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.

For more information about cloud computing, see What is Cloud Computing?

Features of Amazon EC2

Amazon EC2 provides the following features:

- 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 IP 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)

For more information about the features of Amazon EC2, see the Amazon EC2 product page.

For more information about running your website on AWS, see Websites & Website Hosting.

How to Get Started with Amazon EC2

The first thing you need to do is 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 a feature of Amazon EC2, you can read the technical documentation.

Get Up and Running

- Setting Up with Amazon EC2 (p. 20)
- Getting Started with Amazon EC2 Linux Instances (p. 26)

Basics

- Instances and AMIs (p. 4)
- Regions and Availability Zones (p. 7)
- Instance Types (p. 103)
- Tags (p. 610)

Networking and Security

- Amazon EC2 Key Pairs (p. 413)
- Security Groups (p. 421)
- Elastic IP Addresses (EIP) (p. 489)
- Amazon EC2 and Amazon VPC (p. 468)

Storage

- Amazon EBS (p. 519)
- Instance Store (p. 581)

Working with Linux Instances

- Tutorial: Installing a LAMP Web Server (p. 38)
- Getting Started with AWS Web Application Hosting for Linux

If you have questions about whether AWS is right for you, contact AWS Sales. If you have technical questions about Amazon EC2, use the Amazon EC2 forum.

Related Services

You can provision Amazon EC2 resources, such as instances and volumes, directly using Amazon EC2. You can also provision Amazon EC2 resources using other services in AWS. For more information, see the following documentation:

- Auto Scaling Developer Guide
- AWS CloudFormation User Guide
- AWS Elastic Beanstalk Developer Guide
- AWS OpsWorks User Guide

To automatically distribute incoming application traffic across multiple instances, use Elastic Load Balancing. For more information, see Elastic Load Balancing Developer Guide.

To monitor basic statistics for your instances and Amazon EBS volumes, use Amazon CloudWatch. For more information, see the Amazon CloudWatch Developer Guide.

To monitor the calls made to the Amazon EC2 API for your account, including calls made by the AWS Management Console, command line tools, and other services, use AWS CloudTrail. For more information, see the AWS CloudTrail User Guide.

To get a managed relational database in the cloud, use Amazon Relational Database Service (Amazon RDS) to launch a database instance. Although you can set up a database on an EC2 instance, Amazon RDS offers the advantage of handling your database management tasks, such as patching the software, backing up, and storing the backups. For more information, see Amazon Relational Database Service Developer Guide.

Accessing Amazon EC2

Amazon EC2 provides a web-based user interface, the Amazon EC2 console. If you've signed up for an AWS account, you can access the Amazon EC2 console by signing into the AWS Management Console and selecting **EC2** from the console home page.

If you prefer to use a command line interface, you have several options:

AWS Command Line Interface (CLI)
: Provides commands for a broad set of AWS products, and is supported on Windows, Mac, and Linux. To get started, see AWS Command Line Interface User Guide. For more information about the commands for Amazon EC2, see ec2 in the *AWS Command Line Interface Reference*.

Amazon EC2 Command Line Interface (CLI) Tools
: Provides commands for Amazon EC2, Amazon EBS, and Amazon VPC, and is supported on Windows, Mac, and Linux. To get started, see Setting Up the Amazon EC2 Command Line Interface Tools on Linux and Commands (CLI Tools) in the *Amazon EC2 Command Line Reference*.

AWS Tools for Windows PowerShell
: Provides commands for a broad set of AWS products for those who script in the PowerShell environment. To get started, see the AWS Tools for Windows PowerShell User Guide. For more information about the cmdlets for Amazon EC2, see the AWS Tools for Windows PowerShell Reference.

Amazon EC2 provides a Query API. These requests are HTTP or HTTPS requests that use the HTTP verbs GET or POST and a Query parameter named `Action`. For more information about the API actions for Amazon EC2, see Actions in the *Amazon EC2 API Reference*.

If you prefer to build applications using language-specific APIs instead of submitting a request over HTTP or HTTPS, AWS provides libraries, sample code, tutorials, and other resources for software developers. These libraries provide basic functions that automate tasks such as cryptographically signing your requests, retrying requests, and handling error responses, making it is easier for you to get started. For more information, see AWS SDKs and Tools.

Pricing for Amazon EC2

When you sign up for AWS, you can get started with Amazon EC2 for free using the AWS Free Tier.

Amazon EC2 provides the following purchasing options for instances:

On-Demand Instances
: Pay for the instances that you use by the hour, with no long-term commitments or up-front 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
: Specify the maximum hourly price that you are willing to pay to run a particular instance type. The Spot Price fluctuates based on supply and demand, but you never pay more than the maximum price you specified. If the Spot Price moves higher than your maximum price, Amazon EC2 shuts down your Spot Instances.

