- South America (São Paulo)
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)

## Storage options for Amazon ECS tasks

Amazon ECS provides you with flexible, cost effective, and easy-to-use data storage options depending on your needs. Amazon ECS supports the following data volume options for containers:

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| Amazon Elastic Block Store (Amazon EBS) | Fargate, Amazon EC2, Amazon ECS Managed Instances | Linux, Windows (on Amazon EC2 only) | Can be persisted when attached to a standalone task. Ephemeral when attached to a task maintained by a service. | Amazon EBS volumes provide cost-effective, durable, high-performance block storage for data-inte nsive container ized workloads . Common use cases include transactional workloads such as databases, virtual desktops and root volumes, and throughpu t intensive workloads such as log processin g and ETL workloads. For |

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| | | | | more informati on, see Use Amazon EBS volumes with Amazon ECS. |

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| Amazon Elastic File System (Amazon EFS) | Fargate, Amazon EC2, Amazon ECS Managed Instances | Linux | Persistent | Amazon EFS volumes provide simple, scalable, and persisten t shared file storage for use with your Amazon ECS tasks that grows and shrinks automatically as you add and remove files. Amazon EFS volumes support concurrency and are useful for container ized applicati ons that scale horizontally and need storage functionalities like low latency, high throughpu t, and read-afte r-write consisten cy. Common use cases include workloads such as data analytics, |

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| | | | | media processing, content management, and web serving. For more information, see Use Amazon EFS volumes with Amazon ECS. |

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| Amazon FSx for Windows File Server | Amazon EC2 | Windows | Persistent | FSx for Windows File Server volumes provide fully managed Windows file servers that you can use to provision your Windows tasks that need persisten t, distribut ed, shared, and static file storage. Common use cases include .NET applications that might require local folders as persisten t storage to save applicati on outputs. Amazon FSx for Windows File Server offers a local folder in the container which allows for multiple |

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| | | | | containers to read-write on the same file system that's backed by a SMB Share. For more information, see Use FSx for Windows File Server volumes with Amazon ECS. |

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| Amazon FSx for NetApp ONTAP | Amazon EC2 | Linux | Persistent | Amazon FSx for NetApp ONTAP volumes provide fully managed NetApp ONTAP file systems that you can use to provision your Linux tasks that need persistent, high-performance, and feature-rich shared file storage. Amazon FSx for NetApp ONTAP supports NFS and SMB protocols and provides enterprise-grade features like snapshots, cloning, and data deduplication. Common use cases include high-performance computing workloads, content |

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| | | | | repositories, and applications requiring POSIX-compliant shared storage. For more information, see Mounting Amazon FSx for NetApp ONTAP file systems from Amazon ECS containers. |

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| Docker volumes | Amazon EC2 | Windows, Linux | Persistent | Docker volumes are a feature of the Docker container runtime that allow container s to persist data by mounting a directory from the file system of the host. Docker volume drivers (also referred to as plugins) are used to integrate container volumes with external storage systems. Docker volumes can be managed by third-par ty drivers or by the built in `local` driver. Common use cases for Docker volumes include providing persistent data volumes or |

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| | | | | sharing volumes at different locations on different containers on the same container instance. For more informati on, see Use Docker volumes with Amazon ECS. |

| Data volume | Supported capacity | Supported operating systems | Storage persistence | Use cases |
|---|---|---|---|---|
| Bind mounts | Fargate, Amazon EC2, Amazon ECS Managed Instances | Windows, Linux | Ephemeral | Bind mounts consist of a file or directory on the host, such as an Amazon EC2 instance or AWS Fargate, that is mounted onto a container . Common use cases for bind mounts include sharing a volume from a source container with other containers in the same task, or mounting a host volume or an empty volume in one or more containers. For more informati on, see Use bind mounts with Amazon ECS. |

## Use Amazon EBS volumes with Amazon ECS

Amazon Elastic Block Store (Amazon EBS) volumes provide highly available, cost-effective, durable, high-performance block storage for data-intensive workloads. Amazon EBS volumes can be used with Amazon ECS tasks for high throughput and transaction-intensive applications. For more information about Amazon EBS volumes, see Amazon EBS volumes in the *Amazon EBS User Guide*.

