a mount point directory for the volume. The mount point is where the volume is located in the file system tree and where you read and write files to after you mount the volume. Substitute a location for *mount\_point*, such as /data.

[ec2-user ~]$ **sudo mkdir *mount\_point***

1. Use the following command to mount the volume at the location you just created.

[ec2-user ~]$ **sudo mount *device\_name* *mount\_point***

1. (Optional) To mount this EBS volume on every system reboot, add an entry for the device to the /etc/fstab file.
   1. Create a backup of your /etc/fstab file that you can use if you accidentally destroy or delete this file while you are editing it.

[ec2-user ~]$ **sudo cp /etc/fstab /etc/fstab.orig**

* 1. Open the /etc/fstab file using any text editor, such as **nano** or **vim**.

**Note**

You must open the file as root or by using the **sudo** command.

* 1. Add a new line to the end of the file for your volume using the following format:

*device\_name* *mount\_point* *file\_system\_type* *fs\_mntops* *fs\_freq* *fs\_passno*

The last three fields on this line are the file system mount options, the dump frequency of the file system, and the order of file system checks done at boot time. If you don't know what these values should be, then use the values in the following example for them (defaults,nofail 0 2). For more information on/etc/fstab entries, see the **fstab** manual page (by entering **man fstab** on the command line).

You can use the system's current device name (/dev/sda1, /dev/xvda1, etc.) in /etc/fstab, but we recommend using the device's 128-bit universally unique identifier (UUID) instead. System-declared block-device names may change under a variety of circumstances, but the UUID is assigned to a volume partition when it is formatted and persists throughout the partition's service life. By using the UUID, you reduce the chances of the block-device mapping in /etc/fstab leaving the system unbootable after a hardware reconfiguration.

To find the UUID of a device, first display the available devices:

[ec2-user ~]$ **df**

The following is example output:

Filesystem 1K-blocks Used Available Use% Mounted on

/dev/xvda1 8123812 1876888 6146676 24% /

devtmpfs 500712 56 500656 1% /dev

tmpfs 509724 0 509724 0% /dev/shm

Next, continuing this example, examine the output of either of two commands to find the UUID of /dev/xvda1:

* + - **sudo file -s */dev/xvda1***
    - **ls -al /dev/disk/by-uuid/**

Assuming that you find /dev/xvda1 to have UUID de9a1ccd-a2dd-44f1-8be8-0123456abcdef, you would add the following entry to /etc/fstab to mount an ext4 file system at mount point /data:

UUID=de9a1ccd-a2dd-44f1-8be8-0123456abcdef /data ext4 defaults,nofail 0 2

**Note**

If you ever intend to boot your instance without this volume attached (for example, so this volume could move back and forth between different instances), you should add the nofailmount option that allows the instance to boot even if there are errors in mounting the volume. Debian derivatives, including Ubuntu versions earlier than 16.04, must also add the nobootwaitmount option.

* 1. After you've added the new entry to /etc/fstab, you must check that your entry works. Run the following commands to unmount the device and then mount all file systems in /etc/fstab.
  2. [ec2-user ~]$ **sudo umount /data**

[ec2-user ~]$ **sudo mount -a**

If the **mount** command does not produce an error, then your /etc/fstab file is OK and your file system will mount automatically at the next boot. If the command does produce any errors, examine the errors and try to correct your /etc/fstab.

**Warning**

Errors in the /etc/fstab file can render a system unbootable. Do not shut down a system that has errors in the /etc/fstab file.

* 1. (Optional) If you are unsure how to correct /etc/fstab errors, you can always restore your backup /etc/fstab file with the following command.

[ec2-user ~]$ **sudo mv /etc/fstab.orig /etc/fstab**

1. Review the file permissions of your new volume mount to make sure that your users and applications can write to the volume. For more information about file permissions, see [File security](http://tldp.org/LDP/intro-linux/html/sect_03_04.html) at The Linux Documentation Project.

# Detaching an Amazon EBS Volume from an Instance

You can detach an Amazon EBS volume from an instance explicitly or by terminating the instance. However, if the instance is running, you must first unmount the volume from the instance.

If an EBS volume is the root device of an instance, you must stop the instance before you can detach the volume.

When a volume with an AWS Marketplace product code is detached from an instance, the product code is no longer associated with the instance.

**Important**

After you detach a volume, you are still charged for volume storage as long as the storage amount exceeds the limit of the AWS Free Tier. You must delete a volume to avoid incurring further charges. For more information, see [Deleting an Amazon EBS Volume](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-deleting-volume.html).

This example unmounts the volume and then explicitly detaches it from the instance. This is useful when you want to terminate an instance or attach a volume to a different instance. To verify that the volume is no longer attached to the instance, see [Viewing Volume Information](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-describing-volumes.html).

You can reattach a volume that you detached (without unmounting it), but it might not get the same mount point. If there were writes to the volume in progress when it was detached, the data on the volume might be out of sync.

**To detach an EBS volume using the console**

1. Use the following command to unmount the /dev/sdh device.

[ec2-user ~]$ **umount -d */dev/sdh***

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the navigation pane, choose **Volumes**.
3. Select a volume and choose **Actions**, **Detach Volume**.
4. In the confirmation dialog box, choose **Yes, Detach**.

**To detach an EBS volume from an instance using the command line**

You can use one of the following commands. For more information about these command line interfaces, see [Accessing Amazon EC2](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html#access-ec2).