For a complete list of charges and specific prices for Amazon EC2, see Amazon EC2 Pricing.

To calculate the cost of a sample provisioned environment, see AWS Economics Center.

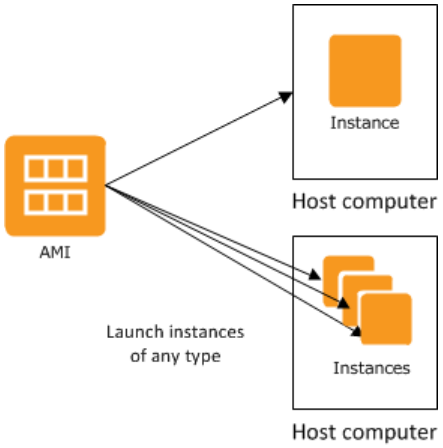
To see your bill, go to your AWS Account Activity page. Your bill contains links to usage reports that provide details about your bill. To learn more about AWS account billing, see AWS Account Billing.

If you have questions concerning AWS billing, accounts, and events, contact AWS Support.

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.

Instances and AMIs

An *Amazon Machine Image (AMI)* is a template that contains a software configuration (for example, an operating system, an application server, and applications). From an AMI, you launch an *instance*, which is a copy of the AMI running as a virtual server in the cloud. You can launch multiple instances of an AMI, as shown in the following figure.



Your instances keep running until you stop or terminate them, or until they fail. If an instance fails, you can launch a new one from the AMI.

Instances

You can launch different types of instances from a single AMI. An *instance type* essentially determines the hardware of the host computer used for your instance. Each instance type offers different compute and memory capabilities. Select an instance type based on the amount of memory and computing power that you need for the application or software that you plan to run on the instance. For more information about the hardware specifications for each Amazon EC2 instance type, see Instance Type Details.

After you launch an instance, it looks like a traditional host, and you can interact with it as you would any computer. You have complete control of your instances; you can use **sudo** to run commands that require root privileges.

Your AWS account has a limit on the number of instances that you can have running. For more information about this limit, and how to request an increase, see How many instances can I run in Amazon EC2 in the Amazon EC2 General FAQ.

In addition to the limit on running instances, there is a limit on the overall number of instances that you can have (whether running, stopped, or in any other state except for terminated). This overall instance limit is two times your running instance limit.

Storage for Your Instance

The root device for your instance contains the image used to boot the instance. For more information, see Amazon EC2 Root Device Volume (p. 14).

Your instance may include local storage volumes, known as instance store volumes, which you can configure at launch time with block device mapping. For more information, see Block Device Mapping (p. 591). After these volumes have been added to and mapped on your instance, they are available for you to mount and use. If your instance fails, or if your instance is stopped or terminated, the data on these volumes is lost; therefore, these volumes are best used for temporary data. For important data, you should use a replication strategy across multiple instances in order to keep your data safe, or store your persistent data in Amazon S3 or Amazon EBS volumes. For more information, see Storage (p. 518).

Security Best Practices

- Use AWS Identity and Access Management (IAM) to control access to your AWS resources, including your instances. You can create IAM users and groups under your AWS account, assign security credentials to each, and control the access that each has to resources and services in AWS. For more information, see Controlling Access to Amazon EC2 Resources (p. 429).
- Restrict access by only allowing trusted hosts or networks to access ports on your instance. For example, you can restrict SSH access by restricting incoming traffic on port 22. For more information, see Amazon EC2 Security Groups (p. 421).
- Review the rules in your security groups regularly, and ensure that you apply the principle of *least privilege*—only open up permissions that you require. You can also create different security groups to deal with instances that have different security requirements. Consider creating a bastion security group that allows external logins, and keep the remainder of your instances in a group that does not allow external logins.
- Disable password-based logins for instances launched from your AMI. Passwords can be found or cracked, and are a security risk. For more information, see Disable Password-Based Logins for Root (p. 67). For more information about sharing AMIs safely, see Shared AMIs (p. 60).

Stopping, Starting, and Terminating Instances

Stopping an instance

When an instance is stopped, the instance performs a normal shutdown, and then transitions to a `stopped` state. All of its Amazon EBS volumes remain attached, and you can start the instance again at a later time.

You are not charged for additional instance hours while the instance is in a stopped state. A full instance hour will be charged for every transition from a stopped state to a running state, even if this happens multiple times within a single hour. If the instance type was changed while the instance was stopped, you will be charged the rate for the new instance type after the instance is started. All of the associated Amazon EBS usage of your instance, including root device usage, is billed using typical Amazon EBS prices.

When an instance is in a stopped state, you can attach or detach Amazon EBS volumes. You can also create an AMI from the instance, and you can change the kernel, RAM disk, and instance type.