Amazon EBS volumes that are attached to Amazon ECS tasks are managed by Amazon ECS on your behalf. During standalone task launch, you can provide the configuration that will be used to attach one EBS volume to the task. During service creation or update, you can provide the configuration that will be used to attach one EBS volume per task to each task managed by the Amazon ECS service. You can either configure new, empty volumes for attachment, or you can use snapshots to load data from existing volumes.

**Note**

When you use snapshots to configure volumes, you can specify a `volumeInitializationRate`, in MiB/s, at which data is fetched from the snapshot to create volumes that are fully initialized in a predictable amount of time. For more information about volume initialization, see Initialize Amazon EBS volumes in the *Amazon EBS User Guide*. For more information about configuring Amazon EBS volumes, see Defer volume configuration to launch time in an Amazon ECS task definition and Specify Amazon EBS volume configuration at Amazon ECS deployment.

Volume configuration is deferred to launch time using the `configuredAtLaunch` parameter in the task definition. By providing volume configuration at launch time rather than in the task definition, you get to create task definitions that aren't constrained to a specific data volume type or specific EBS volume settings. You can then reuse your task definitions across different runtime environments. For example, you can provide more throughput during deployment for your production workloads than your pre-prod environments.

Amazon EBS volumes attached to tasks can be encrypted with AWS Key Management Service (AWS KMS) keys to protect your data. For more information see, Encrypt data stored in Amazon EBS volumes attached to Amazon ECS tasks.

To monitor your volume's performance, you can also use Amazon CloudWatch metrics. For more information about Amazon ECS metrics for Amazon EBS volumes, see Amazon ECS CloudWatch metrics and Amazon ECS Container Insights metrics.

Attaching an Amazon EBS volume to a task is supported in all commercial and China AWS Regions that support Amazon ECS.

### Supported operating systems and capacity

The following table provides the supported operating system and capacity configurations.

| Capacity | Linux | Windows |
|---|---|---|
| Fargate | Amazon EBS volumes are supported on platform version `1.4.0` or later (Linux). For more information, see Fargate platform versions for Amazon ECS. | Not supported |
| EC2 | Amazon EBS volumes are supported for tasks hosted on Nitro-based instances with Amazon ECS-optimized Amazon Machine Images (AMIs). For more informati on about instance types, see Instance types in the *Amazon EC2 User Guide*. <br><br> Amazon EBS volumes are supported on ECS-optim ized AMI `20231219` or later. For more information, see Retrieving Amazon ECS-Optimized AMI metadata. | Tasks hosted on Nitro-based instances with Amazon ECS-optimized Amazon Machine Images (AMIs). For more information about instance types, see Instance types in the *Amazon EC2 User Guide*. <br><br> Amazon EBS volumes are supported on ECS-optim ized AMI `20241017` or later. For more information, see Retrieving Amazon ECS-Optimized Windows AMI metadata. |
| Amazon ECS Managed Instances | Amazon EBS volumes are supported for tasks hosted on Amazon ECS Managed Instances on Linux. | Not supported |

### Considerations

Consider the following when using Amazon EBS volumes:

- You can't configure Amazon EBS volumes for attachment to Fargate Amazon ECS tasks in the `us-east-1c` Availability Zone.
- The magnetic (`standard`) Amazon EBS volume type is not supported for tasks hosted on Fargate. For more information about Amazon EBS volume types, see Amazon EBS volumes in the *Amazon EC2 User Guide*.
- An Amazon ECS infrastructure IAM role is required when creating a service or a standalone task that is configuring a volume at deployment. You can attach the AWS managed `AmazonECSInfrastructureRolePolicyForVolumes` IAM policy to the role, or you can use the managed policy as a guide to create and attach your own policy with permissions that meet your specific needs. For more information, see Amazon ECS infrastructure IAM role.
- You can attach at most one Amazon EBS volume to each Amazon ECS task, and it must be a new volume. You can't attach an existing Amazon EBS volume to a task. However, you can configure a new Amazon EBS volume at deployment using the snapshot of an existing volume.
- To use Amazon EBS volumes with Amazon ECS services, the deployment controller must be ECS. Both rolling and blue/green deployment strategies are supported when using this deployment controller.
- For a container in your task to write to the mounted Amazon EBS volume, you must run the container as a root user.
- Amazon ECS automatically adds the reserved tags `AmazonECSCreated` and `AmazonECSManaged` to the attached volume. If you remove these tags from the volume, Amazon ECS won't be able to manage the volume on your behalf. For more information about tagging Amazon EBS volumes, see Tagging Amazon EBS volumes. For more information about tagging Amazon ECS resources, see Tagging your Amazon ECS resources.
- Provisioning volumes from a snapshot of an Amazon EBS volume that contains partitions isn't supported.
- Volumes that are attached to tasks that are managed by a service aren't preserved and are always deleted upon task termination.
- You can't configure Amazon EBS volumes for attachment to Amazon ECS tasks that are running on AWS Outposts.

### Defer volume configuration to launch time in an Amazon ECS task definition

To configure an Amazon EBS volume for attachment to your task, you must specify the mount point configuration in your task definition and name the volume. You must also set `configuredAtLaunch` to `true` because Amazon EBS volumes can't be configured for attachment in the task definition. Instead, Amazon EBS volumes are configured for attachment during deployment.

To register the task definition by using the AWS Command Line Interface (AWS CLI), save the template as a JSON file, and then pass the file as an input for the `register-task-definition` command.

To create and register a task definition using the AWS Management Console, see Creating an Amazon ECS task definition using the console.

The following task definition shows the syntax for the `mountPoints` and `volumes` objects in the task definition. For more information about task definition parameters, see Amazon ECS task definition parameters for Fargate. To use this example, replace the *`user input placeholders`* with your own information.

### Linux

```
{
    "family": "mytaskdef",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/nginx/nginx:latest",
            "networkMode": "awsvpc",
           "portMappings": [
                {
                    "name": "nginx-80-tcp",
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp",
                    "appProtocol": "http"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "/mount/ebs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

### Windows

```
{
    "family": "mytaskdef",
     "memory": "4096",
     "cpu": "2048",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["EC2"]
    "containerDefinitions": [
        {
             "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 443,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "drive:\ebs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

`mountPoints`

Type: Object array

Required: No

The mount points for the data volumes in your container. This parameter maps to `Volumes` in the create-container Docker API and the `--volume` option to docker run.

Windows containers can mount whole directories on the same drive as `$env:ProgramData`. Windows containers cannot mount directories on a different drive, and mount points cannot be used across drives. You must specify mount points to attach an Amazon EBS volume directly to an Amazon ECS task.

`sourceVolume`

Type: String

Required: Yes, when `mountPoints` are used

The name of the volume to mount.

`containerPath`

Type: String

Required: Yes, when `mountPoints` are used

The path in the container where the volume will be mounted.

`readOnly`

Type: Boolean

Required: No

If this value is `true`, the container has read-only access to the volume. If this value is `false`, then the container can write to the volume. The default value is `false`.

For tasks that run on EC2 instances running the Windows operating system, leave the value as the default of `false`.

`name`

Type: String

Required: No

The name of the volume. Up to 255 letters (uppercase and lowercase), numbers, hyphens (`-`), and underscores (`_`) are allowed. This name is referenced in the `sourceVolume` parameter of the container definition `mountPoints` object.

`configuredAtLaunch`

Type: Boolean

Required: Yes, when you want to use attach an EBS volume directly to a task.

Specifies whether a volume is configurable at launch. When set to `true`, you can configure the volume when you run a standalone task, or when you create or update a service. When set to `false`, you won't be able to provide another volume configuration in the task definition. This parameter must be provided and set to `true` to configure an Amazon EBS volume for attachment to a task.

