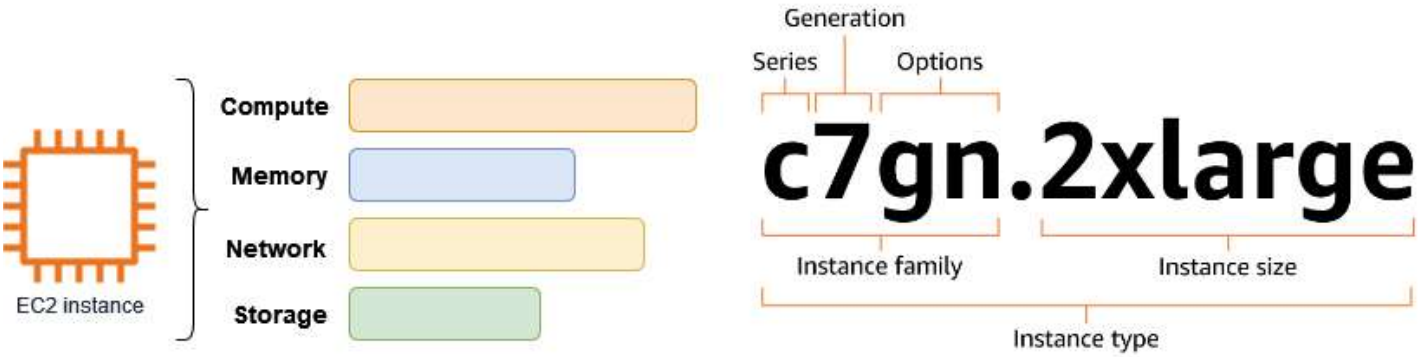
What is Amazon EC2?

Amazon Elastic Compute Cloud (Amazon EC2) provides on-demand, scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 reduces hardware costs 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. You can add capacity (scale up) to handle compute-heavy tasks, such as monthly or yearly processes, or spikes in website traffic. When usage decreases, you can reduce capacity (scale down) again.

An EC2 instance is a virtual server in the AWS Cloud. When you launch an EC2 instance, the instance type that you specify determines the hardware available to your instance. Each instance type offers a different balance of compute, memory, network, and storage resources. For more information, see the Amazon EC2 Instance Types Guide.



Features of Amazon EC2

Amazon EC2 provides the following high-level features:

Instances

Virtual servers.

Amazon Machine Images (AMIs)

Preconfigured templates for your instances that package the components you need for your server (including the operating system and additional software).

Instance types

Various configurations of CPU, memory, storage, networking capacity, and graphics hardware for your instances.

Amazon EBS volumes

Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS).

Instance store volumes

Storage volumes for temporary data that is deleted when you stop, hibernate, or terminate your instance.

Key pairs

Secure login information for your instances. AWS stores the public key and you store the private key in a secure place.

Security groups

A virtual firewall that allows you to specify the protocols, ports, and source IP ranges that can reach your instances, and the destination IP ranges to which your instances can connect.

Amazon EC2 supports the processing, storage, and transmission of credit card data by a merchant or service provider, and has been validated as being compliant with Payment Card Industry (PCI) Data Security Standard (DSS). For more information about PCI DSS, including how to request a copy of the AWS PCI Compliance Package, see PCI DSS Level 1.

Related services

Services to use with Amazon EC2

You can use other AWS services with the instances that you deploy using Amazon EC2.

Amazon EC2 Auto Scaling

Helps ensure you have the correct number of Amazon EC2 instances available to handle the load for your application.

AWS Backup

Automate backing up your Amazon EC2 instances and the Amazon EBS volumes attached to them.

Amazon CloudWatch

Monitor your instances and Amazon EBS volumes.

Elastic Load Balancing

Automatically distribute incoming application traffic across multiple instances.

Amazon GuardDuty

Detect potentially unauthorized or malicious use of your EC2 instances.

EC2 Image Builder

Automate the creation, management, and deployment of customized, secure, and up-to-date server images.

AWS Launch Wizard

Size, configure, and deploy AWS resources for third-party applications without having to manually identify and provision individual AWS resources.

AWS Systems Manager

Perform operations at scale on EC2 instances with this secure end-to-end management solution.

Additional compute services

You can launch instances using another AWS compute service instead of using Amazon EC2.

Amazon Lightsail

Build websites or web applications using Amazon Lightsail, a cloud platform that provides the resources that you need to deploy your project quickly, for a low, predictable monthly price. To compare Amazon EC2 and Lightsail, see Amazon Lightsail or Amazon EC2.

Amazon Elastic Container Service (Amazon ECS)

Deploy, manage, and scale containerized applications on a cluster of EC2 instances. For more information, see Choosing an AWS container service.

Amazon Elastic Kubernetes Service (Amazon EKS)

Run your Kubernetes applications on AWS. For more information, see Choosing an AWS container service.

Access Amazon EC2

You can create and manage your Amazon EC2 instances using the following interfaces:

Amazon EC2 console

A simple web interface to create and manage Amazon EC2 instances and resources. 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.

AWS Command Line Interface

Enables you to interact with AWS services using commands in your command-line shell. It is supported on Windows, Mac, and Linux. For more information about the AWS CLI , see AWS Command Line Interface User Guide. You can find the Amazon EC2 commands in the AWS CLI Command Reference.

AWS CloudFormation

Amazon EC2 supports creating resources using AWS CloudFormation. You create a template, in JSON or YAML format, that describes your AWS resources, and AWS CloudFormation provisions and configures those resources for you. You can reuse your CloudFormation templates to provision the same resources multiple times, whether in the same Region and account or in multiple Regions and accounts. For more information about supported resource types and properties for Amazon EC2, see EC2 resource type reference in the *AWS CloudFormation User Guide*.

AWS SDKs

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 easier for you to get started. For more information, see Tools to Build on AWS.

AWS Tools for PowerShell

A set of PowerShell modules that are built on the functionality exposed by the SDK for .NET. The Tools for PowerShell enable you to script operations on your AWS resources from the PowerShell command line. To get started, see the AWS Tools for Windows PowerShell User Guide. You can find the cmdlets for Amazon EC2, in the AWS Tools for PowerShell Cmdlet Reference.

Query API

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*.

Pricing for Amazon EC2

Amazon EC2 provides the following pricing options:

Free Tier

You can get started with Amazon EC2 for free. To explore the Free Tier options, see AWS Free Tier.

On-Demand Instances

Pay for the instances that you use by the second, with a minimum of 60 seconds, with no long-term commitments or upfront payments.

Savings Plans

You can reduce your Amazon EC2 costs by making a commitment to a consistent amount of usage, in USD per hour, for a term of 1 or 3 years.

Reserved Instances

You can reduce your Amazon EC2 costs by making a commitment to a specific instance configuration, including instance type and Region, for a term of 1 or 3 years.

Spot Instances

Request unused EC2 instances, which can reduce your Amazon EC2 costs significantly.

Dedicated Hosts

Reduce costs by using a physical EC2 server that is fully dedicated for your use, either On-Demand or as part of a Savings Plan. You can use your existing server-bound software licenses and get help meeting compliance requirements.