Terminating an instance

When an instance is terminated, the instance performs a normal shutdown, then the attached Amazon EBS volumes are deleted unless the volume's `deleteOnTermination` attribute is set to `false`. The instance itself is also deleted, and you can't start the instance again at a later time.

To prevent accidental termination, you can disable instance termination. If you do so, ensure that the `disableApiTermination` attribute is set to `true` for the instance. To control the behavior of an instance shutdown, such as `shutdown -h` in Linux or `shutdown` in Windows, set the `instanceInitiatedShutdownBehavior` instance attribute to `stop` or `terminate` as desired. Instances with Amazon EBS volumes for the root device default to `stop`, and instances with instance-store root devices are always terminated as the result of an instance shutdown.

For more information, see Instance Lifecycle (p. 286).

AMIs

Amazon Web Services (AWS) publishes many Amazon Machine Images (AMIs) that contain common software configurations for public use. In addition, members of the AWS developer community have published their own custom AMIs. You can also create your own custom AMI or AMIs; doing so enables you to quickly and easily start new instances that have everything you need. For example, if your application is a website or a web service, your AMI could include a web server, the associated static content, and the code for the dynamic pages. As a result, after you launch an instance from this AMI, your web server starts, and your application is ready to accept requests.

All AMIs are categorized as either *backed by Amazon EBS*, which means that the root device for an instance launched from the AMI is an Amazon EBS volume, or *backed by instance store*, which means that the root device for an instance launched from the AMI is an instance store volume created from a template stored in Amazon S3.

The description of an AMI indicates the type of root device (either `ebs` or `instance store`). This is important because there are significant differences in what you can do with each type of AMI. For more information about these differences, see Storage for the Root Device (p. 55).

Regions and Availability Zones

Amazon EC2 is hosted in multiple locations world-wide. These locations are composed of regions and Availability Zones. Each *region* is a separate geographic area. Each region has multiple, isolated locations known as *Availability Zones*. Amazon EC2 provides you the ability to place resources, such as instances, and data in multiple locations. Resources aren't replicated across regions unless you do so specifically.

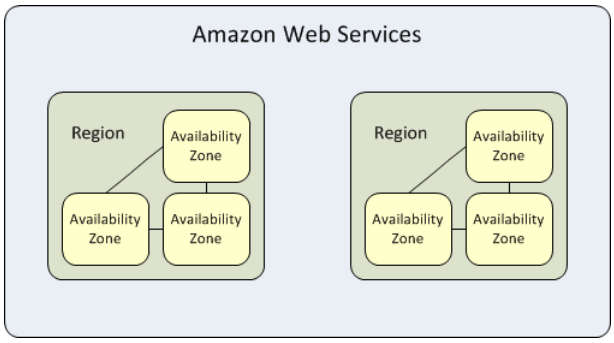
Amazon operates state-of-the-art, highly-available data centers. Although rare, failures can occur that affect the availability of instances that are in the same location. If you host all your instances in a single location that is affected by such a failure, none of your instances would be available.

> **Note**
> Some AWS resources might not be available in all regions and Availability Zones. Ensure that you can create the resources you need in the desired regions or Availability Zone before deploying your applications.

Topics

Region and Availability Zone Concepts

Each region is completely independent. Each Availability Zone is isolated, but the Availability Zones in a region are connected through low-latency links. The following diagram illustrates the relationship between regions and Availability Zones.



Amazon EC2 resources are either global, tied to a region, or tied to an Availability Zone. For more information, see Resource Locations (p. 605).

Regions

Each Amazon EC2 region is designed to be completely isolated from the other Amazon EC2 regions. This achieves the greatest possible fault tolerance and stability.

Amazon EC2 provides multiple regions so that you can launch Amazon EC2 instances in locations that meet your requirements. For example, you might want to launch instances in Europe to be closer to your European customers or to meet legal requirements. The following table lists the regions that provide support for Amazon EC2.

Code	Name
ap-northeast-1	Asia Pacific (Tokyo)
ap-southeast-1	Asia Pacific (Singapore)
ap-southeast-2	Asia Pacific (Sydney)
eu-central-1	EU (Frankfurt)
eu-west-1	EU (Ireland)
sa-east-1	South America (Sao Paulo)
us-east-1	US East (N. Virginia)
us-west-1	US West (N. California)
us-west-2	US West (Oregon)

When you view your resources, you'll only see the resources tied to the region you've specified. This is because regions are isolated from each other, and we don't replicate resources across regions automatically.

When you work with an instance using the command line interface or API actions, you must specify its regional endpoint. For more information about the regions and endpoints for Amazon EC2, see Regions and Endpoints in the *Amazon Web Services General Reference*. For more information about endpoints and protocols in AWS GovCloud (US), see AWS GovCloud (US) Endpoints in the *AWS GovCloud (US) User Guide*.