* [detach-volume](https://docs.aws.amazon.com/cli/latest/reference/ec2/detach-volume.html) (AWS CLI)
* [Dismount-EC2Volume](https://docs.aws.amazon.com/powershell/latest/reference/items/Dismount-EC2Volume.html) (AWS Tools for Windows PowerShell)

## Troubleshooting

The following are common problems encountered when detaching volumes, and how to resolve them.

**Note**

To guard against the possibility of data loss, take a snapshot of your volume before attempting to unmount it. Forced detachment of a stuck volume can cause damage to the file system or the data it contains or an inability to attach a new volume using the same device name, unless you reboot the instance.

* If you encounter problems while detaching a volume through the Amazon EC2 console, it may be helpful to use the **describe-volumes** CLI command to diagnose the issue. For more information, see [describe-volumes](https://docs.aws.amazon.com/cli/latest/reference/ec2/describe-volumes.html).
* If your volume stays in the detaching state, you can force the detachment by choosing **Force Detach**. Use this option only as a last resort to detach a volume from a failed instance, or if you are detaching a volume with the intention of deleting it. The instance doesn't get an opportunity to flush file system caches or file system metadata. If you use this option, you must perform the file system check and repair procedures.
* If you've tried to force the volume to detach multiple times over several minutes and it stays in the detachingstate, you can post a request for help to the [Amazon EC2 forum](https://forums.aws.amazon.com/forum.jspa?forumID=30). To help expedite a resolution, include the volume ID and describe the steps that you've already taken.
* When you attempt to detach a volume that is still mounted, the volume can become stuck in the busy state while it is trying to detach. The following output from **describe-volumes** shows an example of this condition:
* **aws ec2 describe-volumes --region us-west-2 --volume-ids vol-1234abcd**
* {
* "Volumes": [
* {
* "AvailabilityZone": "us-west-2b",
* "Attachments": [
* {
* "AttachTime": "2016-07-21T23:44:52.000Z",
* "InstanceId": "i-fedc9876",
* "VolumeId": "vol-1234abcd",
* "State": "busy",
* "DeleteOnTermination": false,
* "Device": "/dev/sdf"
* }

....

When you encounter this state, detachment can be delayed indefinitely until you unmount the volume, force detachment, reboot the instance, or all three.

# Deleting an Amazon EBS Volume

After you no longer need an Amazon EBS volume, you can delete it. After deletion, its data is gone and the volume can't be attached to any instance. However, before deletion, you can store a snapshot of the volume, which you can use to re-create the volume later.

To delete a volume, it must be in the available state (not attached to an instance). For more information, see [Detaching an Amazon EBS Volume from an Instance](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-detaching-volume.html).

**To delete an EBS volume using the console**

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the navigation pane, choose **Volumes**.
3. Select a volume and choose **Actions**, **Delete Volume**.
4. In the confirmation dialog box, choose **Yes, Delete**.

**To delete an EBS volume using the command line**

You can use one of the following commands. For more information about these command line interfaces, see [Accessing Amazon EC2](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html#access-ec2).

* [delete-volume](https://docs.aws.amazon.com/cli/latest/reference/ec2/delete-volume.html) (AWS CLI)
* [Remove-EC2Volume](https://docs.aws.amazon.com/powershell/latest/reference/items/Remove-EC2Volume.html) (AWS Tools for Windows PowerShell)

# Modifying the Size, IOPS, or Type of an EBS Volume on Linux

If your current-generation Amazon EBS volume is attached to a current-generation EC2 instance type, you can increase its size, change its volume type, or (for an io1 volume) adjust its IOPS performance, all without detaching it. You can apply these changes to detached volumes as well. For more information, see [Current Generation Instances](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-types.html#current-gen-instances).

The following previous-generation instance types support modification of EBS volumes without detachment: C1, C3, CC2, CR1, G2, I2, M1, M3, and R3. You can safely ignore any warnings that occur with these instance types. If you are using an unsupported previous-generation instance type, or if you encounter an error while attempting a volume modification, follow the procedures in [Appendix: Starting and Stopping an Instance to Modify an EBS Volume](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/stop-start.html).

In general, use the following steps when modifying a volume:

1. **Issue the modification command.** For more information, see [Modifying an EBS Volume from the Console](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/console-modify.html) and [Modifying an EBS Volume from the Command Line](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/cli-modify.html).
2. **Monitor the progress of the modification.** For more information, see [Monitoring the Progress of Volume Modifications](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring_mods.html).
3. **If the size of the volume was modified, extend the volume's file system to take advantage of the increased storage capacity.** For more information, see [Extending a Linux File System after Resizing the Volume](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/recognize-expanded-volume-linux.html) .

Additionally, you can use [Amazon CloudWatch Events](https://docs.aws.amazon.com/AmazonCloudWatch/latest/events/) and [AWS CloudFormation](https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/) to automate the actions associated with volume modification.

There is no charge to modify the configuration of a volume. You are charged at the new volume configuration price after a modification starts. For more information, see the Amazon Elastic Block Store section on the [Amazon EBS Pricing](https://aws.amazon.com/ebs/pricing) page.

**Important**

Before modifying a volume that contains valuable data, it is a best practice to create a snapshot of the volume in case you need to roll back your changes. For information about EBS snapshots, see [Creating an Amazon EBS Snapshot](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html).