On-Demand Capacity Reservations

Reserve compute capacity for your EC2 instances in a specific Availability Zone for any duration of time.

Per-second billing

Removes the cost of unused minutes and seconds from your bill.

For a complete list of charges and prices for Amazon EC2 and more information about the purchase models, see Amazon EC2 pricing.

Estimates, billing, and cost optimization

To create estimates for your AWS use cases, use the AWS Pricing Calculator.

To estimate the cost of transforming **Microsoft workloads** to a modern architecture that uses open source and cloud-native services deployed on AWS, use the AWS Modernization Calculator for Microsoft Workloads.

To see your bill, go to the **Billing and Cost Management Dashboard** in the AWS Billing and Cost Management console. Your bill contains links to usage reports that provide details about your bill. To learn more about AWS account billing, see AWS Billing and Cost Management User Guide.

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

To calculate the cost of a sample provisioned environment, see Cloud Economics Center. When calculating the cost of a provisioned environment, remember to include incidental costs such as snapshot storage for EBS volumes.

You can optimize the cost, security, and performance of your AWS environment using AWS Trusted Advisor.

You can use AWS Cost Explorer to analyze the cost and usage of your EC2 instances. You can view data up to the last 13 months, and forecast how much you are likely to spend for the next 12 months. For more information, see Analyzing your costs and usage with AWS Cost Explorer in the *AWS Cost Management User Guide*.

Resources

- Amazon EC2 features
- AWS re:Post
- AWS Skill Builder
- AWS Support

- Hands-on Tutorials
- Web Hosting
- Windows on AWS

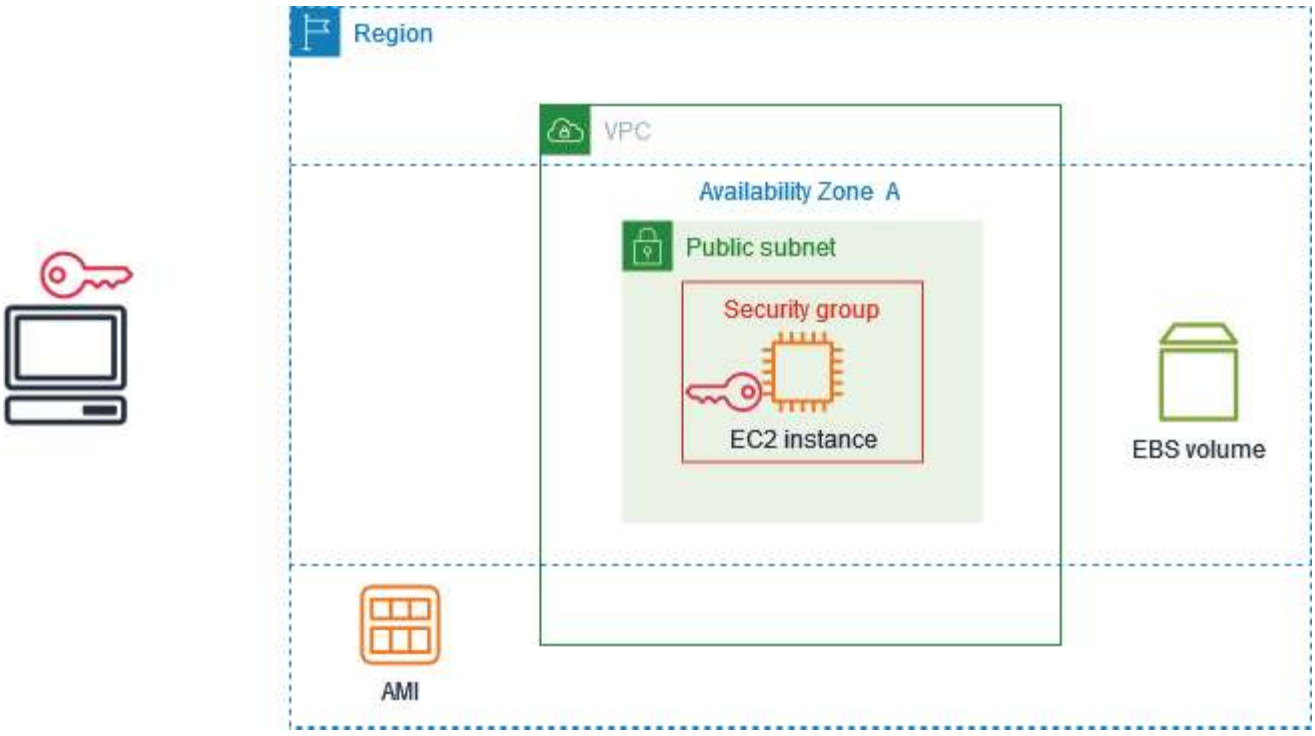
Get started with Amazon EC2

Use this tutorial to get started with Amazon Elastic Compute Cloud (Amazon EC2). You'll learn how to launch and connect to an EC2 instance. An *instance* is a virtual server in the AWS Cloud. With Amazon EC2, you can set up and configure the operating system and applications that run on your instance.

Overview

The following diagram shows the key components that you'll use in this tutorial:

- **An image** – A template that contains the software to run on your instance, such as the operating system.
- **A key pair** – A set of security credentials that you use to prove your identity when connecting to your instance. The public key is on your instance and the private key is on your computer.
- **A network** – A virtual private cloud (VPC) is a virtual network dedicated to your AWS account. To help you get started quickly, your account comes with a default VPC in each AWS Region, and each default VPC has a default subnet in each Availability Zone.
- **A security group** – Acts as a virtual firewall to control inbound and outbound traffic.
- **An EBS volume** – We require a root volume for the image. You can optionally add data volumes.



Cost for this tutorial

When you sign up for AWS, you can get started with Amazon EC2 using the AWS Free Tier. If you created your AWS account less than 12 months ago, and have not already exceeded the Free Tier benefits for Amazon EC2, it won't cost you anything to complete this tutorial, because we help you select options that are within the Free Tier benefits. Otherwise, you'll incur the standard Amazon EC2 usage fees from the time that you launch the instance until you terminate the instance (which is the final task of this tutorial), even if it remains idle.

For instructions to determine whether you are eligible for the Free Tier, see the section called "Track your Free Tier usage".

Tasks

- Step 1: Launch an instance
- Step 2: Connect to your instance
- Step 3: Clean up your instance
- Next steps

Step 1: Launch an instance

You can launch an EC2 instance using the AWS Management Console as described in the following procedure. This tutorial is intended to help you quickly launch your first instance, so it doesn't cover all possible options.