When you launch an instance, you must select an AMI that's in the same region. If the AMI is in another region, you can copy the AMI to the region you're using. For more information, see Copying an AMI (p. 85).

All communications between regions is across the public Internet. Therefore, you should use the appropriate encryption methods to protect your data. Data transfer between regions is charged at the Internet data transfer rate for both the sending and the receiving instance. For more information, see Amazon EC2 Pricing - Data Transfer.

Availability Zones

You can list the Availability Zones that are available to your account. For more information, see Describing Your Regions and Availability Zones (p. 9).

When you launch an instance, you can select an Availability Zone or let us choose one for you. If you distribute your instances across multiple Availability Zones and one instance fails, you can design your application so that an instance in another Availability Zone can handle requests.

You can also use Elastic IP addresses to mask the failure of an instance in one Availability Zone by rapidly remapping the address to an instance in another Availability Zone. For more information, see Elastic IP Addresses (EIP) (p. 489).

To ensure that resources are distributed across the Availability Zones for a region, we independently map Availability Zones to identifiers for each account. For example, your Availability Zone `us-east-1a` might not be the same location as `us-east-1a` for another account. Note that there's no way for you to coordinate Availability Zones between accounts.

As Availability Zones grow over time, our ability to expand them can become constrained. If this happens, we might restrict you from launching an instance in a constrained Availability Zone unless you already have an instance in that Availability Zone. Eventually, we might also remove the constrained Availability

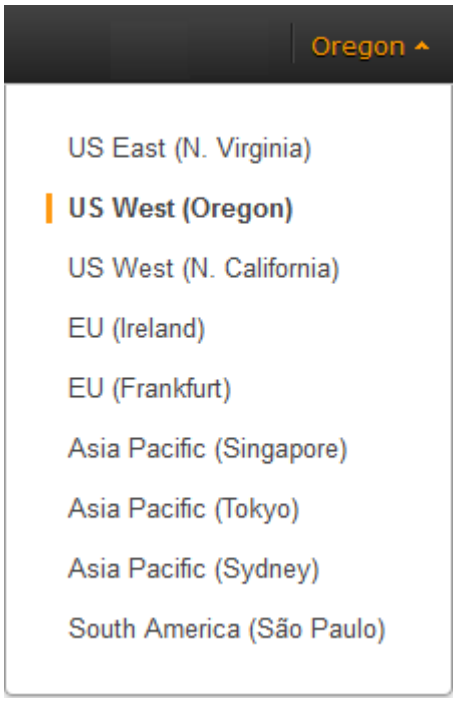
Zone from the list of Availability Zones for new customers. Therefore, your account might have a different number of available Availability Zones in a region than another account.

Describing Your Regions and Availability Zones

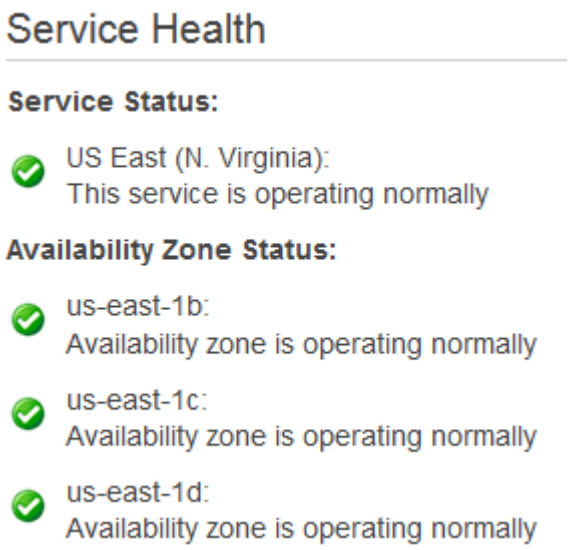
You can use the AWS Management Console or the command line interface to determine which regions and Availability Zones are available for your use. For more information about these command line interfaces, see Accessing Amazon EC2 (p. 3).

To find your regions and Availability Zones using the AWS Management Console

1. Open the AWS Management Console.
2. From the navigation bar, view the options in the region selector.



3. Open the Amazon EC2 console. Your Availability Zones are listed on the dashboard under **Service Health**, under **Availability Zone Status**.