### Encrypt data stored in Amazon EBS volumes attached to Amazon ECS tasks

You can use AWS Key Management Service (AWS KMS) to make and manage cryptographic keys that protect your data. Amazon EBS volumes are encrypted at rest by using AWS KMS keys. The following types of data are encrypted:

- Data stored at rest on the volume
- Disk I/O
- Snapshots created from the volume
- New volumes created from encrypted snapshots

Amazon EBS volumes that are attached to tasks can be encrypted by using either a default AWS managed key with the alias `alias/aws/ebs`, or a symmetric customer managed key specified in the volume configuration. Default AWS managed keys are unique to each AWS account per AWS Region and are created automatically. To create a symmetric customer managed key, follow the steps in Creating symmetric encryption KMS keys in the *AWS KMS Developer Guide*.

You can configure Amazon EBS encryption by default so that all new volumes created and attached to a task in a specific AWS Region are encrypted by using the KMS key that you specify for your account. For more information about Amazon EBS encryption and encryption by default, see Amazon EBS encryption in the *Amazon EBS User Guide*.

#### Amazon ECS Managed Instances behavior

You encrypt Amazon EBS volumes by enabling encryption, either using encryption by default or by enabling encryption when you create a volume that you want to encrypt. For information about how to enable encryption by default (at the account-level, see Encryption by default in the *Amazon EBS User Guide*.

You can configure any combination of these keys. The order of precedence of KMS keys is as follows:

1. The KMS key specified in the volume configuration. When you specify a KMS key in the volume configuration, it overrides the Amazon EBS default and any KMS key that is specified at the account level.
2. The KMS key specified at the account level. When you specify a KMS key for cluster-level encryption of Amazon ECS managed storage, it overrides Amazon EBS default encryption but does not override any KMS key that is specified in the volume configuration.
3. Amazon EBS default encryption. Default encryption applies when you don't specify either a account-level KMS key or a key in the volume configuration. If you enable Amazon EBS encryption by default, the default is the KMS key you specify for encryption by default. Otherwise, the default is the AWS managed key with the alias `alias/aws/ebs`.

> **Note**
>
> If you set `encrypted` to `false` in your volume configuration, specify no account-level KMS key, and enable Amazon EBS encryption by default, the volume will still be encrypted with the key specified for Amazon EBS encryption by default.

**Non-Amazon ECS Managed Instances behavior**

You can also set up Amazon ECS cluster-level encryption for Amazon ECS managed storage when you create or update a cluster. Cluster-level encryption takes effect at the task level and can be used to encrypt the Amazon EBS volumes attached to each task running in a specific cluster by using the specified KMS key. For more information about configuring encryption at the cluster level for each task, see ManagedStorageConfiguration in the *Amazon ECS API reference*.

You can configure any combination of these keys. The order of precedence of KMS keys is as follows:

1. The KMS key specified in the volume configuration. When you specify a KMS key in the volume configuration, it overrides the Amazon EBS default and any KMS key that is specified at the cluster level.
2. The KMS key specified at the cluster level. When you specify a KMS key for cluster-level encryption of Amazon ECS managed storage, it overrides Amazon EBS default encryption but does not override any KMS key that is specified in the volume configuration.
3. Amazon EBS default encryption. Default encryption applies when you don't specify either a cluster-level KMS key or a key in the volume configuration. If you enable Amazon EBS

encryption by default, the default is the KMS key you specify for encryption by default. Otherwise, the default is the AWS managed key with the alias `alias/aws/ebs`.

**Note**

If you set `encrypted` to `false` in your volume configuration, specify no cluster-level KMS key, and enable Amazon EBS encryption by default, the volume will still be encrypted with the key specified for Amazon EBS encryption by default.