To launch an instance

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation bar at the top of the screen, we display the current AWS Region — for example, **Ohio**. You can use the selected Region, or optionally select a Region that is closer to you.
3. From the EC2 console dashboard, in the **Launch instance** pane, choose **Launch instance**.
4. Under **Name and tags**, for **Name**, enter a descriptive name for your instance.
5. Under **Application and OS Images (Amazon Machine Image)**, do the following:
   a. Choose **Quick Start**, and then choose the operating system (OS) for your instance. For your first Linux instance, we recommend that you choose Amazon Linux.
   b. From **Amazon Machine Image (AMI)**, select an AMI that is marked **Free Tier eligible**.
6. Under **Instance type**, for **Instance type**, choose `t2.micro`, which is eligible for the Free Tier. In Regions where `t2.micro` is not available, `t3.micro` is eligible for the Free Tier.
7. Under **Key pair (login)**, for **Key pair name**, choose an existing key pair or choose **Create new key pair** to create your first key pair.

   > **Warning**
   >
   > If you choose **Proceed without a key pair (Not recommended)**, you won't be able to connect to your instance using the methods described in this tutorial.

8. Under **Network settings**, notice that we selected your default VPC, selected the option to use the default subnet in an Availability Zone that we choose for you, and configured a security group with a rule that allows connections to your instance from anywhere (`0.0.0.0/0`).

   > **Warning**
   >
   > If you specify `0.0.0.0/0`, you are enabling traffic from any IP addresses in the world. For the SSH and RDP protocols, you might consider this acceptable for a short time

in a test environment, but it's unsafe for production environments. In production, be sure to authorize access only from the appropriate individual IP address or range of addresses.

For your first instance, we recommend that you use the default settings. Otherwise, you can update your network settings as follows:

- (Optional) To use a specific default subnet, choose **Edit** and then choose a subnet.
- (Optional) To use a different VPC, choose **Edit** and then choose an existing VPC. If the VPC isn't configured for public internet access, you won't be able to connect to your instance.
- (Optional) To restrict inbound connection traffic to a specific network, choose **Custom** instead of **Anywhere**, and enter the CIDR block for your network.
- (Optional) To use a different security group, choose **Select existing security group** and choose an existing security group. If the security group does not have a rule that allows connection traffic from your network, you won't be able to connect to your instance. For a Linux instance, you must allow SSH traffic. For a Windows instance, you must allow RDP traffic.

9. Under **Configure storage**, notice that we configured a root volume but no data volumes. This is sufficient for test purposes.
10. Review a summary of your instance configuration in the **Summary** panel, and when you're ready, choose **Launch instance**.
11. If the launch is successful, choose the ID of the instance from the **Success** notification to open the **Instances** page and monitor the status of the launch.
12. Select the checkbox for the instance. The initial instance state is `pending`. After the instance starts, its state changes to `running`. Choose the **Status and alarms** tab. After your instance passes its status checks, it is ready to receive connection requests.

Step 2: Connect to your instance

The procedure that you use depends on the operating system of the instance. If you can't connect to your instance, see Troubleshoot issues connecting to your Amazon EC2 Linux instance for assistance.

Linux instances

You can connect to your Linux instance using any SSH client. If you are running Windows on your computer, open a terminal and run the `ssh` command to verify that you have an SSH client installed. If the command is not found, install OpenSSH for Windows.

To connect to your instance using SSH

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Instances**.
3. Select the instance and then choose **Connect**.
4. On the **Connect to instance** page, choose the **SSH client** tab.
5. (Optional) If you created a key pair when you launched the instance and downloaded the private key (.pem file) to a computer running Linux or macOS, run the example **chmod** command to set the permissions for your private key.
6. Copy the example SSH command. The following is an example, where *key-pair-name*.pem is the name of your private key file, *ec2-user* is the username associated with the image, and the string after the @ symbol is the public DNS name of the instance.

   ```
   ssh -i key-pair-name.pem ec2-user@ec2-198-51-100-1.us-east-2.compute.amazonaws.com
   ```

7. In a terminal window on your computer, run the **ssh** command that you saved in the previous step. If the private key file is not in the current directory, you must specify the fully-qualified path to the key file in this command.

   The following is an example response:

   ```
   The authenticity of host 'ec2-198-51-100-1.us-east-2.compute.amazonaws.com
    (198-51-100-1)' can't be established.
   ECDSA key fingerprint is l4UB/neBad9tvkgJf1QZWxheQmR59WgrgzEimCG6kZY.
   Are you sure you want to continue connecting (yes/no)?
   ```

8. (Optional) Verify that the fingerprint in the security alert matches the instance fingerprint contained in the console output when you first start an instance. To get the console output, choose **Actions**, **Monitor and troubleshoot**, **Get system log**. If the fingerprints don't match, someone might be attempting a man-in-the-middle attack. If they match, continue to the next step.
9. Enter **yes**.

The following is an example response:

```
Warning: Permanently added 'ec2-198-51-100-1.us-
east-2.compute.amazonaws.com' (ECDSA) to the list of known hosts.
```

Windows instances

To connect to a Windows instance using RDP, you must retrieve the initial administrator password and then enter this password when you connect to your instance. It takes a few minutes after instance launch before this password is available. Your account must have permission to call the GetPasswordData action. For more information, see Example policies to control access the Amazon EC2 API.

The default username for the Administrator account depends on the language of the operating system (OS) contained in the AMI. To determine the correct username, identify the language of the OS, and then choose the corresponding username. For example, for an English OS, the username is `Administrator`, for a French OS it's `Administrateur`, and for a Portuguese OS it's `Administrador`. If a language version of the OS does not have a username in the same language, choose the username `Administrator (Other)`. For more information, see Localized Names for Administrator Account in Windows in the Microsoft website.

To retrieve the initial administrator password

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Instances**.
3. Select the instance and then choose **Connect**.
4. On the **Connect to instance** page, choose the **RDP client** tab.
5. For **Username**, choose the default username for the Administrator account. The username you choose must match the language of the operating system (OS) contained in the AMI that you used to launch your instance. If there is no username in the same language as your OS, choose **Administrator (Other)**.
6. Choose **Get password**.
7. On the **Get Windows password** page, do the following:

a. Choose **Upload private key file** and navigate to the private key (`.pem`) file that you specified when you launched the instance. Select the file and choose **Open** to copy the entire contents of the file to this window.

b. Choose **Decrypt password**. The **Get Windows password** page closes, and the default administrator password for the instance appears under **Password**, replacing the **Get password** link shown previously.

c. Copy the password and save it in a safe place. This password is required to connect to the instance.

The following procedure uses the Remote Desktop Connection client for Windows (MSTSC). If you're using a different RDP client, download the RDP file and then see the documentation for the RDP client for the steps to establish the RDP connection.

To connect to a Windows instance using an RDP client

1. On the **Connect to instance** page, choose **Download remote desktop file**. When the file download is finished, choose **Cancel** to return to the **Instances** page. The RDP file is downloaded to your `Downloads` folder.
2. Run `mstsc.exe` to open the RDP client.
3. Expand **Show options**, choose **Open**, and select the .rdp file from your `Downloads` folder.
4. By default, **Computer** is the public IPv4 DNS name of the instance and **User name** is the administrator account. To connect to the instance using IPv6 instead, replace the public IPv4 DNS name of the instance with its IPv6 address. Review the default settings and change them as needed.
5. Choose **Connect**. If you receive a warning that the publisher of the remote connection is unknown, choose **Connect** to continue.
6. Enter the password that you saved previously, and then choose **OK**.
7. Due to the nature of self-signed certificates, you might get a warning that the security certificate could not be authenticated. Do one of the following:
   - If you trust the certificate, choose **Yes** to connect to your instance.
   - [Windows] Before you proceed, compare the thumbprint of the certificate with the value in the system log to confirm the identity of the remote computer. Choose **View certificate** and then choose **Thumbprint** from the **Details** tab. Compare this value to the value of `RDPCERTIFICATE-THUMBPRINT` in **Actions**, **Monitor and troubleshoot**, **Get system log**.