To find your regions and Availability Zones using the AWS CLI

1. Use the describe-regions command as follows to describe your regions.

```
$ aws ec2 describe-regions
{
    "Regions": [
        {
            "Endpoint": "ec2.us-east-1.amazonaws.com",
            "RegionName": "us-east-1"
        },
        {
            "Endpoint": "ec2.ap-southeast-1.amazonaws.com",
            "RegionName": "ap-southeast-1"
        },
        {
            "Endpoint": "ec2.ap-southeast-2.amazonaws.com",
            "RegionName": "ap-southeast-2"
        },
        ...
    ]
}
```

2. Use the describe-availability-zones command as follows to describe your Availability Zones within the `us-east-1` region.

```
$ aws ec2 describe-availability-zones --region us-east-1
{
    "AvailabilityZones": [
        {
            "State": "available",
            "RegionName": "us-east-1",
            "Messages": [],
            "ZoneName": "us-east-1b"
        },
```

```
        {
            "State": "available",
            "RegionName": "us-east-1",
            "Messages": [],
            "ZoneName": "us-east-1c"
        },
        {
            "State": "available",
            "RegionName": "us-east-1",
            "Messages": [],
            "ZoneName": "us-east-1d"
        }
    ]
}
```

To find your regions and Availability Zones using the Amazon EC2 CLI

1. Use the ec2-describe-regions command as follows to describe your regions.

```
PROMPT> ec2-describe-regions
REGION us-east-1  ec2.us-east-1.amazonaws.com
REGION  ap-northeast-1  ec2.ap-northeast-1.amazonaws.com
REGION  ap-southeast-1  ec2.ap-southeast-1.amazonaws.com
..
```

2. Use the ec2-describe-availability-zones command as follows to describe your Availability Zones within the `us-east-1` region.

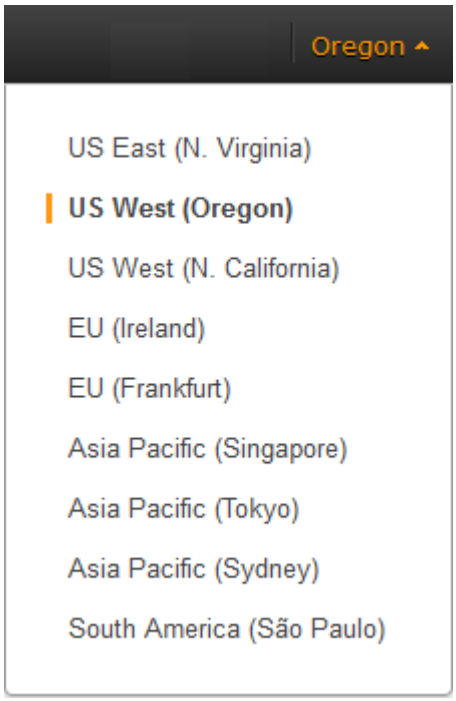
```
PROMPT>  ec2-describe-availability-zones --region us-east-1
AVAILABILITYZONE us-east-1a available us-east-1
AVAILABILITYZONE us-east-1b available us-east-1
AVAILABILITYZONE us-east-1c available us-east-1
AVAILABILITYZONE us-east-1d available us-east-1
```

Specifying the Region for a Resource

Every time you create an Amazon EC2 resource, you can specify the region for the resource. You can specify the region for a resource using the AWS Management Console or the command line.

To specify the region for a resource using the console

1. Open the Amazon EC2 console.
2. Use the region selector in the navigation bar.



To specify the default region using the command line

You can set the value of an environment variable to the desired regional endpoint (for example, `https://ec2.us-west-1.amazonaws.com`):

- `AWS_DEFAULT_REGION` (AWS CLI)
- `EC2_URL` (Amazon EC2 CLI)

Alternatively, you can use the `--region` command line option with each individual command. For example, `--region us-west-1`.

For more information about the endpoints for Amazon EC2, see Amazon Elastic Compute Cloud Endpoints.

Launching Instances in an Availability Zone

When you launch an instance, select a region that puts your instances closer to specific customers, or meets the legal or other requirements you have. By launching your instances in separate Availability Zones, you can protect your applications from the failure of a single location.

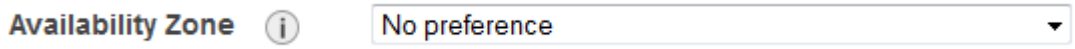
When you launch an instance, you can optionally specify an Availability Zone in the region that you are using. If you do not specify an Availability Zone, we select one for you. When you launch your initial instances, we recommend that you accept the default Availability Zone, because this enables us to select the best Availability Zone for you based on system health and available capacity. If you launch additional instances, only specify an Availability Zone if your new instances must be close to, or separated from, your running instances.

To specify an Availability Zone for your instance using the console

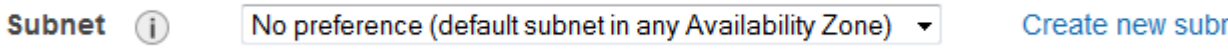
1. Open the Amazon EC2 console.
2. On the dashboard, click **Launch Instance**.

3. Follow the directions for the wizard. On the **Configure Instance Details** page, do the following:

   - [EC2-Classic] Select one of the Availability Zone options from the list, or select **No Preference** to enable us to select the best one for you.