### Customer managed KMS key policy

To encrypt an EBS volume that's attached to your task by using a customer managed key, you must configure your KMS key policy to ensure that the IAM role that you use for volume configuration has the necessary permissions to use the key. The key policy must include the `kms:CreateGrant` and `kms:GenerateDataKey*` permissions. The `kms:ReEncryptTo` and `kms:ReEncryptFrom` permissions are necessary for encrypting volumes that are created using snapshots. If you want to configure and encrypt only new, empty volumes for attachment, you can exclude the `kms:ReEncryptTo` and `kms:ReEncryptFrom` permissions.

The following JSON snippet shows key policy statements that you can attach to your KMS key policy. Using these statements will provide access for Amazon ECS to use the key for encrypting the EBS volume. To use the example policy statements, replace the *`user input placeholders`* with your own information. As always, only configure the permissions that you need.

```
{
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:DescribeKey",
      "Resource":"*"
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": [
      "kms:GenerateDataKey*",
      "kms:ReEncryptTo",
      "kms:ReEncryptFrom"
      ],
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:CreateGrant",
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
        },
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
```

For more information about key policies and permissions, see Key policies in AWS KMS and AWS KMS permissions in the *AWS KMS Developer Guide*. For troubleshooting EBS volume attachment issues related to key permissions, see Troubleshooting Amazon EBS volume attachments to Amazon ECS tasks .

### Specify Amazon EBS volume configuration at Amazon ECS deployment

After you register a task definition with the `configuredAtLaunch` parameter set to `true`, you can configure an Amazon EBS volume at deployment when you run a standalone task, or when you create or update a service. For more information about deferring volume configuration to launch time using the `configuredAtLaunch` parameter, see Defer volume configuration to launch time in an Amazon ECS task definition.

To configure a volume, you can use the Amazon ECS APIs, or you can pass a JSON file as input for the following AWS CLI commands:

- `run-task` to run a standalone ECS task.
- `start-task` to run a standalone ECS task in a specific container instance. This command is not applicable for Fargate tasks.
- `create-service` to create a new ECS service.
- `update-service` to update an existing service.

**Note**

For a container in your task to write to the mounted Amazon EBS volume, you must run the container as a root user.

You can also configure an Amazon EBS volume by using the AWS Management Console. For more information, see Running an application as an Amazon ECS task, Creating an Amazon ECS rolling update deployment, and Updating an Amazon ECS service.

The following JSON snippet shows all the parameters of an Amazon EBS volume that can be configured at deployment. To use these parameters for volume configuration, replace the *`user input placeholders`* with your own information. For more information about these parameters, see Volume configurations.

```
"volumeConfigurations": [
        {
            "name": "ebs-volume",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "volumeType": "gp3",
                "sizeInGiB": 10,
                "snapshotId": "snap-12345",
                "volumeInitializationRate":100,
                "iops": 3000,
                "throughput": 125,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "key1",
                                "value": "value1"
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                 "terminationPolicy": {
                    "deleteOnTermination": true//can't be configured for service-
managed tasks, always true
                },
                "filesystemType": "ext4"
            }
        }
    ]
```

**Important**

Ensure that the `volumeName` you specify in the configuration is the same as the `volumeName` you specify in your task definition.

For information about checking the status of volume attachment, see Troubleshooting Amazon EBS volume attachments to Amazon ECS tasks . For information about the Amazon ECS infrastructure AWS Identity and Access Management (IAM) role necessary for EBS volume attachment, see Amazon ECS infrastructure IAM role.

The following are JSON snippet examples that show the configuration of Amazon EBS volumes. These examples can be used by saving the snippets in JSON files and passing the files as parameters (using the `--cli-input-json file://filename` parameter) for AWS CLI commands. Replace the *`user input placeholders`* with your own information.