- [Mac OS X] Before you proceed, compare the fingerprint of the certificate with the value in the system log to confirm the identity of the remote computer. Choose **Show Certificate**, expand **Details**, and choose **SHA1 Fingerprints**. Compare this value to the value of `RDPCERTIFICATE-THUMBPRINT` in **Actions**, **Monitor and troubleshoot**, **Get system log**.

8. If the RDP connection is successful, the RDP client displays the Windows login screen and then the Windows desktop. If you receive an error message instead, see the section called "Remote Desktop can't connect to the remote computer". When you are finished with the RDP connection, you can close the RDP client.

Step 3: Clean up your instance

After you've finished with the instance that you created for this tutorial, you should clean up by terminating the instance. If you want to do more with this instance before you clean up, see Next steps.

> **Important**
>
> Terminating an instance effectively deletes it; you can't reconnect to an instance after you've terminated it.

You'll stop incurring charges for that instance or usage that counts against your Free Tier limits as soon as the instance status changes to `shutting down` or `terminated`. To keep your instance for later, but not incur charges or usage that counts against your Free Tier limits, you can stop the instance now and then start it again later. For more information, see Stop and start Amazon EC2 instances.

To terminate your instance

1. In the navigation pane, choose **Instances**. In the list of instances, select the instance.
2. Choose **Instance state**, **Terminate (delete) instance**.
3. Choose **Terminate (delete)** when prompted for confirmation.

   Amazon EC2 shuts down and terminates your instance. After your instance is terminated, it remains visible on the console for a short while, and then the entry is automatically deleted. You cannot remove the terminated instance from the console display yourself.

Next steps

After you start your instance, you might want to explore the following next steps:

- Explore the Amazon EC2 core concepts with the introductory tutorials. For more information, see Tutorials for launching EC2 instances.
- Learn how to track your Amazon EC2 Free Tier usage using the console. For more information, see the section called “Track your Free Tier usage”.
- Configure a CloudWatch alarm to notify you if your usage exceeds the Free Tier. For more information, see Tracking your AWS Free Tier usage in the *AWS Billing User Guide*.
- Add an EBS volume. For more information, see Create an Amazon EBS volume in the *Amazon EBS User Guide*.
- Learn how to remotely manage your EC2 instance using the Run command. For more information, see AWS Systems Manager Run Command in the *AWS Systems Manager User Guide*.
- Learn about instance purchasing options. For more information, see Amazon EC2 billing and purchasing options.
- Get advice about instance types. For more information, see Get recommendations from EC2 instance type finder.

Best practices for Amazon EC2

To ensure the maximum benefit from Amazon EC2, we recommend that you perform the following best practices.

Security

- Manage access to AWS resources and APIs using identity federation with an identity provider and IAM roles whenever possible. For more information, see Creating IAM policies in the *IAM User Guide*.
- Implement the least permissive rules for your security group.
- Regularly patch, update, and secure the operating system and applications on your instance. For more information, see Update management. For guidelines specific to Windows operating systems, see Security best practices for Windows instances.
- Use Amazon Inspector to automatically discover and scan Amazon EC2 instances for software vulnerabilities and unintended network exposure. For more information, see the Amazon Inspector User Guide.
- Use AWS Security Hub controls to monitor your Amazon EC2 resources against security best practices and security standards. For more information about using Security Hub, see Amazon Elastic Compute Cloud controls in the *AWS Security Hub User Guide*.

Storage

- Understand the implications of the root device type for data persistence, backup, and recovery. For more information, see Root device type.
- Use separate Amazon EBS volumes for the operating system versus your data. Ensure that the volume with your data persists after instance termination. For more information, see Preserve data when an instance is terminated.
- Use the instance store available for your instance to store temporary data. Remember that the data stored in instance store is deleted when you stop, hibernate, or terminate your instance. If you use instance store for database storage, ensure that you have a cluster with a replication factor that ensures fault tolerance.
- Encrypt EBS volumes and snapshots. For more information, see Amazon EBS encryption in the *Amazon EBS User Guide*.

Resource management

- Use instance metadata and custom resource tags to track and identify your AWS resources. For more information, see Use instance metadata to manage your EC2 instance and Tag your Amazon EC2 resources.
- View your current limits for Amazon EC2. Plan to request any limit increases in advance of the time that you'll need them. For more information, see Amazon EC2 service quotas.
- Use AWS Trusted Advisor to inspect your AWS environment, and then make recommendations when opportunities exist to save money, improve system availability and performance, or help close security gaps. For more information, see AWS Trusted Advisor in the *AWS Support User Guide*.

Backup and recovery

- Regularly back up your EBS volumes using Amazon EBS snapshots, and create an Amazon Machine Image (AMI) from your instance to save the configuration as a template for launching future instances. For more information about AWS services that help achieve this use case, see AWS Backup and Amazon Data Lifecycle Manager.
- Deploy critical components of your application across multiple Availability Zones, and replicate your data appropriately.
- Design your applications to handle dynamic IP addressing when your instance restarts. For more information, see Amazon EC2 instance IP addressing.
- Monitor and respond to events. For more information, see Monitor Amazon EC2 resources.
- Ensure that you are prepared to handle failover. For a basic solution, you can manually attach a network interface or Elastic IP address to a replacement instance. For more information, see Elastic network interfaces. For an automated solution, you can use Amazon EC2 Auto Scaling. For more information, see the Amazon EC2 Auto Scaling User Guide.
- Regularly test the process of recovering your instances and Amazon EBS volumes to ensure data and services are restored successfully.

Networking

- Set the time-to-live (TTL) value for your applications to 255, for IPv4 and IPv6. If you use a smaller value, there is a risk that the TTL will expire while application traffic is in transit, causing reachability issues for your instances.