   - [EC2-VPC] Select one of the subnet options from the list, or select **No preference (default subnet in any Availability Zone)** to enable us to select the best one for you.

To specify an Availability Zone for your instance using the AWS CLI

You can use the run-instances command with one of the following options:

- [EC2-Classic] `--placement`
- [EC2-VPC] `--subnet-id`

To specify an Availability Zone for your instance using the Amazon EC2 CLI

You can use the ec2-run-instances command with one of the following options:

- [EC2-Classic] `--availability-zone`
- [EC2-VPC] `--subnet`

Migrating an Instance to Another Availability Zone

If you need to, you can migrate an instance from one Availability Zone to another. For example, if you are trying to modify the instance type of your instance and we can't launch an instance of the new instance type in the current Availability Zone, you could migrate the instance to an Availability Zone where we can launch an instance of that instance type.

The migration process involves creating an AMI from the original instance, launching an instance in the new Availability Zone, and updating the configuration of the new instance, as shown in the following procedure.

To migrate an instance to another Availability Zone

1. Create an AMI from the instance. The procedure depends on the operating system and the type of root device volume for the instance. For more information, see the documentation that corresponds to your operating system and root device volume:

   - Creating an Amazon EBS-Backed Linux AMI (p. 74)
   - Creating an Instance Store-Backed Linux AMI (p. 77)
   - Creating an Amazon EBS-Backed Windows AMI
   - Creating an Instance Store-Backed Windows AMI

2. [EC2-VPC] If you need to preserve the private IP address of the instance, you must delete the subnet in the current Availability Zone and then create a subnet in the new Availability Zone with the same IP address range as the original subnet. Note that you must terminate all instances in a subnet before you can delete it. Therefore, you should move all instances in the current subnet to the new subnet.

3. Launch an instance from the AMI that you just created, specifying the new Availability Zone or subnet. You can use the same instance type as the original instance, or select a new instance type. For more information, see Launching Instances in an Availability Zone (p. 12).
4. If the original instance has an associated Elastic IP address, associate it with the new instance. For more information, see Associating an Elastic IP Address with a Different Running Instance (p. 492).
5. If the original instance is a Reserved Instance, change the Availability Zone for your reservation. (If you also changed the instance type, you can also change the instance type for your reservation.) For more information, see Submitting Modification Requests (p. 224).
6. (Optional) Terminate the original instance. For more information, see Terminating an Instance (p. 315).

Amazon EC2 Root Device Volume

When you launch an instance, the *root device volume* contains the image used to boot the instance. When we introduced Amazon EC2, all AMIs were backed by Amazon EC2 instance store, which means the root device for an instance launched from the AMI is an instance store volume created from a template stored in Amazon S3. After we introduced Amazon EBS, we introduced AMIs that are backed by Amazon EBS. This means that the root device for an instance launched from the AMI is an Amazon EBS volume created from an Amazon EBS snapshot. You can choose between AMIs backed by Amazon EC2 instance store and AMIs backed by Amazon EBS. We recommend that you use AMIs backed by Amazon EBS, because they launch faster and use persistent storage.

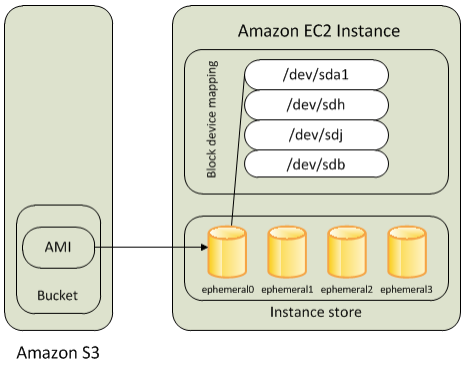
Topics

Root Device Storage Concepts

You can launch an instance from one of two types of AMIs: an instance store-backed AMI or an Amazon EBS-backed AMI. The description of an AMI includes which type of AMI it is; you'll see the root device referred to in some places as either `ebs` (for Amazon EBS-backed) or `instance store` (for instance store-backed). This is important because there are significant differences between what you can do with each type of AMI. For more information about these differences, see Storage for the Root Device (p. 55).

Instance Store-backed Instances

Instances that use instance stores for the root device automatically have instance store volumes available, with one serving as the root device volume. When an instance is launched, the image that is used to boot the instance is copied to the root volume (typically sda1). Any data on the instance store volumes persists as long as the instance is running, but this data is deleted when the instance is terminated (instance store-backed instances do not support the **Stop** action) or if it fails (such as if an underlying drive has issues).