### Configure a volume for a standalone task

The following snippet shows the syntax for configuring Amazon EBS volumes for attachment to a standalone task. The following JSON snippet shows the syntax for configuring the `volumeType`, sizeInGiB, encrypted, and kmsKeyId settings. The configuration specified in the JSON file is used to create and attach an EBS volume to the standalone task.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "volumeConfigurations": [
        {
            "name": "datadir",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
   ]
}
```

### Configure a volume at service creation

The following snippet shows the syntax for configuring Amazon EBS volumes for attachment to tasks managed by a service. The volumes are sourced from the snapshot specified using the snapshotId parameter at a rate of 200 MiB/s. The configuration specified in the JSON file is used to create and attach an EBS volume to each task managed by the service.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
              "snapshotId": "snap-12345",
              "volumeInitializationRate": 200
            }
        }
```

```
    ]
}
```

### Configure a volume at service update

The following JSON snippet shows the syntax for updating a service that previously did not have Amazon EBS volumes configured for attachment to tasks. You must provide the ARN of a task definition revision with `configuredAtLaunch` set to `true`. The following JSON snippet shows the syntax for configuring the `volumeType`, `sizeInGiB`, `throughput`, and `iops`, and `filesystemType` settings. This configuration is used to create and attach an EBS volume to each task managed by the service.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "service": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
               "volumeType": "gp3",
                "sizeInGiB": 100,
                 "iops": 3000,
                "throughput": 125,
                "filesystemType": "ext4"
            }
        }
   ]
}
```

### Configure a service to no longer utilize Amazon EBS volumes

The following JSON snippet shows the syntax for updating a service to no longer utilize Amazon EBS volumes. You must provide the ARN of a task definition with `configuredAtLaunch` set to `false`, or a task definition without the `configuredAtLaunch` parameter. You must also provide an empty `volumeConfigurations` object.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "service": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": []
}
```

### Termination policy for Amazon EBS volumes

When an Amazon ECS task terminates, Amazon ECS uses the `deleteOnTermination` value to determine whether the Amazon EBS volume that's associated with the terminated task should be deleted. By default, EBS volumes that are attached to tasks are deleted when the task is terminated. For standalone tasks, you can change this setting to instead preserve the volume upon task termination.

> **Note**
>
> Volumes that are attached to tasks that are managed by a service are not preserved and are always deleted upon task termination.

### Tag Amazon EBS volumes

You can tag Amazon EBS volumes by using the `tagSpecifications` object. Using the object, you can provide your own tags and set propagation of tags from the task definition or the service, depending on whether the volume is attached to a standalone task or a task in a service. The maximum number of tags that can be attached to a volume is 50.

> **Important**
>
> Amazon ECS automatically attaches the `AmazonECSCreated` and `AmazonECSManaged` reserved tags to an Amazon EBS volume. This means you can control the attachment of a maximum of 48 additional tags to a volume. These additional tags can be user-defined, ECS-managed, or propagated tags.

If you want to add Amazon ECS-managed tags to your volume, you must set `enableECSManagedTags` to `true` in your `UpdateService`, `CreateService`,`RunTask` or `StartTask` call. If you turn on Amazon ECS-managed tags, Amazon ECS will tag the volume automatically with cluster and service information (`aws:ecs:`*`clusterName`* and aws:ecs:*serviceName*). For more information about tagging Amazon ECS resources, see Tagging your Amazon ECS resources.