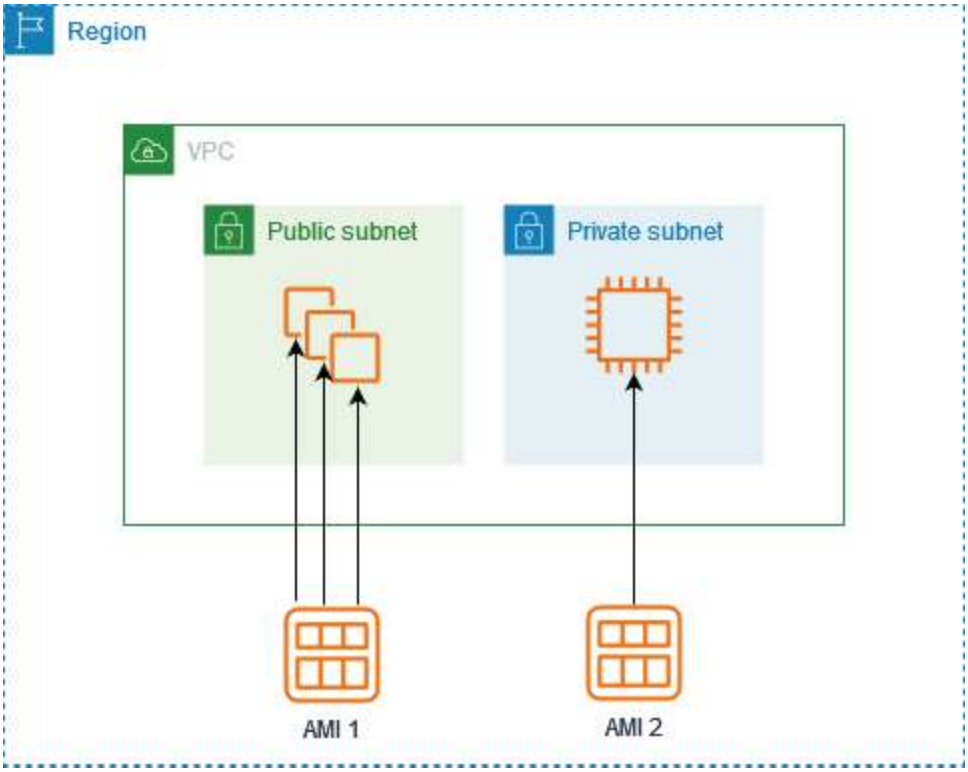
Amazon Machine Images in Amazon EC2

An Amazon Machine Image (AMI) is an image that provides the software that is required to set up and boot an Amazon EC2 instance. Each AMI also contains a block device mapping that specifies the block devices to attach to the instances that you launch. You must specify an AMI when you launch an instance. The AMI must be compatible with the instance type that you chose for your instance. You can use an AMI provided by AWS, a public AMI, an AMI that someone else shared with you, or an AMI that you purchased from the AWS Marketplace.

An AMI is specific to the following:

- Region
- Operating system
- Processor architecture
- Root device type
- Virtualization type

You can launch multiple instances from a single AMI when you require multiple instances with the same configuration. You can use different AMIs to launch instances when you require instances with different configurations, as shown in the following diagram.



You can create an AMI from your Amazon EC2 instances and then use it to launch instances with the same configuration. You can copy an AMI to another AWS Region, and then use it to launch instances in that Region. You can also share an AMI that you created with other accounts so that they can launch instances with the same configuration. You can sell your AMI using the AWS Marketplace.

Contents

AMI types and characteristics in Amazon EC2

When you launch an instance, the AMI that you choose must be compatible with the instance type that you choose. You can select an AMI to use based on the following characteristics:

- Region
- Operating system
- Processor architecture
- Launch permissions
- Root device type
- Virtualization types

Launch permissions

The owner of an AMI determines its availability by specifying launch permissions. Launch permissions fall into the following categories.

Launch permission	Description
public	The owner grants launch permissions to all AWS accounts.
explicit	The owner grants launch permissions to specific AWS accounts, organizations, or organizational units (OUs).
implicit	The owner has implicit launch permissions for an AMI.

Amazon and the Amazon EC2 community provide a large selection of public AMIs. For more information, see Understand shared AMI usage in Amazon EC2. Developers can charge for their AMIs. For more information, see Paid AMIs in the AWS Marketplace for Amazon EC2 instances.

Root device type

All AMIs are categorized as either *backed by Amazon EBS* or *backed by instance store*.

- Amazon EBS-backed AMI – The root device for an instance launched from the AMI is an Amazon Elastic Block Store (Amazon EBS) volume created from an Amazon EBS snapshot. Supported for both Linux and Windows AMIs.
- Amazon instance store-backed AMI – The root device for an instance launched from the AMI is an instance store volume created from a template stored in Amazon S3. Supported for Linux AMIs only. Windows AMIs do not support instance store for the root device.

For more information, see Root volumes for your Amazon EC2 instances.

The following table summarizes the important differences when using the two types of AMIs.

Characteristic	Amazon EBS-backed AMI	Amazon instance store-backed AMI
Root device volume	EBS volume	Instance store volume
Boot time for an instance	Usually less than 1 minute	Usually less than 5 minutes
Data persistence	By default, the root volume is deleted when the instance terminates.* Data on any other EBS volumes persists after instance termination by default.	Data on any instance store volumes persists only during the life of the instance.
Stopped state	Can be in a stopped state. Even when the instance is stopped and not running, the root volume is persisted in Amazon EBS.	Cannot be in a stopped state; instances are running or terminated.
Modifications	The instance type, kernel, RAM disk, and user data can be changed while the instance is stopped.	Instance attributes are fixed for the life of an instance.
Charges		

Characteristic	Amazon EBS-backed AMI	Amazon instance store-backed AMI
	You're charged for instance usage, EBS volume usage, and storing your AMI as an EBS snapshot.	You're charged for instance usage and storing your AMI in Amazon S3.
AMI creation/bundling	Uses a single command/call	Requires installation and use of AMI tools

* By default, EBS root volumes have the `DeleteOnTermination` flag set to `true`. For information about how to change this flag so that the volume persists after termination, see Keep an Amazon EBS root volume after an Amazon EC2 instance terminates.

** Supported with `io2` EBS Block Express only. For more information, see Provisioned IOPS SSD Block Express volumes in the *Amazon EBS User Guide*.

Determine the root device type of your AMI

The AMI that you use to launch an EC2 instance determines the type of the root volume. The root volume of an EC2 instance is either an EBS volume or an instance store volume.

Nitro-based instances support only EBS root volumes. The following previous generation instance types are the only instance types that support instance store root volumes: C1, C3, D2, I2, M1, M2, M3, R3, and X1.

Console

To determine the root device type of an AMI

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **AMIs**, and select the AMI.
3. On the **Details** tab, check the value of **Root device type** as follows:
   - `ebs` – This is an EBS-backed AMI.
   - `instance store` – This is an instance store-backed AMI.

AWS CLI

To determine the root device type of an AMI

Use the describe-images command.

```
aws ec2 describe-images \
    --image-ids ami-0abcdef1234567890 \
    --query Images[].RootDeviceType
```

The following is example output.

```
ebs
```

PowerShell

To determine the root device type of an AMI

Use the Get-EC2Image cmdlet.

```
(Get-EC2Image `
    -ImageId ami-0abcdef1234567890).RootDeviceType.Value
```

The following is example output.

```
ebs
```

Virtualization types

Amazon Machine Images use one of two types of virtualization: paravirtual (PV) or hardware virtual machine (HVM). The main differences between PV and HVM AMIs are the way in which they boot and whether they can take advantage of special hardware extensions (CPU, network, and storage) for better performance. Windows AMIs are HVM AMIs.

The following table compares HVM and PV AMIs.