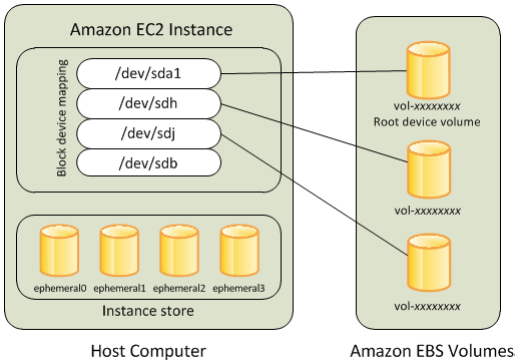


After an instance store-backed instance fails or terminates, it cannot be restored. If you plan to use Amazon EC2 instance store-backed instances, we highly recommend that you distribute the data on your instance stores across multiple Availability Zones. You should also back up the data on your instance store volumes to persistent storage on a regular basis.

For more information, see Amazon EC2 Instance Store (p. 581).

Amazon EBS-backed Instances

Instances that use Amazon EBS for the root device automatically have an Amazon EBS volume attached. When you launch an Amazon EBS-backed instance, we create an Amazon EBS volume for each Amazon EBS snapshot referenced by the AMI you use. You can optionally use other Amazon EBS volumes or instance store volumes.



An Amazon EBS-backed instance can be stopped and later restarted without affecting data stored in the attached volumes. There are various instance– and volume-related tasks you can do when an Amazon EBS-backed instance is in a stopped state. For example, you can modify the properties of the instance, you can change the size of your instance or update the kernel it is using, or you can attach your root volume to a different running instance for debugging or any other purpose.

By default, the root device volume and the other Amazon EBS volumes attached when you launch an Amazon EBS-backed instance are automatically deleted when the instance terminates. For information about how to change this behavior when you launch an instance, see Changing the Root Device Volume to Persist (p. 17).

By default, any Amazon EBS volumes that you attach to a running instance are detached with their data intact when the instance terminates. You can attach a detached volume to any running instance.

If an Amazon EBS-backed instance fails, you can restore your session by following one of these methods:

- Stop and then start again.

- Automatically snapshot all relevant volumes and create a new AMI. For more information, see Creating an Amazon EBS-Backed Linux AMI (p. 74).
- Attach the volume to the new instance by following these steps:
  1. Create a snapshot of the root volume.
  2. Register a new AMI using the snapshot.
  3. Launch a new instance from the new AMI.
  4. Detach the remaining Amazon EBS volumes from the old instance.
  5. Reattach the Amazon EBS volumes to the new instance.

We recommend using either the first or the second method for failed instances with normal volume size, and the third method for failed instances with large volumes.

Choosing an AMI by Root Device Type

The AMI that you specify when you launch your instance determines the type of root device volume that your instance has.

To choose an Amazon EBS-backed AMI using the console

1. Open the Amazon EC2 console.
2. In the navigation pane, click **AMIs**.
3. From the filter lists, select the image type (such as **Public images**), the operating system (such as **Amazon Linux**), and **EBS images**.
4. (Optional) To get additional information to help you make your choice, click the **Show/Hide Columns** icon, update the columns to display, and click **Close**.
5. Choose an AMI and write down its AMI ID.

To choose an instance store-backed AMI using the console

1. Open the Amazon EC2 console.
2. In the navigation pane, click **AMIs**.
3. From the filter lists, select the image type (such as **Public images**), the operating system (such as **Amazon Linux**), and **Instance store images**.
4. (Optional) To get additional information to help you make your choice, click the **Show/Hide Columns** icon, update the columns to display, and click **Close**.
5. Choose an AMI and write down its AMI ID.

To verify the type of the root device volume of an AMI using the command line

You can use one of the following commands. For more information about these command line interfaces, see Accessing Amazon EC2 (p. 3).

- describe-images (AWS CLI)
- ec2-describe-images (Amazon EC2 CLI)
- Get-EC2Image (AWS Tools for Windows PowerShell)

Determining the Root Device Type of Your Instance

To determine the root device type of an instance using the console

1. Open the Amazon EC2 console.
2. In the navigation pane, click **Instances**, and select the instance.
3. Check the value of **Root device type** in the **Description** tab as follows:
   - If the value is `ebs`, this is an Amazon EBS-backed instance.
   - If the value is `instance store`, this is an instance store-backed instance.

To determine the root device type of 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 (p. 3).

- describe-instances (AWS CLI)
- ec2-describe-instances (Amazon EC2 CLI)
- Get-EC2Instance (AWS Tools for Windows PowerShell)

Changing the Root Device Volume to Persist

By default, the root device volume for an AMI backed by Amazon EBS is deleted when the instance terminates. To change the default behavior, set the `DeleteOnTermination` attribute to `false` using a block device mapping.

Changing the Root Volume to Persist Using the Console

Using the console, you can change the `DeleteOnTermination` attribute when you launch an instance. To change this attribute for a running instance, you must use the command line.