The following JSON snippet shows the syntax for tagging each Amazon EBS volume that is attached to each task in a service with a user-defined tag. To use this example for creating a service, replace the *user input placeholders* with your own information.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "enableECSManagedTags": true,
   "volumeConfigurations": [
        {
            "name": "datadir",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                 "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "key1",
                                "value": "value1"
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn":"arn:aws:iam:1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
  "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
   ]
}
```

**Important**

You must specify a `volume` resource type to tag Amazon EBS volumes.

### Performance of Amazon EBS volumes for Fargate on-demand tasks

The baseline Amazon EBS volume IOPS and throughput available for a Fargate on-demand task depends on the total CPU units you request for the task. If you request 0.25, 0.5, or 1 virtual CPU unit (vCPU) for your Fargate task, we recommend that you configure a General Purpose SSD volume (`gp2` or `gp3`) or a Hard Disk Drive (HDD) volume (`st1` or `sc1`). If you request more than 1 vCPU for your Fargate task, the following baseline performance limits apply to an Amazon EBS volume attached to the task. You may temporarily get higher EBS performance than the following limits. However, we recommend that you plan your workload based on these limits.

| CPU units requested (in vCPUs) | Baseline Amazon EBS IOPS(16 KiB I/O) | Baseline Amazon EBS Throughput (in MiBps, 128 KiB I/O) | Baseline bandwidth (in Mbps) |
|---|---|---|---|
| 2 | 3,000 | 75 | 360 |
| 4 | 5,000 | 120 | 1,150 |
| 8 | 10,000 | 250 | 2,300 |
| 16 | 15,000 | 500 | 4,500 |

**Note**

When you configure an Amazon EBS volume for attachment to a Fargate task, the Amazon EBS performance limit for Fargate task is shared between the task's ephemeral storage and the attached volume.

### Performance of Amazon EBS volumes for EC2 tasks

Amazon EBS provides volume types, which differ in performance characteristics and price, so that you can tailor your storage performance and cost to the needs of your applications. For information about performance, including IOPS per volume and throughput per volume, see Amazon EBS volume types in the *Amazon Elastic Block Store User Guide*.

### Performance of Amazon EBS volumes for Amazon ECS Managed Instances tasks

Amazon EBS provides volume types, which differ in performance characteristics and price, so that you can tailor your storage performance and cost to the needs of your applications. For information about performance, including IOPS per volume and throughput per volume, see Amazon EBS volume types in the *Amazon Elastic Block Store User Guide*.

### Troubleshooting Amazon EBS volume attachments to Amazon ECS tasks

You might need to troubleshoot or verify the attachment of Amazon EBS volumes to Amazon ECS tasks.

#### Check volume attachment status

You can use the AWS Management Console to view the status of an Amazon EBS volume's attachment to an Amazon ECS task. If the task starts and the attachment fails, you'll also see a status reason that you can use to troubleshoot. The created volume will be deleted and the task will be stopped. For more information about status reasons, see Status reasons for Amazon EBS volume attachment to Amazon ECS tasks.

**To view a volume's attachment status and status reason using the console**

1. Open the console at https://console.aws.amazon.com/ecs/v2.
2. On the **Clusters** page, choose the cluster that your task is running in. The cluster's details page appears.
3. On the cluster's details page, choose the **Tasks** tab.
4. Choose the task that you want to view the volume attachment status for. You might need to use **Filter desired status** and choose **Stopped** if the task you want to examine has stopped.
5. On the task's details page, choose the **Volumes** tab. You will be able to see the attachment status of the Amazon EBS volume under **Attachment status**. If the volume fails to attach to the task, you can choose the status under **Attachment status** to display the cause of the failure.

You can also view a task's volume attachment status and associated status reason by using the DescribeTasks API.

### Service and task failures

You might encounter service or task failures that aren't specific to Amazon EBS volumes that can affect volume attachment. For more information, see

- Service event messages
- Stopped task error codes
- API failure reasons

### Status reasons for Amazon EBS volume attachment to Amazon ECS tasks

Use the following reference to fix issues that you might encounter in the form of status reasons in the AWS Management Console when you configure Amazon EBS volumes for attachment to Amazon ECS tasks. For more information on locating these status reasons in the console, see Check volume attachment status.