Characteristic	HVM	PV
Description	HVM AMIs are presented with a fully virtualized set	PV AMIs boot with a special boot loader called PV-

Characteristic	HVM	PV
	of hardware and boot by executing the master boot record of the root block device of your image. This virtualization type provides the ability to run an operating system directly on top of a virtual machine without any modification, as if it were run on the bare-metal hardware. The Amazon EC2 host system emulates some or all of the underlying hardware that is presented to the guest.	GRUB, which starts the boot cycle and then chain loads the kernel specified in the `menu.lst` file on your image. Paravirtual guests can run on host hardware that does not have explicit support for virtualization. For more information about PV-GRUB and its use in Amazon EC2, see User provided kernels.
Supported instance types	All current generation instance types support HVM AMIs.	The following previous generation instance types support PV AMIs: C1, C3, M1, M3, M2, and T1. Current generation instance types do not support PV AMIs.

Characteristic	HVM	PV
Support for hardware extensions	HVM guests can take advantage of hardware extensions that provide fast access to the underlying hardware on the host system. They are required to use enhanced networking and GPU processing. To pass through instructions to specialized network and GPU devices, the OS must have access to the native hardware platform, and HVM virtualization provides this access. For more information, see Enhanced networking on Amazon EC2 instances.	No, they can't take advantage of special hardware extensions such as enhanced networking or GPU processing.
How to find	Verify that the virtualization type of the AMI is set to `hvm`, using the console or the describe-images command.	Verify that the virtualization type of the AMI is set to `paravirtual` , using the console or the describe-images command.

PV on HVM

Paravirtual guests traditionally performed better with storage and network operations than HVM guests because they could leverage special drivers for I/O that avoided the overhead of emulating network and disk hardware, whereas HVM guests had to translate these instructions to emulated hardware. Now PV drivers are available for HVM guests, so operating systems that cannot be ported to run in a paravirtualized environment can still see performance advantages in storage and network I/O by using them. With these PV on HVM drivers, HVM guests can get the same, or better, performance than paravirtual guests.

Find an AMI that meets the requirements for your EC2 instance

An AMI includes the components and applications, such as the operating system and type of root volume, required to launch an instance. To launch an instance, you must find an AMI that meets your needs.

When selecting an AMI, consider the following requirements you might have for the instances that you want to launch:

- The AWS Region of the AMI as AMI IDs are unique to each Region.
- The operating system (for example, Linux or Windows).
- The architecture (for example, 32-bit, 64-bit, or 64-bit ARM).
- The root device type (for example, Amazon EBS or instance store).
- The provider (for example, Amazon Web Services).
- Additional software (for example, SQL Server).

Console

You can select from the list of AMIs when you use the launch instance wizard, or you can search all available AMIs using the **Images** page.

To find a Quick Start AMI using the launch instance wizard

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. From the navigation bar, select the Region in which to launch your instances. You can select any Region that's available to you, regardless of your location. AMI IDs are unique to each AWS Region.
3. From the console dashboard, choose **Launch instance**.
4. Under **Application and OS Images (Amazon Machine Image)**, choose **Quick Start**, choose the operating system (OS) for your instance, and then, from **Amazon Machine Image (AMI)**, select from one of the commonly used AMIs in the list. If you don't see the AMI that you want to use, choose **Browse more AMIs** to browse the full AMI catalog. For more information, see Application and OS Images (Amazon Machine Image).

To find an AMI using the AMIs page

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. From the navigation bar, select the Region in which to launch your instances. You can select any Region that's available to you, regardless of your location. AMI IDs are unique to each AWS Region.
3. In the navigation pane, choose **AMIs**.
4. (Optional) Use the filter and search options to scope the list of displayed AMIs to see only the AMIs that match your criteria.

   For example, to list all AMIs provided by AWS, choose **Public images**. Then use the search options to further scope the list of displayed AMIs. Choose the **Search** bar and, from the menu, choose **Owner alias**, then the **=** operator, and then the value **amazon**. To find AMIs that match a specific platform, for example Linux or Windows, choose the **Search** bar again to choose **Platform**, then the **=** operator, and then the operating system from the list provided.
5. (Optional) Choose the **Preferences** icon to select which image attributes to display, such as the root device type. Alternatively, you can select an AMI from the list and view its properties on the **Details** tab.
6. Before you select an AMI, it's important that you check whether it's backed by instance store or by Amazon EBS and that you are aware of the effects of this difference. For more information, see Root device type.
7. To launch an instance from this AMI, select it and then choose **Launch instance from image**. For more information about launching an instance using the console, see Launch an EC2 instance using the launch instance wizard in the console. If you're not ready to launch the instance now, make note of the AMI ID for later.

AWS CLI

Use the describe-images command to find an AMI that meets your requirements. By default, this command returns all AMIs that are public, that you own, and that are shared with you.

To find an AMI owned by Amazon

Use the describe-images command with the `--owners` option.

```
aws ec2 describe-images --owners amazon
```

To find a Windows AMI

Add the following filter to display only Windows AMIs.

```
--filters "Name=platform,Values=windows"
```

To find an EBS-backed AMI

Add the following filter to display only AMIs backed by Amazon EBS.

```
--filters "Name=root-device-type,Values=ebs"
```

PowerShell

Use the Get-EC2Image cmdlet to find an AMI that meets your requirements. By default, this cmdlet returns all AMIs that are public, that you own, or that are shared with you.

To find an AMI owned by Amazon

Use the Get-EC2Image command with the `-Owner` parameter.

```
Get-EC2Image -Owner amazon
```

To find a Windows AMI

Add the following filter to display only Windows AMIs.

```
-Filter @{Name="platform"; Values="windows"}
```

For additional examples, see Find an Amazon Machine Image Using Windows PowerShell in the *AWS Tools for Windows PowerShell User Guide*.

Related resources

For more information about AMIs for a specific operating system, see the following:

- Amazon Linux 2023 – AL2023 on Amazon EC2 in the *Amazon Linux 2023 User Guide*
- Ubuntu – Amazon EC2 AMI Locator on the *Canonical Ubuntu website*

- RHEL – Red Hat Enterprise Linux Images (AMI) Available on Amazon Web Services (AWS) on the Red Hat website
- Windows Server – AWS Windows AMI reference

For information about AMIs that you can subscribe to on the AWS Marketplace see Paid AMIs in the AWS Marketplace for Amazon EC2 instances.

For information about using Systems Manager to help your users find the latest AMI that they should use when launching an instance, see the following:

- Reference AMIs using Systems Manager parameters
- Reference the latest AMIs using Systems Manager public parameters

Reference AMIs using Systems Manager parameters

When you launch an instance using the EC2 launch instance wizard in the Amazon EC2 console, you can either select an AMI from the list, or you can select an AWS Systems Manager parameter that points to an AMI ID (described in this section). If you use automation code to launch your instances, you can specify the Systems Manager parameter instead of the AMI ID.

A Systems Manager parameter is a customer-defined key-value pair that you can create in Systems Manager Parameter Store. The Parameter Store provides a central store to externalize your application configuration values. For more information, see AWS Systems Manager Parameter Store in the *AWS Systems Manager User Guide*.