To change the root device volume of an instance to persist at launch using the console

1. Open the Amazon EC2 console.
2. From the Amazon EC2 console dashboard, click **Launch Instance**.
3. On the **Choose an Amazon Machine Image (AMI)** page, choose the AMI to use and click **Select**.
4. Follow the wizard to complete the **Choose an Instance Type** and **Configure Instance Details** pages.
5. On the **Add Storage** page, deselect the **Delete On Termination** check box for the root volume.
6. Complete the remaining wizard pages, and then click **Launch**.

You can verify the setting by viewing details for the root device volume on the instance's details pane. Next to **Block devices**, click the entry for the root device volume. By default, **Delete on termination** is `True`. If you change the default behavior, **Delete on termination** is `False`.

Changing the Root Volume of an Instance to Persist Using the AWS CLI

Using the AWS CLI, you can change the `DeleteOnTermination` attribute when you launch an instance or while the instance is running. The root device is typically `/dev/sda1` (Linux) or `xvda` (Windows).

Example at Launch

Use the run-instances command to preserve the root volume by including a block device mapping that sets its `DeleteOnTermination` attribute for to `false`.

```
$ aws ec2 run-instances --image-id ami-1a2b3c4d --block-device-mappings
'[{"DeviceName":"/dev/sda1","Ebs":{"DeleteOnTermination":false}}]" other para
meters...
```

You can confirm that `DeleteOnTermination` is `false` by using the describe-instances command and looking for the `BlockDeviceMappings` entry for `/dev/sda1` in the command output, as shown here.

```
...
"BlockDeviceMappings": [
    {
        "DeviceName": "/dev/sda1",
        "Ebs": {
            "Status": "attached",
            "DeleteOnTermination": false,
            "VolumeId": "vol-877166c8",
            "AttachTime": "2013-07-19T02:42:39.000Z"
        }
    }
...
```

Example While the Instance is Running

Use the modify-instance-attribute command to preserve the root volume by including a block device mapping that sets its `DeleteOnTermination` attribute to `false`.

```
$ aws ec2 modify-instance-attribute --instance-id i-5203422c --block-device-
mappings  '[{"DeviceName":"/dev/sda1","Ebs":{"DeleteOnTermination":false}}]'
```

Changing the Root Volume of an Instance to Persist Using the Amazon EC2 CLI

Using the Amazon EC2 CLI, you can change the `DeleteOnTermination` attribute when you launch an instance or while the instance is running. The root device is typically `/dev/sda1` (Linux) or `xvda` (Windows).

Example at Launch

Use the ec2-run-instances command to include a block device mapping that sets the `DeleteOnTermination` flag for the root device to `false`. Include the `-v` option to run the command in verbose mode.

```
$ ec2-run-instances ami-1a2b3c4d  -b "/dev/sda1=::false" other parameters...
-v
```

By running the command in verbose mode, you can see the underlying request and response, and confirm that `DeleteOnTermination` is `false`, as shown here.

```
...
<blockDeviceMapping>
   <item>
     <deviceName>/dev/sda1</deviceName>
     <ebs>
       <deleteOnTermination>false</deleteOnTermination>
     </ebs>
   </item>
</blockDeviceMapping>
...
```

Example While the Instance is Running

Use the ec2-modify-instance-attribute command to preserve the root volume by setting its `DeleteOnTermination` attribute to `false`.

```
$ ec2-modify-instance-attribute i-5203422c -b "/dev/sda1=::false"
```

Setting Up with Amazon EC2

If you've already signed up for Amazon Web Services (AWS), you can start using Amazon EC2 immediately. You can open the Amazon EC2 console, click **Launch Instance**, and follow the steps in the launch wizard to launch your first instance.

If you haven't signed up for AWS yet, or if you need assistance launching your first instance, complete the following tasks to get set up to use Amazon EC2:

1. Sign Up for AWS (p. 20)
2. Create an IAM User (p. 21)
3. Create a Key Pair (p. 22)
4. Create a Virtual Private Cloud (VPC) (p. 24)
5. Create a Security Group (p. 24)

Sign Up for AWS

When you sign up for Amazon Web Services (AWS), your AWS account is automatically signed up for all services in AWS, including Amazon EC2. You are charged only for the services that you use.

With Amazon EC2, you pay only for what you use. If you are a new AWS customer, you can get started with Amazon EC2 for free. For more information, see AWS Free Tier.

If you have an AWS account already, skip to the next task. If you don't have an AWS account, use the following procedure to create one.

To create an AWS account

1. Open http://aws.amazon.com, and then click **Sign Up**.
2. Follow the on-screen instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a PIN using the phone keypad.

Note your AWS account number, because you'll need it for the next task.