ECS was unable to assume the configured ECS Infrastructure Role 'arn:aws:iam::*`111122223333`*:role/*`ecsInfrastructureRole`*'. Please verify that the role being passed has the proper trust relationship with Amazon ECS

This status reason appears in the following scenarios.

- You provide an IAM role without the necessary trust policy attached. Amazon ECS can't access the Amazon ECS infrastructure IAM role that you provide if the role doesn't have the necessary trust policy. The task can get stuck in the `DEPROVISIONING` state. For more information about the necessary trust policy, see Amazon ECS infrastructure IAM role.
- Your IAM user doesn't have permission to pass the Amazon ECS infrastructure role to Amazon ECS. The task can get stuck in the `DEPROVISIONING` state. To avoid this problem, you can attach the `PassRole` permission to your user. For more information, see Amazon ECS infrastructure IAM role.
- Your IAM role doesn't have the necessary permissions for Amazon EBS volume attachment. The task can get stuck in the `DEPROVISIONING` state. For more information about the specific permissions necessary for attaching Amazon EBS volumes to tasks, see Amazon ECS infrastructure IAM role.

> **Note**
>
> You may also see this error message due to a delay in role propagation. If retrying to use the role after waiting for a few minutes doesn't fix the issue, you might have misconfigured the trust policy for the role.

ECS failed to set up the EBS volume. Encountered IdempotentParameterMismatch"; "The client token you have provided is associated with a resource that is already deleted. Please use a different client token."

The following AWS KMS key scenarios can lead to an `IdempotentParameterMismatch` message appearing:

- You specify a KMS key ARN, ID, or alias that isn't valid. In this scenario, the task might appear to launch successfully, but the task eventually fails because AWS authenticates the KMS key asynchronously. For more information, see Amazon EBS encryption in the *Amazon EC2 User Guide*.
- You provide a customer managed key that lacks the permissions that allow the Amazon ECS infrastructure IAM role to use the key for encryption. To avoid key-policy permission issues, see the example AWS KMS key policy in Data encryption for Amazon EBS volumes.

You can set up Amazon EventBridge to send Amazon EBS volume events and Amazon ECS task state change events to a target, such as Amazon CloudWatch groups. You can then use these events to identify the specific customer managed key related issue that affected volume attachment. For more information, see

- How can I create a CloudWatch log group to use as a target for an EventBridge rule? on AWS re:Post.
- Task state change events.
- Amazon EventBridge events for Amazon EBS in the *Amazon EBS User Guide*.

ECS timed out while configuring the EBS volume attachment to your Task.

The following file system format scenarios result in this message.

- The file system format that you specify during configuration isn't compatible with the task's operating system.
- You configure an Amazon EBS volume to be created from a snapshot, and the snapshot's file system format isn't compatible with the task's operating system. For volumes created from

a snapshot, you must specify the same filesystem type that the volume was using when the snapshot was created.

You can utilize the Amazon ECS container agent logs to troubleshoot this message for EC2 tasks. For more information, see Amazon ECS log file locations and Amazon ECS log collector.

## Use Amazon EFS volumes with Amazon ECS

Amazon Elastic File System (Amazon EFS) provides simple, scalable file storage for use with your Amazon ECS tasks. With Amazon EFS, storage capacity is elastic. It grows and shrinks automatically as you add and remove files. Your applications can have the storage they need and when they need it.

You can use Amazon EFS file systems with Amazon ECS to export file system data across your fleet of container instances. That way, your tasks have access to the same persistent storage, no matter the instance on which they land. Your task definitions must reference volume mounts on the container instance to use the file system.

For a tutorial, see Configuring Amazon EFS file systems for Amazon ECS using the console.

### Considerations

Consider the following when using Amazon EFS volumes:

- For tasks that run on EC2, Amazon EFS file system support was added as a public preview with Amazon ECS-optimized AMI version `20191212` with container agent version 1.35.0. However, Amazon EFS file system support entered general availability with Amazon ECS-optimized AMI version `20200319` with container agent version 1.38.0, which contained the Amazon EFS access point and IAM authorization features. We recommend that you use Amazon ECS-optimized AMI version `20200319` or later to use these features. For more information, see Amazon ECS-optimized Linux AMIs.

**Note**

If you create your own AMI, you must use container agent 1.38.0 or later, `ecs-init` version 1.38.0-1 or later, and run the following commands on your Amazon EC2 instance to enable the Amazon ECS volume plugin. The commands are dependent on whether you're using Amazon Linux 2 or Amazon Linux as your base image.

Amazon Linux 2