When you create a parameter that points to an AMI ID, make sure that you specify the data type as `aws:ec2:image`. Specifying this data type ensures that when the parameter is created or modified, the parameter value is validated as an AMI ID. For more information, see Native parameter support for Amazon Machine Image IDs in the *AWS Systems Manager User Guide*.

Contents

Use cases

When you use Systems Manager parameters to point to AMI IDs, it is easier for your users to select the correct AMI when launching instances. Systems Manager parameters can also simplify the maintenance of automation code.

Easier for users

If you require instances to be launched using a specific AMI, and the AMI is regularly updated, we recommend that you require your users to select a Systems Manager parameter to find the AMI. Requiring your users to select a Systems Manager parameter ensures that the latest AMI is used to launch instances.

For example, every month in your organization you might create a new version of your AMI that has the latest operating system and application patches. You also require your users to launch instances using the latest version of your AMI. To ensure that your users use the latest version, you can create a Systems Manager parameter (for example, `golden-ami`) that points to the correct AMI ID. Each time a new version of the AMI is created, you update the AMI ID value in the parameter so that it always points to the latest AMI. Your users don't have to know about the periodic updates to the AMI because they continue to select the same Systems Manager parameter each time. Using a Systems Manager parameter for your AMI makes it easier for them to select the correct AMI for an instance launch.

Simplify automation code maintenance

If you use automation code to launch your instances, you can specify the Systems Manager parameter instead of the AMI ID. If a new version of the AMI is created, you can change the AMI ID value in the parameter so that it points to the latest AMI. The automation code that references the parameter doesn't have to be modified each time a new version of the AMI is created. This simplifies the maintenance of the automation and helps to drive down deployment costs.

> **Note**
>
> Running instances are not affected when you change the AMI ID pointed to by the Systems Manager parameter.

Permissions

If you use Systems Manager parameters that point to AMI IDs in the launch instance wizard, you must add the following permissions to your IAM policy:

- `ssm:DescribeParameters` – Grants permission to view and select Systems Manager parameters.
- `ssm:GetParameters` – Grants permission to retrieve the values of the Systems Manager parameters.

You can also restrict access to specific Systems Manager parameters. For more information and example IAM policies, see Example: Use the EC2 launch instance wizard.

Limitations

AMIs and Systems Manager parameters are Region specific. To use the same Systems Manager parameter name across Regions, create a Systems Manager parameter in each Region with the same name (for example, `golden-ami`). In each Region, point the Systems Manager parameter to an AMI in that Region.

Launch an instance using a Systems Manager parameter

You can launch an instance using the console or the AWS CLI. Instead of specifying an AMI ID, you can specify an AWS Systems Manager parameter that points to an AMI ID.

To find an AMI using a Systems Manager parameter (console)

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. From the navigation bar, select the Region in which to launch your instances. You can select any Region that's available to you, regardless of your location.
3. From the console dashboard, choose **Launch instance**.
4. Under **Application and OS Images (Amazon Machine Image)**, choose **Browse more AMIs**.
5. Choose the arrow button to the right of the search bar, and then choose **Search by Systems Manager parameter**.
6. For **Systems Manager parameter**, select a parameter. The corresponding AMI ID appears below **Currently resolves to**.
7. Choose **Search**. The AMIs that match the AMI ID appear in the list.

8. Select the AMI from the list, and choose **Select**.

For more information about launching an instance using the launch instance wizard, see Launch an EC2 instance using the launch instance wizard in the console.

To launch an instance using an AWS Systems Manager parameter instead of an AMI ID (AWS CLI)

The following example uses the Systems Manager parameter `golden-ami` to launch an `m5.xlarge` instance. The parameter points to an AMI ID.

To specify the parameter in the command, use the following syntax: `resolve:ssm:/parameter-name`, where `resolve:ssm` is the standard prefix and `parameter-name` is the unique parameter name. Note that the parameter name is case-sensitive. Backslashes for the parameter name are only necessary when the parameter is part of a hierarchy, for example, `/amis/production/golden-ami`. You can omit the backslash if the parameter is not part of a hierarchy.

In this example, the `--count` and `--security-group` parameters are not included. For `--count`, the default is 1. If you have a default VPC and a default security group, they are used.

```
aws ec2 run-instances
    --image-id resolve:ssm:/golden-ami
    --instance-type m5.xlarge
    ...
```

To launch an instance using a specific version of an AWS Systems Manager parameter (AWS CLI)

Systems Manager parameters have version support. Each iteration of a parameter is assigned a unique version number. You can reference the version of the parameter as follows `resolve:ssm:parameter-name:version`, where `version` is the unique version number. By default, the latest version of the parameter is used when no version is specified.

The following example uses version 2 of the parameter.

In this example, the `--count` and `--security-group` parameters are not included. For `--count`, the default is 1 If you have a default VPC and a default security group, they are used.

```
aws ec2 run-instances
    --image-id resolve:ssm:/golden-ami:2
    --instance-type m5.xlarge
```

```
    ...
```

To launch an instance using a public parameter provided by AWS

Systems Manager provides public parameters for public AMIs provided by AWS. You can use the public parameters when launching instances to ensure that you're using the latest AMIs.

For more information, see Reference the latest AMIs using Systems Manager public parameters.

Reference the latest AMIs using Systems Manager public parameters

AWS Systems Manager provides public parameters for public AMIs maintained by AWS. You can use the public parameters when launching instances to ensure that you're using the latest AMIs. For example, the public parameter `/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-default-arm64` is available in all Regions and always points to the latest version of the Amazon Linux 2023 AMI for arm64 architecture in a given Region.

The public parameters are available from the following paths:

- **Linux** – `/aws/service/ami-amazon-linux-latest`
- **Windows** – `/aws/service/ami-windows-latest`

To view a list of all the Linux or Windows AMIs in the current AWS Region

Use the following get-parameters-by-path command to view a list of all the Linux or Windows AMIs in the current AWS Region. The value for the `--path` parameter is different for Linux and Windows.

For Linux:

```
aws ssm get-parameters-by-path \
    --path /aws/service/ami-amazon-linux-latest \
    --query "Parameters[].Name"
```

For Windows:

```
aws ssm get-parameters-by-path \
    --path /aws/service/ami-windows-latest \
    --query "Parameters[].Name"
```

To launch an instance using a public parameter

The following example specifies a Systems Manager public parameter for the image ID to launch an instance using the latest Amazon Linux 2023 AMI.

To specify the parameter in the command, use the following syntax: `resolve:ssm:`*`public-parameter`*, where `resolve:ssm` is the standard prefix and *`public-parameter`* is the path and name of the public parameter.

In this example, the `--count` and `--security-group` parameters are not included. For `--count`, the default is 1. If you have a default VPC and a default security group, they are used.

```
aws ec2 run-instances \
    --image-id resolve:ssm:/aws/service/ami-amazon-linux-latest/al2023-ami-kernel-default-x86_64 \
    --instance-type m5.xlarge \
    --key-name MyKeyPair
```

For more information, see Working with public parameters in the *AWS Systems Manager User Guide*.

For examples that use Systems Manager parameters, see Query for the latest Amazon Linux AMI IDs Using AWS Systems Manager Parameter Store and Query for the Latest Windows AMI Using AWS Systems Manager Parameter Store.

Paid AMIs in the AWS Marketplace for Amazon EC2 instances

A *paid AMI* is an AMI that is listed for sale in the AWS Marketplace. The AWS Marketplace is an online store where you can buy software that runs on AWS, including AMIs that you can use to launch your EC2 instance. The AWS Marketplace AMIs are organized into categories, such as Developer Tools, to enable you to find products to suit your requirements. For more information about AWS Marketplace, see the AWS Marketplace website.

You can purchase AMIs in the AWS Marketplace from a third party, including AMIs that come with service contracts from organizations such as Red Hat. You can also create an AMI and sell it in the AWS Marketplace to other Amazon EC2 users. Building a safe, secure, usable AMI for public consumption is a fairly straightforward process, if you follow a few simple guidelines. For information about how to create and use shared AMIs, see Understand shared AMI usage in Amazon EC2.

Launching an instance from a paid AMI is the same as launching an instance from any other AMI. No additional parameters are required. The instance is charged according to the rates set by the owner of the AMI, as well as the standard usage fees for the related web services, for example, the hourly rate for running an m5.small instance type in Amazon EC2. Additional taxes might also apply. The owner of the paid AMI can confirm whether a specific instance was launched using that paid AMI.

Important

Amazon DevPay is no longer accepting new sellers or products. AWS Marketplace is now the single, unified e-commerce platform for selling software and services through AWS. For information about how to deploy and sell software from AWS Marketplace, see Selling in AWS Marketplace. AWS Marketplace supports AMIs backed by Amazon EBS.

Contents

Sell your AMI in the AWS Marketplace

You can sell your AMI using AWS Marketplace. AWS Marketplace offers an organized shopping experience. Additionally, AWS Marketplace also supports AWS features such as Amazon EBS-backed AMIs, Reserved Instances, and Spot Instances.

For information about how to sell your AMI on the AWS Marketplace, see Selling in AWS Marketplace.

Find a paid AMI

A paid AMI is an Amazon Machine Image (AMI) that is available for purchase. A paid AMI also has a product code. You can find AMIs that are available for purchase in the AWS Marketplace.

Console

To find a paid AMI

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **AMIs**.
3. Choose **Public images** for the first filter.
4. Do one of the following:
   - If you know the product code, choose **Product code**, then **=**, and then enter the product code.
   - If you do not know the product code, in the Search bar, specify the following filter: **Owner alias=aws-marketplace**. Specify additional filters as needed.
5. Save the ID of the AMI.

AWS CLI

To find a paid AMI

Use the following describe-images command.

```
aws ec2 describe-images --owners aws-marketplace
```

The output includes a large number of images. You can specify filters to help you determine which AMI you need. After you find an AMI, specify its ID in the following command to get its product code.

```
aws ec2 describe-images \
    --image-ids ami-0abcdef1234567890 \
    --query Images[*].ProductCodes[].ProductCodeId
```

The following is example output.

```
[
    "cdef1234abc567def8EXAMPLE"
]
```

If you know the product code, you can filter the results by product code. This example returns the most recent AMI with the specified product code.

```
aws ec2 describe-images \
    --filters "Name=product-code,Values=cdef1234abc567def8EXAMPLE" \
    --query "sort_by(Images, &CreationDate)[-1].[ImageId]"
```

PowerShell

To find a paid AMI

Use the Get-EC2Image cmdlet.

```
Get-EC2Image -Owner aws-marketplace
```

The output includes a large number of images. You can specify filters to help you determine which AMI you need. After you find an AMI, specify its ID in the following command to get its product code.

```
(Get-EC2Image -ImageId ami-0abcdef1234567890).ProductCodes
```

The following is example output.

```
ProductCodeId             ProductCodeType
-------------             ---------------
cdef1234abc567def8EXAMPLE marketplace
```

If you know the product code, you can filter the results by product code. This example returns the most recent AMI with the specified product code.

```
(Get-EC2Image -Owner aws-marketplace -Filter @{"Name"="product-
code";"Value"="cdef1234abc567def8EXAMPLE"} | sort CreationDate -Descending | Select-
Object -First 1).ImageId
```

Purchase a paid AMI in the AWS Marketplace

You must sign up for (purchase) a paid AMI before you can launch an Amazon EC2 instance using the AMI.

Typically a seller of a paid AMI presents you with information about the AMI, including its price and a link where you can buy it. When you click the link, you're first asked to log into AWS, and then you can purchase the AMI.

Purchase a paid AMI using the console

You can purchase a paid AMI by using the Amazon EC2 launch wizard. For more information, see Launch an Amazon EC2 instance from an AWS Marketplace AMI.

Subscribe to a product using AWS Marketplace

To use the AWS Marketplace, you must have an AWS account. To launch instances from AWS Marketplace products, you must be signed up to use the Amazon EC2 service, and you must be subscribed to the product from which to launch the instance. You can use one of the following methods to subscribe to products in the AWS Marketplace:

- **AWS Marketplace website**: You can launch preconfigured software quickly with the 1-Click deployment feature. For more information, see AMI-based products in AWS Marketplace.
- **Amazon EC2 launch wizard**: You can search for an AMI and launch an instance directly from the wizard. For more information, see Launch an Amazon EC2 instance from an AWS Marketplace AMI.

Retrieve the AWS Marketplace product code from your instance

You can retrieve the AWS Marketplace product code for your instance using its instance metadata. If the instance has a product code, Amazon EC2 returns it. For more information about retrieving metadata, see Access instance metadata for an EC2 instance.

IMDSv2

Linux

Run the following command from your Linux instance.

```
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
    && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/product-codes
```

Windows

Run the following cmdlets from your Windows instance.

```
[string]$token = Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token-ttl-seconds"
 = "21600"} `
    -Method PUT -Uri http://169.254.169.254/latest/api/token
```

```
Invoke-RestMethod -Headers @{"X-aws-ec2-metadata-token" = $token} `
    -Method GET -Uri http://169.254.169.254/latest/meta-data/product-codes
```

IMDSv1

Linux

Run the following command from your Linux instance.

```
curl http://169.254.169.254/latest/meta-data/product-codes
```

Windows

Run the following command from your Windows instance.

```
Invoke-RestMethod -Uri http://169.254.169.254/latest/meta-data/product-codes
```

Use paid support for supported AWS Marketplace offerings

Amazon EC2 also enables developers to offer support for software (or derived AMIs). Developers can create support products that you can sign up to use. During sign-up for the support product, the developer gives you a product code, which you must then associate with your own AMI. This enables the developer to confirm that your instance is eligible for support. It also ensures that when you run instances of the product, you are charged according to the terms for the product specified by the developer.

Limitations

- After you set the product code attribute, it can't be changed or removed.
- You can't use a support product with Reserved Instances. You always pay the price that's specified by the seller of the support product.

AWS CLI

To associate a product code with your AMI